The Institute For Research In Cognitive Science

A Computational Approach to Aspectual Composition (Ph.D. Dissertation)

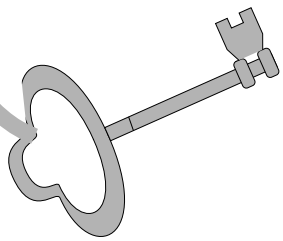
by

Michael White

University of Pennsylvania
3401 Walnut Street, Suite 400C
Philadelphia, PA 19104-6228

August 1994

Site of the NSF Science and Technology Center for Research in Cognitive Science



University of Pennsylvania
Founded by Benjamin Franklin in 1740

IRCS Report 94-11

A Computational Approach to Aspectual Composition

Michael White

A Dissertation
in
Computer and Information Science

Presented to the Faculties of the University of Pennsylvania
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy

1994

This research was partially funded by a National Science Foundation Graduate Fellowship, ARO grants DAAG29-84-K-0061 and DAAL03-89-C0031PRI, DARPA grants N00014-85-K0018 and N00014-90-J-1863, NSF grants MCS-82-19196 and IRI 90-16592, and Ben Franklin Grant 91S.3078C-1 at the University of Pennsylvania.

Acknowledgments

In the course of this dissertation project, I reached a point where I had to just strap on my rollerblades and scream around the block in frustration. If you had told me then that I would soon attain a certain level of contentment with—and even pride in—the dissertation, I would never have believed it.

For helping to bring about this scenario, unlikely as it seemed to me, my heartfelt thanks go out to Mark Steedman, my advisor, whose unfailing enthusiasm and encouragement, not to mention his wisdom and insight, proved to be invaluable. I would also like to thank the members of my committee: Norm Badler, Aravind Joshi, Bonnie Webber, and especially Dick Oehrle, my external examiner, whose timely and detailed comments were tremendously helpful. Many others deserve my gratitude for their input, including the members of the Epistemic Glee Club, the AnimNL group, and the CLiFF group, much of the CIS and Linguistics Faculty, various IRCS visitors, and other friends in strategic locations throughout the world.

Perhaps even larger is the collection of people and other animals who helped me to make it through this process. Certainly I owe a great debt to Cosmo, the alarm cat, who helped me wake up almost every morning. And of course there is Libby Levison, who took it upon herself to replenish my water supply when, during my defense, she noticed that I kept looking longingly at my mug without drinking from it. And what would I have possibly done without George and Nancy at Home Sweet Homebrew? The list could go on and on, but instead I will simply extend a very special thanks to all those whose names should appear here, including, but not limited to, Breck Baldwin, Eric Brill, Chris Geib and Charlie Ortiz, the members of my WPE group; Jamie Henderson, Rich Pito, and Guy Whitten, my lifting and otherwise buddies; Barbara Di Eugenio, my officemate; Abigail Gertner and Mark Little, my roommates; and especially Christy Doran, my consultant and confidante.

Last, but of course most, I would like to thank my sister, Karen, and my parents, Betty and Ray.

Abstract

In recent years, it has become common in the linguistics and philosophy literature to assume that **events** and **processes** are ontologically distinct entities, on a par with **objects** and **substances**. At the same time, the idea that time-based (episodic) knowledge should be represented as a collection of interrelated **eventualities** has gained increasing acceptance in the computational linguistics and artificial intelligence literature.

Contrary to what one might expect, a search through the prior literature in linguistics and philosophy reveals no account in which these sortal distinctions play a direct role in adequately explaining the problem of **aspectual composition** and the closely related **imperfective paradox**. In the computational linguistics and artificial intelligence literature, moreover, relatively little attention has been paid to either problem.

In the first part of the dissertation, I investigate the hypothesis that the parallel ontological distinctions introduced above may be directly employed in an explanatory formal account of the problem of aspectual composition and the imperfective paradox. In so doing, I develop a synthesis of proposals by Hinrichs (1985), Krifka (1989; 1992) and Jackendoff (1991) which makes correct predictions in many cases not considered by these authors. In particular, the account is the first to adequately explain the syntactic and semantic behavior of **non-individuating accomplishment expressions**, such as *Jack pour some amount of wort into the carboy*, which are too vague to individuate a single event but nevertheless behave like other Vendlerian accomplishments.

In the second part of the dissertation, I explore the potential computational applications of the linguistic account, by way of two case studies. In the first one, I follow Moens (1987) in showing how a **calculus of eventualities** can facilitate the implementation of a simple statement verifier which allows for a much greater range of natural language queries than is usually the case with temporal databases. In the second, more preliminary study, I examine the relevance of the model-theoretic analysis to discourse interpretation, within the context of devising a program which produces simple microworld animations using short narrative descriptions as input specifications.

Contents

List of Figures

Chapter 1

Introduction

At an intuitive level, English provides us with expressions for describing **processes** that may go on for indeterminate periods, e.g. the tenseless sentence *Jack pour wort into the carboy*; it also provides us with expressions for describing **events** with determinate endpoints, e.g. *Jack fill the carboy with wort.*[1] That these two types of verbal expressions are in fact distinct becomes most evident when we look at their interaction with measure adverbials, e.g. *for ten seconds.* In the case of process expressions, such measure phrases are unproblematic: cf. *Jack poured wort into the carboy for ten seconds.* In the case of event expressions, however, measure adverbials are anomalous: cf. ** Jack filled the carboy with wort for ten seconds.*[2]

As we know from the works of Dowty (1972), Verkuyl (1972) and many since, English is not so inflexible as to localize this distinction to the verb alone, contra Vendler (1967).[3] Given different arguments, such as a direct object with an amount phrase, a verb like *pour* yields an event expression: *Jack pour twenty liters of wort into the carboy.* This interaction is the problem of **aspectual composition**. Contrariwise, given modifiers such as the progressive, a verb like *fill* yields an expression useful for describing an ongoing process: *Jack be filling the carboy with wort.* Curiously, such expressions may be employed without entailing that the endpoints are ever actually reached—note that the past tense

[1] In case the reader is unfamiliar with these terms, wort is an infusion of malt fermented to make beer, and a carboy is a large glass bottle used to store liquids. Since these particulars do not matter for present purposes, the reader is invited to substitute his or her own favorite liquid and container in these examples and those that follow.

[2] By the use of '*', I intend to indicate that this expression fails to have a single event reading, i.e. a reading which asserts the occurrence of an event of duration ten seconds in which Jack fills the carboy with wort; as such, '*' does not exclude the possibility of iterative or partitive readings.

[3] While examining other languages is beyond the scope of this work, it is important to mention that English is not at all unusual in this regard. Also, note that for descriptive purposes, I will follow recent tradition in continuing to employ Vendler's well-known classificatory terms at the level of verbal expressions.

version of the preceding expression may be true even in the case where Jack is interrupted and never finishes filling the carboy. This phenomenon is known as the **imperfective paradox**.

In recent years it has become common in the the linguistics and philosophy literature to view the distinction between event and process expressions in the verbal domain as paralleling the more readily apparent distinction between count and mass expressions in the nominal domain. Taking this intuitive analogy quite seriously, a number of researchers—including Hinrichs (1985), Bach (1986), Link (1987) and Jackendoff (1991)—have suggested pursuing an approach where events and processes are considered ontologically distinct entities, on a par with objects and substances.

One might expect that an adequate formalization of these entities would directly yield an explanation of the problem of aspectual composition and the closely related imperfective paradox. It may come as a surprise then to find, as we shall when we review prior approaches in the next chapter, that no such account has been developed to date—though several proposals do come close to providing such a direct explanation, namely those of Hinrichs (1985), Krifka (1989; 1992) and Jackendoff (1991). As we shall see, while both Hinrichs and Krifka do make use of these ontological distinctions in their accounts of aspectual composition, they do so only partially or incidentally; moreover, though Jackendoff (1991) does seem to suggest such an approach, his account is not sufficiently formalized to make precise predictions, as Verkuyl and Zwarts (1992) observe.

In the first part of the dissertation, I investigate the hypothesis that the parallel ontological distinctions introduced above, if properly understood, may be directly employed in an explanatory formal account of these two problems. In so doing, I develop a synthesis of the accounts of Hinrichs, Krifka and Jackendoff which makes correct predictions in many cases not considered by these authors.[4]

The account follows the Montagovian tradition of translating English expressions into an unambiguous formal language with a well-defined semantics. In the present case, an **order-sorted** logic is employed as the translation language, which permits the formal analysis of both logical entailments and semantic anomalies arising from sort incompatibilities. For example, in the case of spatial *for*-adverbials—which have been largely ignored in the literature—the account successfully explains the striking contrast between the pair *Jack ran two miles along the river* and *Jack ran along the river for two miles*, on the one hand, and the pair *Jack ran two miles to the bridge* and ** Jack ran to the bridge for two miles*, on the other: in the first pair, the two sentences are shown to be logically equivalent; in the second pair, in contrast, the latter sentence (unlike the former one) is shown to be semantically anomalous, due to a sortal clash.

A hallmark of the present account is that it is the first to adequately explain the problem of **non-individuating accomplishment expressions**, such as

[4] The exact connection to these prior approaches is made explicit in section 2.5.

Jack pour some amount of wort into the carboy, which are too vague to individuate a single event but nevertheless behave like other Vendlerian accomplishments. These expressions are troublesome for the analyses of Dowty, Hinrichs and Krifka, where tests for individuation are assumed to correctly identify the class of accomplishment expressions. Because the present account does not rely on such tests, these expressions are unproblematic. To take a case in point, consider the pair *Jack ran for ten minutes* and *Jack ran some distance in ten minutes*. Despite the fact that the latter sentence contains the non-individuating expression *Jack run some distance*, the present account is able to explain not only why *Jack ran for ten minutes* entails *Jack ran some distance in ten minutes*, but also why different temporal adverbials are appropriate in the two cases. Moreover, as we shall see, the assumptions underlying the present explanation are much the same as those independently motivated by the nominal counterpart of this problem.

A second distinguishing feature of the account is its use of **aspectual type coercions**, in the spirit of Moens and Steedman (1988).[5] By formalizing the analyses they suggest for the imperfective paradox, iterated readings of *for*-adverbials, and other interesting cases, the coverage of the account is greatly extended. As we shall see, however, closer inspection of these coercions reveals that they overgenerate in strange ways if allowed to apply freely; to control this overgeneration, the present account simply requires all coercions to be lexicalized.

In the second part of the dissertation, I explore the potential computational applications of the linguistic account. In recent years, the computational linguistics and artificial intelligence literature has seen a gradual shift in the way time-related knowledge is predominantly represented: increasingly, the idea that such episodic information should be represented as a collection of interrelated **eventualities** (Bach, 1986) has replaced the older view in which the world is represented as a series of snapshots and state-changing functions. In this literature, however, relatively little attention has been paid to the problem of aspectual composition and the imperfective paradox; as such, the goal of this part of the thesis is to demonstrate the suitability of the present approach to these problems to computational purposes.

Part II consists of two separate case studies. In the first one, I present **a calculus of eventualities** which covers a subset of the model-theoretic account. Following Moens (1987), I show that the calculus facilitates the implementation of a simple statement verifier which allows for a much greater range of natural language queries than is usually the case with temporal databases; going beyond Moens, I also address the problem of aspectual composition and resolve some problematic aspects of his treatment of the imperfective paradox.

In the second, more preliminary case study, I examine the relevance of the

[5] As we shall see in the next chapter, these coercions are much the same as Jackendoff's independently developed **rules of construal**.

model-theoretic analysis to discourse interpretation. This study is set within the context of the following task: to devise a program which allows a short narrative description within a small English fragment to be used as a specification for a simple microworld animation. Besides serving as a test upon the adequacy of the spatio-temporal semantics, this task context also enables a detailed examination of the communicative functionality of aspect in situated narrative discourse. The findings of the study are illustrated by a program implemented using a constraint-based abductive interpretation procedure falling within the **interpretation as abduction** paradigm set forth by Hobbs et al. (1993).

The contributions of the dissertation and the questions it leaves open are summarized in greater detail in sections 4.9, 5.4 and 6.4. Chapter 7 reviews these contributions and elaborates upon the relationship between parts I and II, focusing on how the computational studies presented in part II informed the development of the model-theoretic account in part I. Finally, the code for the two case studies appears in appendices A and B.

Part I

A Sortal Approach to Aspectual Composition

Chapter 2

Issues and Desiderata

Since the literature on aspect and related issues is so vast, I have not attempted an exhaustive review in this chapter.[1] Instead, I first focus rather narrowly on the relevant parts of several recent works, with an eye towards identifying the role of the ontology in each of them; this review is found in sections 1 and 2. In sections 3 and 4, I then discuss two additional issues of particular interest in this context, namely, aspectual type coercion and individuation. In section 5, I present an interim summary, and sketch out the present account. Finally, in sections 6 and 7, I set forth an explicit list of syntactic and semantic desiderata which the present account is intended to meet.

2.1 Aspectual Composition

The problem of aspectual composition is to systematically account for the aspectual effects of various verbal arguments and modifiers. These effects were first pointed out by both Verkuyl (1972) and Dowty (1972), demonstrating the inadequacy of Vendler's (1967) quadpartition of lexical verbs. In what follows I will concentrate on how mass terms, bare plurals and locative modifiers interact with the lexical semantics of various classes of verbs.

Aspectual effects are detected using a variety of syntactic (cooccurrence) and semantic (entailment) tests. On the syntactic side, I will follow Verkuyl (1989) in emphasizing the **temporal adverbial test**. This test consists of determining whether a verbal expression felicitously cooccurs with a *for*-adverbial or an *in*-adverbial under a single event reading. For example, consider the contrast between *pour* and *fill* below:

(2.1) (a) Jack poured wort into the carboy $\left\{ \begin{array}{c} * \text{ in} \\ \text{for} \end{array} \right\}$ ten seconds.

[1] See (Dowty, 1979), (Binnick, 1991) and references therein for a historical review.

(b) Jack filled the carboy with wort $\left\{ \begin{array}{c} \text{in} \\ *\,\text{for} \end{array} \right\}$ ten seconds.

Here we see that the appropriate temporal adverbial is determined by the choice of verb. As an example of the effect of mass terms, we may contrast (2.1a) with (2.2):

(2.2) Jack poured twenty liters of wort into the carboy $\left\{ \begin{array}{c} \text{in} \\ *\,\text{for} \end{array} \right\}$ ten seconds.

In (2.2) we observe that the addition of a measure phrase in the object NP switches the appropriate temporal adverbial. As for bare plurals, we may contrast (2.1b) with (2.3):

(2.3) Jack filled carboys with wort $\left\{ \begin{array}{c} *\,\text{in} \\ \text{for} \end{array} \right\}$ ten minutes.

Finally, as an example of the effect of locative modifiers, consider (2.4):

(2.4) (a) Jack ran along the river $\left\{ \begin{array}{c} *\,\text{in} \\ \text{for} \end{array} \right\}$ ten minutes.

(b) Jack ran to the museum $\left\{ \begin{array}{c} \text{in} \\ *\,\text{for} \end{array} \right\}$ ten minutes.

Turning now to the semantic tests, we find that matters become somewhat trickier. Many of these tests involve the imperfective paradox; these will be discussed in the next subsection. Here I will focus on the property of **homogeneous reference**, which dates back at least to Quine (1960). This property concerns the way in which an entity as a whole stands in relation to its subparts. For example, if we consider an event of Jack running along the river, we realize that all its subevents are also events of Jack running along the river; this is the property of **divisive reference**. In the other direction, if we consider a collection of subevents of Jack running along the river, we realize that their sum is also an event of Jack running along the river; this is the property of **cumulative reference**.[2] Taken together, these two properties make up the homogeneous reference property.

The notion of homogeneous reference is not without problems. The most notorious one is the **minimal parts problem**, which concerns the failure of divisive reference with subparts smaller than a certain size; for example, a subevent of Jack lifting his leg is perhaps too small to be considered an event of Jack running along the river. Further subtleties concerning homogeneous reference will be discussed in section 2.4. For present purposes, it suffices to observe

[2] An interesting question which is often ignored is whether such a collection of events must be topologically connected; we will return to this point in section 4.3.

that this property is often useful in distinguishing Vendlerian activities from accomplishments—clearly, most subevents of Jack running to the museum are only events of Jack running partway to the museum.

Devising an adequate theory-neutral semantic test for this property has proved difficult. Dowty (1979, p. 57) suggests the following semantic test:

(2.5) If ϕ is an activity verb, then *x ϕed for y time* entails that at any time during *y*, *x ϕed* was true. If ϕ is an accomplishment verb, then *x ϕed for y time* does not entail that *x ϕed* was true during any time within *y* at all.

However, as Verkuyl (1989) points out, (2.5) must be rephrased: one cannot base entailments on anomalous sentences like * *Jack filled the carboy for ten minutes*, which fail to have single event readings. Furthermore, though Dowty (1979, p. 171) mentions that (2.5) suffers from the minimal parts problem, he does not return to fix it. As a variation on Dowty's test, let us consider the following one:

(2.6) A sentential expression ϕ is **downward entailing** if ϕ *for/in n time* entails ϕ *for/in m time*, where *m* is less than *n* and of a reasonable size for ϕ.

This test differs from Dowty's insofar as it allows for the appropriate temporal adverbial and adds a facile notion of granularity. According to (2.6), *Jack runs along the river* turns out downward entailing: if Jack ran along the river for ten minutes, he must have run along the river for nine minutes, eight minutes, and so on; whether or not he ran along the river for 100 milliseconds does not matter. Conversely, *Jack runs to the museum* does not turn out downward entailing: if Jack ran to the museum in exactly ten minutes, then he cannot have run to the museum in nine minutes.

With this brief introduction to the problem of aspectual composition behind us, we turn now to prior approaches.

2.1.1 Dowty (1979)

Dowty (1979) presents a Montagovian treatment of a sizable fragment of English. Working within an interval-based semantics, he proposes a quantificational analysis of *for*-adverbials in order to account for both the temporal adverbial test and the downwards entailments test. While he does not explicitly address the problem of aspectual composition, except in the case of locative modifiers, he does outline an approach to mass terms and bare plurals which is later taken up in Hinrichs (1985).

Let us examine Dowty's (pp. 332-336) analysis of the two temporal adverbials in detail:

(2.7) (a) *for* translates into:[3]
$\lambda P_t \lambda P \lambda x[P_t\{n\} \wedge \bigwedge t[t \subseteq n \rightarrow \mathrm{AT}(t, P\{x\})]]$

(b) *x P*'s *for* P_t iff the current interval n satisfies P_t and for all subintervals t of n, $P\{x\}$ is true.

(2.8) (a) *in* translates into:
$\lambda P_t \lambda P \lambda x[P_t\{n\} \wedge \bigvee t_1[t_1 \subseteq n \wedge \mathrm{AT}(t_1, P\{x\}) \wedge$
$\bigwedge t_2[[t_2 \subseteq n \wedge \mathrm{AT}(t_2, P\{x\})] \rightarrow t_2 = t_1]]]$

(b) *x P*'s *in* P_t iff the current interval n satisfies P_t and there exists a unique subinterval t_1 of n such that $P\{x\}$ is true.

These analyses yield the following reduced translations for the given example sentences:

(2.9) (a) John slept for an hour.

(b) $\bigvee t_1[\mathrm{PAST}(t_1) \wedge \text{an-hour}'(t_1) \wedge \bigwedge t_2[t_2 \subseteq t_1 \rightarrow \mathrm{AT}(t_2, \text{sleep}'(j))]]$

(c) There is some past time interval t_1 of an hour's length such that for all subintervals t_2 of t_1, John sleeps is true.

(2.10) (a) John awakened in an hour.

(b) $\bigvee t_1[\mathrm{PAST}(t_1) \wedge \text{an-hour}'(t_1) \wedge$
$\bigvee t_2[t_2 \subseteq t_1 \wedge \mathrm{AT}(t_2, [\text{BECOME awake}'(j)]) \wedge$
$\bigwedge t_3[[t_3 \subseteq t_1 \wedge \mathrm{AT}(t_3, [\text{BECOME awake}'(j)]) \rightarrow t_2 = t_1]]]]$

(c) There is some past time interval t_1 of an hour's length such that there is a unique subinterval t_2 of t_1 in which John becomes awake is true.

Given Dowty's analysis of sleep$'$ and BECOME, these examples do not work the other way around: since sleep$'$ is a homogeneous predicate, it cannot be true at a unique interval; conversely, since BECOME ϕ is true only for the smallest interval during which ϕ switches from false to true, it cannot be true for all subintervals.

These examples illustrate how Dowty's quantificational analysis of *for*-adverbials makes the right predictions in many cases. However, Dowty himself recognizes that there are problems with his analysis of *for*-adverbials. First, there is the minimal parts problem: (2.7a) should not make reference to literally *all* subintervals, but rather those large enough for the activity in question. Second, there is the non-contiguous intervals problem: *John served on that committee for four years* can be true even if John served non-consecutive terms totaling four years in duration. These observations lead Dowty to the following conclusion:

[3]Here P and P_t are property variables, i.e. variables typed for curried functions from indices to individuals to truth values. By the brace convention, $\alpha\{\beta\}$ stands for $\check{}\alpha(\beta)$, i.e. the extension of α at the current index applied to β.

> Perhaps the best view of *for* α is that it asserts that something is the case at each one of some set S of possibly non-contiguous intervals of time, the total duration of which is α though the exact choice of members of S is left to contextual interpretation.

I concur with Dowty's assessment that his analysis could be improved by adding an explicit notion of duration in order to solve the problem of non-contiguous intervals. However, in order to address the problem of contextually relevant minimal parts, I will ultimately argue that his quantificational analysis should be dropped altogether, leaving *for*-adverbials the task of simply specifying the (total) duration of a given situation.

2.1.2 Hinrichs (1985)

Hinrichs (1985) extends Dowty's treatment to better explain the problem of aspectual composition. Unlike Dowty, Hinrichs bases his treatment upon a modified version of Carlson's (1977a) ontology of **kinds**, **objects** and **stages**. The changes are twofold: first, Hinrichs proposes an analogous triple of **event types**, **individual events**, and **event stages**; second, he identifies stages with real-world locations and endows them with the structure of a **complete join-semilattice**—this enables him to address the issue of homogeneous reference in the spirit of Link (1983) and Bach (1986).

Hinrichs (p. 65) observes that at first glance "Carlson's three-tiered ontology might strike the reader as either extremely baroque, or completely counterintuitive, or possibly as being both of these things." He then goes on to review the impressive range of evidence that Carlson provides for his ontology. Since there is not space here to review this evidence, it is worth emphasizing that despite any perceived ungainliness the ontology should not be dismissed out of hand.[4]

Let us examine how Hinrichs' assuming stages to form a lattice of spatio-temporal locations enables him to significantly improve upon Dowty's analysis of locative modifiers. Consider (2.11):[5]

(2.11) (a) Fangs slithered to the rock.
(b) Fangs slithered toward the rock.
(c) Fangs slithered.
(d) Fangs was at the rock.

Hinrichs observes that while Dowty's treatment does account for the entailments from (2.11a) to (2.11c) and (2.11d), it fails to account for the entailment to (2.11b). In contrast, by making use of the notion of stages as spatio-temporal locations, Hinrichs is able to account for all three.

Dowty's (1979, p. 214) translation of *to* appears below:

[4]Cf. Schubert and Pelletier (1987) and also Krifka et al. (1992) for an excellent critical discussion of Carlson's ontology.

[5]Presumably, Fangs is the name of a snake.

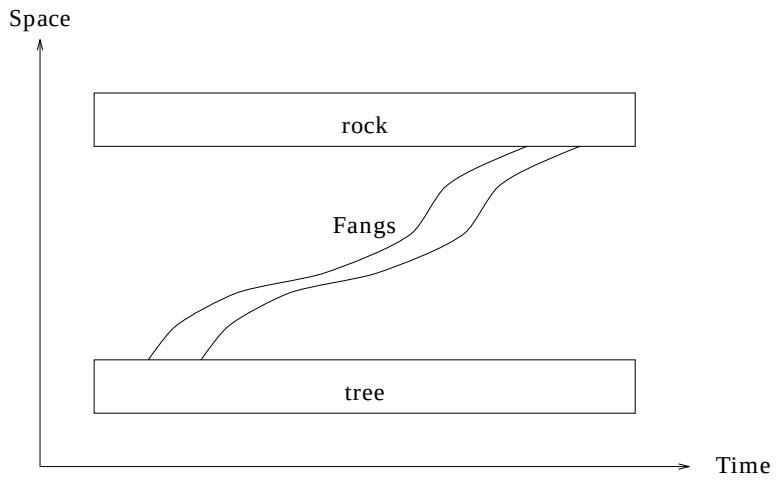


Figure 2.1: Fangs slithering from the tree to the rock (Hinrichs, 1985, p. 208). The *rock* and *tree* stages are depicted as spatio-temporal boxes, indicating that they are stationary. The path Fangs takes from one to the other makes up a "sausage" of space-time locations.

(2.12) *to* translates into:[6]
$\lambda\mathcal{P}\lambda P\lambda x\mathcal{P}\{\dot{y}[P\{x\}$ AND $\bigvee z[$BECOME $\neg$be-at$'(x,z)]$ AND BECOME be-at$'(x,y)]\}$

This translation conjoins the unmodified verb phrase $P\{x\}$ with two clauses which together describe a change from an unspecified location z to the location y (the variable bound by the quantificational object $\mathcal{P}$ of *to*). As such, it does verify the entailments to (2.11c) and to (2.11d), as suggested above; however, since nothing is required of the spatial properties of the transition, the entailment to (2.11b) does not go through.

Hinrichs illustrates his notion of stage as spatio-temporal location using the diagram reproduced in Figure 2.1. To formalize this notion he proceeds as follows. First, he (p. 207) requires motion verbs to identify the event and object stages using the following meaning postulate schema:[7]

(2.13) $\forall x^{s}, e^{s}[\delta^{+}(x^{s}, e^{s}) \rightarrow x^{s} = e^{s}]$, where δ translates *slither*, *walk*, etc.

In this way he (pp. 216-220) accounts for the entailment to (2.11d). Next he (pp. 210, 223) defines the above locative prepositions in terms of a relation

[6] Here $\mathcal{P}$ is a quantifier variable, i.e. a variable typed for a property of a property of individuals.

[7] The superscript 's' is used here to indicate that x and e are stage-level variables.

PATH holding between a spatio-temporal location (the path), its origin and its destination:[8]

(2.14) (a) *to* translates as[9]
$\lambda P \lambda \mathrm{P} \lambda l_1 \lambda x^{\mathrm{i}} P[\lambda y^{\mathrm{i}} \exists l_2[\mathrm{R}(l_2, y^{\mathrm{i}}) \,\&\, \mathrm{PATH}(l_1, l_{\mathrm{r}}, l_2) \,\&\, \mathrm{P}(x^{\mathrm{i}})(l_1)]]$

(b) *to* adds to the VP that l_1 is a stage which forms a path with implicit origin[10] l_{r} and destination l_2, where l_2 is a stage realizing (R) the individual y.

(2.15) (a) *toward* translates as
$\lambda P \lambda \mathrm{P} \lambda e^{\mathrm{s}} \lambda x^{\mathrm{i}} P(\lambda y^{\mathrm{i}} \exists l[\mathrm{R}(l, y^{\mathrm{i}}) \,\&\, \exists l'[\mathrm{PATH}(l', l_{\mathrm{r}}, l) \,\&$
$e^{\mathrm{s}} \leq l' \,\&\, l_{\mathrm{r}} < e^{\mathrm{s}} \,\&\, \mathrm{P}(x^{\mathrm{i}})(e^{\mathrm{s}})]])$

(b) *toward* adds to the VP that e is a stage which is an initial part of ($\leq$) a path l' with implicit origin l_{r} and destination l, where l is a stage realizing the individual y.

Since (2.15a) requires the stage e to be an initial part of a stage l' satisfying the *to* translation, i.e. one having destination l realizing y, the entailment to (2.11b) is explained.

It should be evident that (2.14) and (2.15) differentiate *to* and *toward* with respect to the property of cumulative reference: whereas *to* requires a complete path from origin to destination, *toward* only requires an initial part of such a path; as such, *toward* but not *to* turns out cumulative.[11]

We now turn to how this difference enables Hinrichs to make the correct predictions on the temporal adverbial test. Hinrichs (pp. 234, 241) modifies Dowty's analysis as follows:

(2.16) (a) *for* translates as
$\lambda P_t \lambda S \lambda e_1^{\mathrm{s}} \lambda x^{\mathrm{i}} P_t(\lambda l[l \leq e_1^{\mathrm{s}} \,\&\, S(x^{\mathrm{i}})(e_1^{\mathrm{s}}) \,\&$
$\forall l'[l' < l \rightarrow \exists e_2^{\mathrm{s}}[e_2^{\mathrm{s}} < e_1^{\mathrm{s}} \,\&\, l' \leq e_2^{\mathrm{s}} \,\&\, S(x^{\mathrm{i}})(e_2^{\mathrm{s}})]]])$

(b) x S's *for* P_t iff there is a stage e_1 where x S's which has a subpart l satisfying both the temporal description in P_t and the condition that all proper subparts l' of l must be contained in a proper subpart e_2 of e_1 which is also a stage where x S's.

---

[8] Hinrichs notes several natural restrictions on PATH on pp. 211-213.

[9] Note that Hinrichs' notational conventions differ slightly from Dowty's: here P is a quantifier variable, and P is property variable (the brace convention is not needed since Hinrichs' logic is extensional). The superscript 'i' is used here to indicate that x and y are individual-level variables; by convention, the variable l and its subscripted variants are always stage-level variables.

[10] Hinrichs states (p. 214) that the point of reference l_{r} is supposed the same indexical parameter that contains the Reichenbachian reference point for a given tense; this is possible within his framework since locations are spatio-temporal entities, rather than purely spatial ones.

[11] It is worth noting that the implicit origin l_{r} ensures the heterogeneous reference of *to* y; without it, all final subpaths of a *to* y path would also be *to* y paths.

(2.17) (a) *in* translates as
$\lambda P_t \lambda S \lambda e_1^{\mathrm{s}} \lambda x^{\mathrm{i}} P_t(\lambda l[e_1^{\mathrm{s}} \leq l \ \& \ S(x^{\mathrm{i}})(e_1^{\mathrm{s}}) \ \&$
$\forall e_2^{\mathrm{s}}[e_2^{\mathrm{s}} \leq e_1^{\mathrm{s}} \ \& \ S(x^{\mathrm{i}})(e_2^{\mathrm{s}}) \ \rightarrow \ e_1^{\mathrm{s}} = e_2^{\mathrm{s}}]])$

(b) x S's *in* P_t iff there is a unique stage e_1 where x S's which is a subpart of a stage l satisfying the temporal description in P_t.

Using (2.16a) and (2.17a) Hinrichs (pp. 242-246) explains the aspectual difference between (2.11a) and (2.11)b as follows:[12]

(2.18) (a) Fangs slithered toward the rock for an hour.

(b) There has to be an individual event e^{i} realized by an event stage e_1^{s} such that e_1^{s} is some initial segment of some path l_2 between some implicit point of origin l_{r} and the location of some unique rock. Moreover, e_1^{s} has to be at least one hour l_1 in its temporal dimension, and all spatial-temporal parts of that hour have to be covered by proper substages e_2^{s} of e_1^{s} such that e_2^{s} are event stages of Fangs' slithering toward some unique rock.

(2.19) (a) Fangs slithered to the rock in an hour.

(b) There has to be an individual event e^{i} realized by an event stage e_1^{s} such that e_1^{s} is at most one hour long ($e_1^{\mathrm{s}} \leq l_1$). Moreover e_1^{s} has to be the unique event stage which constitutes a path between some implicit point of origin l_{r} and the location of some unique rock, i.e. there cannot be any substages e_2^{s} of slithering by some Fangs stage such that e_2^{s} would constitute a path between the same point of origin and some location realizing the same rock.

From (2.19) it should be evident that Hinrich's analysis of *in*-adverbials is essentially the same as Dowty's. Example (2.18), in contrast, reveals that Hinrich's analysis of *for*-adverbials is considerably more complex. Looking back at (2.16), we see that while he maintains Dowty's quantificational analysis, insofar as he quantifies over all substages l' of the temporally-measured stage l, he only requires these substages to be *covered* by some stage e_2 satisfying the description $S(x)$. This move enables him to cleverly sidestep the minimal parts problem. However, in order to still account for the temporal adverbial test, he must ensure that there is no unique stage satisfying the description $S(x)$; this he does by requiring the substages l' to be covered by stages e_2 which are *proper* substages of a stage e_1 satisfying $S(x)$. Finally, he (p. 237) solves the problem of non-contiguous intervals by simply not requiring the denotations of expressions such as *hour* or *year* to contain contiguous stages.

Next we turn to Hinrich's analysis of mass terms and bare plurals. As mentioned previously, Hinrichs generalizes Carlson's ontology of kinds, objects and

[12] These explanations are reproduced almost verbatim, with the exception of replacing Fang's slithering for John's walking. For space reasons I will omit the rather lengthy logical forms.

stages by adding an analogous triple of event types, individual events, and event stages, where the stages are endowed with a semi-lattice structure. Adding this additional structure enables Hinrichs to account for the inherent cumulativity of activities such as *John eating cake* as illustrated below:

(2.20)

$$\frac{\begin{array}{c}\mathrm{eat}^{+}(c_1^{\mathrm{s}})(j_1^{\mathrm{s}})(e_1^{\mathrm{s}}) \\ \mathrm{eat}^{+}(c_2^{\mathrm{s}})(j_2^{\mathrm{s}})(e_2^{\mathrm{s}})\end{array}}{\mathrm{eat}^{+}(c_1^{\mathrm{s}}+c_2^{\mathrm{s}})(j_1^{\mathrm{s}}+j_2^{\mathrm{s}})(e_1^{\mathrm{s}}+e_2^{\mathrm{s}})}$$

Schema (2.20) indicates that if j_i^{s} are stages realizing the individual John, c_i^{s} are stages realizing the kind cake, and e_i^{s} are stages which are eatings of c_i by j_i, then the sum of these eating stages $\sum_i e_i$ must be an eating of $\sum_i c_i$ by $\sum_i j_i$.[13]

Let us now examine in some detail how Hinrichs uses this idea to account for the effects of mass terms and bare plurals on the temporal adverbial test. Consider the following sentences:

(2.21) (a) John ate cake for an hour.

(b) John ate a cake in an hour.

(c) John ate cakes for an hour.

We begin with (2.21a). Following Carlson, Hinrichs assumes mass terms name the unique kind such that all its realizations have the property of the corresponding mass noun. This is exemplified below in the case of *gold*:

(2.22) *gold* names $\iota x^{\mathrm{k}}[\forall z^{\circ}[\mathrm{R}'(z,x) \leftrightarrow \mathrm{gold}'(z)]]$

Letting c name the unique kind for the mass noun *cake*, Hinrichs (p. 278) derives the following logical form for (2.21a):

(2.23) $\exists e_1^{\mathrm{s}}, e^{\mathrm{i}}[\mathrm{R}(e_1^{\mathrm{s}}, e^{\mathrm{i}}) \,\&\, \mathrm{PAST}(e_1^{\mathrm{s}}) \,\&\, \exists l[\mathrm{hour}'(l) \,\&\, l \leq e_1^{\mathrm{s}} \,\&$
$\exists x_1^{\mathrm{s}}, y_1^{\mathrm{s}}[\mathrm{R}(x_1^{\mathrm{s}}, j) \,\&\, \mathrm{R}(y_1^{\mathrm{s}}, c) \,\&\, \mathrm{eat}^{+}(y_1^{\mathrm{s}})(x_1^{\mathrm{s}})(e_1^{\mathrm{s}})] \,\&$
$\forall l'[l' < l \;\rightarrow\; \exists e_2^{\mathrm{s}}[e_2^{\mathrm{s}} < e_1^{\mathrm{s}} \,\&\, l' \leq e_2^{\mathrm{s}} \,\&$
$\exists x_2^{\mathrm{s}}, y_2^{\mathrm{s}}[\mathrm{R}(x_2^{\mathrm{s}}, j) \,\&\, \mathrm{R}(y_2^{\mathrm{s}}, c) \,\&\, \mathrm{eat}^{+}(y_2^{\mathrm{s}})(x_2^{\mathrm{s}})(e_2^{\mathrm{s}})]]]]]$

Logical form (2.23) requires that each subpart l' of the hour l be covered by a stage e_2 which is an eating stage of y_2 realizing the kind cake (and that e_2 be non-unique). Because kinds can realize multiple quantities, this is unproblematic; moreover, since such eating stages are not unique, the incompatibility of *in*-adverbials here is also explained.

[13] Note that the consequent here follows from a meaning postulate ensuring closure under cumulative reference for predicates such as eat^{+}.

Example (2.21b) is relatively straightforward. Here Hinrichs (p. 279) simply provides a meaning postulate for eat^+ and its ilk that requires eating events involving individual objects (rather than kinds) to have unique stage realizations; this postulate then suffices to predict the appearance of *in*-adverbials (but not *for*-adverbials) in such sentences.

Finally we turn to (2.21c). Extending Carlson, Hinrichs assumes bare plurals name the unique kind such that all its realizations are plural individuals in the denotation of the plural predicate formed with Link's (1983) **circle-star operator**. This operator applies to a predicate ranging over atomic individuals to form a predicate denoting all the non-atomic sums of these individuals. For example, if cat' denotes all individual cats, then $^{\oplus}\ \text{cat}'$ denotes all those non-atomic individuals consisting of at least two cats. Bare plurals such as *cats* thus name the kind that is realized by such objects, as shown below:

(2.24) *cats* names $\iota x^{\text{k}}[\forall z^{\circ}[\text{R}'(z,x) \leftrightarrow {}^{\oplus}\ \text{cat}'(z)]]$

If we now let c name the unique kind for the bare plural *cakes*, then it turns out that the logical form for (2.21c) is just (2.23), which is again unproblematic because c can realize multiple plural individuals.

Hinrichs' analysis is as clever as it is complex. Nevertheless, it is worth considering carefully whether all of its complexities are actually necessary. In particular, it is a striking feature of his account that the event types and individual events, which are supposed to be analogous to Carlon's kinds and objects, play essentially no role in his treatment of aspectual composition. In large part, Hinrichs' under-utilization of these entities stems from his decision to follow Dowty in proposing a quantificational analysis of *for*-adverbials, rather than letting them be sensitive to sortal distinctions in the ontology. As was hinted previously, in order to arrive at a simpler and more explanatory treatment, I will ultimately argue against the quantificational analysis and in favor of one relying solely on such sortal distinctions.

2.1.3 Krifka (1989; 1992)

Krifka (1989; 1992) presents an analysis of the problem of aspectual composition which is reminiscent of Hinrichs' in its lattice-theoretic formulation. However, unlike Hinrichs, Krifka employs neither Carlson's ontology nor Dowty's quantificational analysis. Instead Krifka relies upon the shared reference properties of nominal arguments and complex verbal expressions, which he suggests is more in line with the insights of earlier feature-based syntactic approaches such as (Verkuyl, 1972).

Again following Link (1983), Krifka bases his analysis upon complete join semi-lattices for the sorts object, event and time interval. This enables him (Krifka, 1989, p. 78) to define the following reference properties of arbitrary

predicates:[14]

(2.25) (a) $\forall P[\mathrm{DIV}(P) \leftrightarrow \forall x \forall y[P(x) \wedge y \sqsubseteq x \rightarrow P(y)]]$

(b) Predicate P is **divisive** iff whenever it applies to an entity x it also applies to all subparts y of x.

(2.26) (a) $\forall P[\mathrm{CUM}(P) \leftrightarrow \forall x \forall y[P(x) \wedge P(y) \rightarrow P(x \sqcup y)]]$

(b) Predicate P is **cumulative** iff whenever it applies to an entity x and an entity y it also applies to their sum $x \sqcup y$.

(2.27) (a) $\forall P[\mathrm{QUA}(P) \leftrightarrow \forall x \forall y[P(x) \wedge P(y) \rightarrow \neg[y \sqsubset x]]]$

(b) Predicate P is **quantized** iff whenever it applies to an entity x it does not apply to any proper subpart y of x.

Ignoring complications arising when predicates have empty or singular extensions, it is easy enough to check that a predicate cannot be quantized if it is cumulative or divisive, and vice-versa.

Krifka (pp. 81-88) discusses how nominal measure phrases such as *five ounces of* can change a cumulative predicate to a quantized one:

(2.28) (a) *gold* translates as the cumulative predicate gold'

(b) *five ounces of* translates as $\lambda P \lambda x[P(x) \wedge \mathrm{oz}'(x) = 5]$

(c) *five ounces of gold* translates as $\lambda x[\mathrm{gold}'(x) \wedge \mathrm{oz}'(x) = 5]$

Krifka assumes that measure functions such as oz' are **extensive** (pp. 78-81), i.e. compatible with the part-of lattice; in particular, he requires them to preserve sums of non-overlapping entities, and to assign positive measures to all entities in its domain. As a consequence of these assumptions, measure functions cannot assign the same measure to an entity and one of its proper subparts. For this reason, the translation of *five ounces of gold* in (2.28c) turns out quantized.

Observing that nominal measure phrases resist iteration, as illustrated by examples such as * *five ounces of seven cubic centimeters of gold*, Krifka suggests that this change in reference properties should be made a well-formedness condition on the modifier. To do so, he introduces the relation of 'quantizing modification' QMOD:

(2.29) $\forall P \forall Q[\mathrm{QMOD}(P, Q) \leftrightarrow \neg\mathrm{QUA}(P) \wedge \mathrm{QUA}(Q(P))]$

This same approach is then carried over to adverbial measure constructions (pp. 97-98):

(2.30) *for an hour* translates as
$\lambda P \lambda e[P(e) \wedge \mathrm{hour}'(e) = 1 \ / \ \mathrm{QMOD}(P, \lambda P \lambda e[P(e) \wedge \mathrm{hour}'(e) = 1])]$

[14] For readability I have left the sorts implicit.

Note that the slash is used here to suggest that QMOD is supposed to add a well-formedness condition; for reasons of simplicity, however, it is technically treated as part of the assertion.

Krifka's analysis of *in*-adverbials is superficially quite similar. As shown below, he requires the **temporal trace** $\tau(e)$ (p. 97) to lie within a one-hour interval:[15]

(2.31) *in an hour* translates as
$\lambda P \lambda e[P(e) \wedge \exists t[\text{hour}'(t) = 1 \wedge \tau(e) \sqsubseteq t]]$

In order to understand the differing behavior of *in*-adverbials, Krifka (pp. 98-100) suggests we should look to pragmatics. First he notes that under his analysis *for*-adverbials are downward entailing; thus, for example, if *Ann read for three hours* is true, then *Ann read for two hours* must be true as well. Of course, these sentences are not equally informative; since the second sentence is less informative than the first, it is dispreferred by Grice's Maxim of Quantity. Turning to *in*-adverbials, Krifka observes that the situation is reversed; since *in*-adverbials are upward entailing, we have, for example, *Ann drank a bottle of wine in one hour; in fact, she did it in 53 minutes.* In order to be maximally informative then, the duration given by the *in*-adverbial should be as small as possible. From this Krifka concludes that *in*-adverbials must modify quantized event predicates,[16] as otherwise there would be smaller and smaller events falling under the event predicate, and thus no such smallest duration.

Given his analysis of temporal adverbials, the problem of aspectual composition for Krifka reduces to the problem of explaining the transfer of reference properties from nominal arguments to complex verbal expressions. This he does in a manner reminiscent of Hinrichs, as can be seen by considering Figure 2.2, which illustrates the idea. Technically, he adopts a neo-Davidsonian representation[17] and formalizes the transfer properties of thematic relations. To characterize these properties, he first defines the following predicates:

(2.32) **Summativity**:
$\forall R[\text{SUM}(R) \leftrightarrow \forall e \forall e' \forall x \forall x'[R(e,x) \wedge R(e',x') \rightarrow R(e \sqcup e', x \sqcup x')]]$

(2.33) **Uniqueness of Objects**:
$\forall R[\text{UNI-O}(R) \leftrightarrow \forall e \forall x \forall x'[R(e,x) \wedge R(e,x') \rightarrow x = x']]$

[15] Krifka additionally requires the interval to be convex. However, it seems to me that as in the case of *for*-adverbials, convexity is only the normal case.

[16] Krifka suggests that this requirement should be relaxed to atomic reference, rather than quantized reference, i.e. that the event predicate should have some smallest event in its extension. Krifka justifies this position by claiming that in the contex of some unusual contest, one might felicitously refer to such an atomic event in *Ann drank wine in 0.43 seconds.* To me, however, such a sentence can only say how much time it took her to *start* drinking wine.

[17] Note that Krifka does so for convenience only. For a critical discussion of the possible empirical content of the neo-Davidsonian view, cf. Dowty (1989) and Dowty (1991).

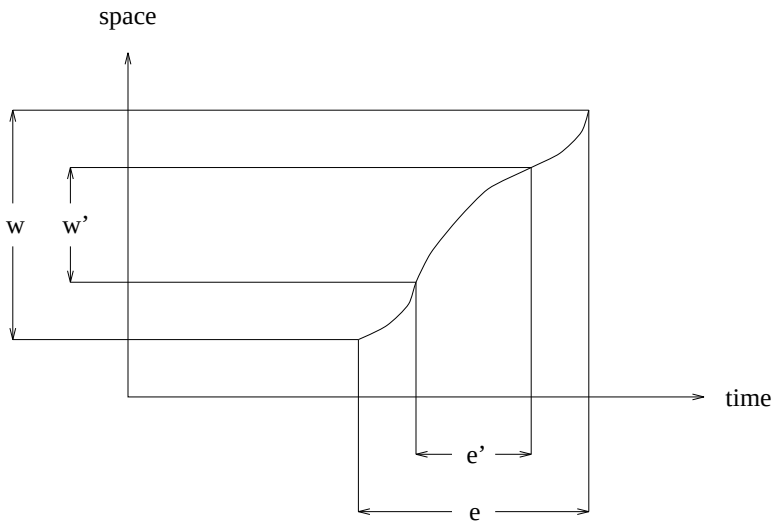


Figure 2.2: An (idealized) event e of gradually drinking an amount w of wine, which is a glass of wine (Krifka, 1989, p. 91). The event e' is a subevent of drinking an amount w' of wine. Since *wine* is cumulative, both w' and w satisfy *wine*, and thus both e' and e satisfy *drink wine*; in contrast, since *a glass of wine* is quantized, only w satisfies *a glass of wine*, and thus only e satifies *drink a glass of wine*. This illustrates why the former verbal expression turns out cumulative and the latter quantized.

(2.34) **Uniqueness of Events**:
$\forall R[\text{UNI-E}(R) \leftrightarrow \forall e \forall e' \forall x[R(e,x) \wedge R(e',x) \rightarrow e = e']]$

(2.35) **Mapping to Objects**:
$\forall R[\text{MAP-O}(R) \leftrightarrow \forall e \forall e' \forall x[R(e,x) \wedge e' \sqsubseteq e \rightarrow \exists x'[x' \sqsubseteq x \wedge R(e',x')]]]$

(2.36) **Mapping to Events**:
$\forall R[\text{MAP-E}(R) \leftrightarrow \forall e \forall x \forall x'[R(e,x) \wedge x' \sqsubseteq x \rightarrow \exists e'[e' \sqsubseteq e \wedge R(e',x')]]]$

(2.37) **Graduality**:
$\forall R[\text{GRAD}(R) \leftrightarrow \text{UNI-O}(R) \wedge \text{MAP-O}(R) \wedge \text{MAP-E}(R)]$

Krifka (pp. 92-96) then considers the conditions necessary to prove transfer of reference for the schematic complex verbal predicate $\lambda e \exists x[\alpha(e) \wedge \delta(x) \wedge \theta(e,x)]$, where α is the one-place verbal predicate, δ the nominal one, and θ a specific thematic relation. To summarize, he finds that

1. in order to capture the transfer of cumulativity, e.g. from *wine* to *drink wine*, α should be cumulative and θ summative; and

2. in order to capture the transfer of quantization, e.g. from *a glass of wine* to *drink a glass of wine*, θ should additionally satisfy graduality, i.e. uniqueness of objects, mapping to objects and mapping to events.

Note that uniqueness of events is simply an additional requirement which may be used to rule out iteration in the case of effected or consumed patients.

From the preceding discussion, it should be apparent that sortal distinctions play a very small part in Krikfa's analysis: rather than relying on shared sortal distinctions between events and processes on the one hand and objects and substances on the other, as in the approach to be developed here, Krifka instead relies on the shared reference properties of complex verbal expressions and their nominal arguments. Interestingly, Krifka does briefly sketch (p. 87) how Link and Bach's sortal distinction between individuals and quantities of matter might be used in resolving a problem which arises in the case of count nouns that fail to exhibit quantized reference. However, he does not consider whether this same sortal distinction might be useful in analyzing complex verbal expressions; moreover, he glosses over this distinction in the rest of (Krifka, 1989), and does not mention it at all in (Krifka, 1992). This point is taken up again in section 2.4, where I will ultimately argue that shared reference properties do not suffice to account for the linguistic data.

2.1.4 Verkuyl (1989)

Verkuyl (1989) presents a model-theoretic reformulation of the appoach originally set forth in his thesis work (Verkuyl, 1972).[18] Unlike the preceding

[18] Cf. also Verkuyl (1993), which became available too late to be reviewed here.

works, Verkuyl does not rely on shared reference properties, either directly, as in (Krifka, 1989), or indirectly, as in (Dowty, 1979) and (Hinrichs, 1985). Instead, Verkuyl relies on the interaction of two model-theoretically interpreted features [±ADD-TO] and [±SQA], where SQA stands for **Specified Quantity of** A. These two features combine to determine whether a sentence has **durative** or **terminative** aspect[19] according to the **plus principle**, which states that terminative aspect is only possible when all the features involved have a plus-value.[20]

The feature [±ADD-TO] (pp. 84-90) distinguishes stative and non-stative lexical verbs. In the case of [+ADD-TO] verbs, this feature requires there to be a 'counting' function s from time intervals to time intervals which returns successive including intervals; with this feature, Verkuyl intends to capture the intuitive notion of processes inherently involving successive phases in time. When applied to a [+SQA] direct object NP, the function s composes with a participancy function p_v mapping partitioning subparts to intervals to produce a terminative function t, i.e. one which exhausts the domain of partitions.[21] Applying a [+ADD-TO] verb to a [-SQA] direct object NP, in contrast, does not yield such a terminative function. This should become clearer momentarily, when we consider Verkuyl's worked-out example.

The feature [±SQA] (pp. 81-83) distinguishes those NPs which denote a 'Specified Quantity of A' from those that do not, where 'specified quantity' pertains to having an (in principle) bounded non-zero cardinality. Verkuyl defines this notion as shown in (2.38) below (the roles of the sets $A^{\#}$, A, B, and E should also become clearer presently):[22]

(2.38) (a) **Specified Quantity of A**:
An NP of the form Det N, where $[\![N]\!] = A$ and where $[\![\text{Det}]\!]$ relates a set B to A in a specific model M_i denotes a specified quantity of A in E, $A^{\#}$, $(A^{\#} \subseteq A \subseteq E)$ iff

(i) E is bounded
(ii) $A^{\#} = A \cap B$
(iii) $|\, A^{\#} \,| > 0$

(b) **Unspecified Quantity of A**:
An NP of the form Det N denotes an unspecified quantity of A

[19] For Verkuyl, sentences with durative aspect are those that select for *for*-adverbials, and sentences with terminative aspect are those that select for *in*-adverbials.

[20] Presumably, what motivates Verkuyl to call this particular method of feature combination a principle is that it is supposed to follow from the interaction of the model-theoretic interpretations of the features [±ADD-TO] and [±SQA].

[21] Verkuyl (fn. 5, p. 92) notes that Vendler's *push the cart* and others like *stroke a cat* require more to be said. More generally, it seems that Verkuyl is restricting his attention to 'gradually affected' objects here.

[22] Note that Verkuyl does not specify whether the subclauses in (2.38) should be interpreted conjunctively or disjunctively; presumably, those in (a) should be interpreted conjunctively, and those in (b) disjunctively.

(i) if $A \cap B = \emptyset$

(ii) if there is no number given by the definition of the quantifier by which the cardinality of the intersection is bounded.

Let us now examine Verkuyl's analysis of how these features can be used to explain the aspectual contrast between the VPs *lift four tables* and *lift tables*.[23] In the generalized quantifier tradition (Barwise and Cooper, 1981), Verkuyl takes determiners to denote relations between sets. In *lift four tables* then, the noun *table* denotes a set A of tables, which is a subset of the universe of discourse E; the determiner *four* denotes a relation between this set and the set B denoted by the verb *lift*. The set $A^{\#}$ is defined to be the set $A \cap B$ of tables involved in the lift-predication. For the moment, let us assume *four* is interpreted as meaning exactly four, i.e. as the quantifier $\{X \subseteq E : A \subseteq X \ \& \ |A \cap X| = 4\}$; in this case, $A^{\#}$ must pertain to a set of exactly four tables.

Simplifying Verkuyl's treatment slightly,[24] the participancy function p_v is defined to be the function which maps each element of $A^{\#}$ to the interval in which it is lifted. Because the domain of $A^{\#}$ is finite, the composition of the counting function s with the participancy function p_v produces a function t which 'exhausts the domain', so s 'comes to a stop', in Verkuyl's words.[25] It is for this reason that the [+SQA] NP *four tables* yields a terminative VP when combined with the verb *lift*.

Turning now to the [-SQA] NP *tables*, Verkuyl suggests that bare plural NPs such as this one should be defined as $\{X \subseteq E^* : A \subseteq X \ \& \ |A \cap X| = \text{undetermined}\}$, the idea being that bare plurals contain in their definition the information that the cardinality of E is not bounded, as indicated by E^*, and hence that the cardinality of the intersection $A \cap X$ cannot be determined.[26] From this it is supposed to follow that the cardinality of $A^{\#}$ cannot be determined, and thus that the composition of the counting function s and the participancy function p_v does not yield a terminative function t in this case.

Unfortunately, Verkuyl's formalization of [±SQA] is somewhat vague exactly where it needs to be precise. One problem is that the notion of bounded cardinality in subclause (2.38bii) does not adequately explain how the two values of the feature [±SQA] could correspond to taking different perspectives upon a state of affairs. For example, in the case of bare plurals, it seems clear enough that they may be felicitously employed even when the speaker has in mind only finite domains. Conversely, it is not at all clear why [+SQA] NPs such as *at least four tables* should presuppose a finite universe E, as is required by subclause

[23] For the sake of brevity, we will not examine how this distinction is carried over to the sentence level.

[24] Verkuyl considers partitions of $A^{\#}$, rather than just the elements of this set.

[25] The details of exactly how this is supposed to work out (pp. 82-83) are rather obscure. For present purposes, it suffices to point out Verkuyl's reliance upon the finite domain of $A^{\#}$.

[26] One is left to wonder what happens if A is finite; presumably, this case would be covered by the caveat in (2.38bii).

(2.38ai). This latter problem is at least partially acknowledged in the following comment (p. 83):

> So even though we know from arithmetic that *at least four* opens up an infinite domain of numerical values, the use of language can close off this domain such that *at least n* is going to mean 'some value $\geq n$ but we do not know which one'.

Although Verkuyl seems somewhat aware of these problems with his notion of specified quantity, he does not propose solutions to them; as a more satisfactory alternative, I will ultimately propose replacing this distinction with a sortal one. It is worth mentioning nevertheless that, despite this difference in setup, Verkuyl's approach has much in the common with the present one, insofar as it does not rely on reference properties to account for the problem of aspectual composition.

2.1.5 Jackendoff (1991)

Unlike the previous authors, Jackendoff (1991) does not present a model-theoretic analysis of the problem of aspectual composition, preferring instead to remain at a more informal level. This makes his approach rather difficult to assess, as we shall see below. Nevertheless, despite his somewhat underformalized approach, Jackendoff presents a number of ideas worth pursuing.

Zwarts and Verkuyl (1994) discuss a number of ways one might formalize Jackendoff's rather idiosyncratic notation in fairly standard model-theoretic terms.[27] At the simplest level, they suggest, Jackendoff's **conceptual structures** can be considered to consist of **features** and **concepts**, where n-ary features are taken to be functions from zero or more entities to unary predicates, and concepts are taken to be indefinite terms denoting entities satisfying the given featural conditions. For example, consider the conceptual structure in (2.39b) below, which corresponds to the syntactic structure in (2.39a):

(2.39) (a) [[Bill] [went [into [the house]]]]

(b) $[\mathrm{GO}([\mathrm{BILL}]_x,[\mathrm{INTO}([\mathrm{HOUSE}]_y)]_p)]_e$

According to the conception of Jackendoff's notation suggested above, the conceptual structure in (2.39b) involves four entities, indexed by x, y, p, and e, and four features, GO, BILL, INTO, and HOUSE. The entities should be restricted to the sorts Thing, Path, and Event, respectively, though this is omitted here. As for the features, the zero-ary features BILL and HOUSE are interpreted as conditions on x and y, the unary feature INTO is interpreted as function from y to a condition on p, and the binary feature GO is interpreted as function from x

[27] Cf. also (Verkuyl and Zwarts, 1992).

and p to a condition on e. Under this conception, (2.39b) could be equivalently expressed as shown below:[28]

(2.40) $\exists x.\, \exists y.\, \exists p.\, \exists e.\, [\mathsf{Bill}(x) \wedge \mathsf{house}(y) \wedge \mathsf{into}(y, p) \wedge \mathsf{go}(x, p, e)]$

Here the n-ary features have simply been converted to $(n+1)$-ary predicates, and existential closure has been applied to the variables. In (2.40), it should be easier to see that (2.39a) is taken to assert the existence of an event e in which Bill (x) traverses a path p which is into the house y.

Unfortunately, this conception of Jackendoff's notation does not appear to be suitable for all of the features which Jackendoff employs. For example, consider the conceptual structure $[\mathrm{PL}([\mathrm{DOG}]_x)]_y$, which Jackendoff suggests for the plural NP *dogs*. In this case, it does not make sense to think of the feature PL as mapping some particular dog x to a condition on the collection y;[29] instead, what we want is for y to be a collection all of whose members satisfy the condition DOG. Note, however, that we can make sense of this conceptual structure if we take x to be the type or kind corresponding to the noun *dog*, which might be realized formally in any number of ways.

Because of this imprecision on Jackendoff's part, it is difficult to determine exactly how one should conceive of his approach. This matter becomes particularly problematic when one examines his treatment of aspect, as we shall see presently.

Jackendoff (1991) proposes to replace the ontological categories found in (Jackendoff, 1990) with supercategories and cross-categorial features. Thus, for example, events and states become two different types of situations, distinguished by the feature [± DIR] for 'directed'; this feature together with one for dimensionality likewise distinguishes places and paths as two types of spaces. For present purposes, the two most important features are [±b] for 'bounded' and [±i] for 'internal structure'.[30] These two features cross-classify different types of situations, spaces and material entities. In the latter case, these yield the following categories:

(2.41)

features	mnemonic	example
[+b, −i]	individuals	*a pig*
[+b, +i]	groups	*a committee*
[−b, −i]	substances	*water*
[−b, +i]	aggregates	*buses, cattle*

[28] Zwarts and Verkuyl also discuss a more DRT-like notation for conceptual structures which is like (2.39b) in that it preserves much of the syntactic structure of (2.39a); interestingly, they suggest that this notation points the way to a possible integration of DRT and Jackendoff's conceptual semantics. I will not pursue this matter here, since the issues for which DRT was developed (such as the interpretation of anaphora) are not taken to be central to the present investigation.

[29] At best, this might be appropriate for a collection y of clones of x.

[30] Note that this latter feature, despite its somewhat opaque name, is simply meant to distinguish plural individuals from singular ones.

As correlates of these categories, Jackendoff gives the following examples in the situational domain:

(2.42)

features	example
[+b, −i]	John ran to the store
[+b, +i]	The light flashed until dawn
[−b, −i]	John slept
[−b, +i]	The light flashed continually

As an aside, it is worth mentioning that these parallels would be more exact if Jackendoff had chosen something like 'plurality' (e.g. *three little pigs*) over the term 'group', insofar as a committee is more than the sum of its parts—while a committee can have all its members change or even have no members, a series of flashings seems to be nothing more than that.[31]

While the plurality feature [±i] seems relatively unproblematic, the same cannot be said for the boundedness feature [±b]. Recognizing this, Jackendoff attempts to clarify his notion of boundedness as follows:

> Let me be slightly more specific about what is intended by −b. As suggested in the previous section, a speaker uses a −b constituent to refer to an entity whose boundaries are not in view or not of concern; one can think of the boundaries as outside the current field of view. This does not entail that the entity is absolutely unbounded in space or time; it is just that we can't see the boundaries from the present vantage point.

From this description, one is left to wonder whether the feature [±b] is really meant to classify entities rather than expressions. Certainly his use of the terminology '−b constituent' could lead one to believe that this feature should be conceived of as classifying the reference properties of predicates, as in Krifka's approach.[32] If one does take this feature to classify entities, one is then left to wonder what if anything Jackendoff's notion of boundedness has to do with the standard, topological one. Nevertheless, I will take Jackendoff literally here, and assume that this feature should be interpreted as a sortal distinction. Naturally, this assumption raises a number of important questions, including the one just mentioned; for ease of exposition, however, I will postpone discussion of these until section 2.5 and chapter 4.

To complement the zero-ary sortal features above, Jackendoff proposes several unary features which map between the sorts. Of particular interest is the

[31] Something analogous may be at work in examples like *I rang the bell three times twice.* It seems to me that such examples are felicitous only if there is some reason to group the ringings into threes; for example, if two rings is for Gerd Jan, three for Hanno, etc.

[32] Note that statements such as 'A closed event such as *John ran to the store* is [+b,−i]' do not help to clarify matters either.

feature COMP, for 'composed of', which maps a −b entity to its +b counterpart. In introducing this feature, Jackendoff suggests it can be used to encode one meaning of the N of NP construction, as shown below:

(2.43) (a) a house of wood

(b) $$\left[\begin{array}{l} +\mathrm{b}, -\mathrm{i} \\ \mathrm{HOUSE} \\ \mathrm{COMP}\left(\left[\begin{array}{l} -\mathrm{b}, -\mathrm{i} \\ \mathrm{WOOD} \end{array}\right]_x\right) \end{array}\right]_y$$

(2.44) (a) a stack of bricks

(b) $$\left[\begin{array}{l} +\mathrm{b}, -\mathrm{i} \\ \mathrm{STACK} \\ \mathrm{COMP}\left(\left[\begin{array}{l} -\mathrm{b}, +\mathrm{i} \\ \mathrm{PL}\left(\left[\begin{array}{l} +\mathrm{b}, -\mathrm{i} \\ \mathrm{BRICK} \end{array}\right]_x\right) \end{array}\right]_y\right) \end{array}\right]_z$$

An issue which Jackendoff does not address is whether two distinct entities can be composed of the same substance: in (2.43b), for example, we are left to wonder whether there can be a table z which is composed of the same substance x as is the house y. I will assume that this is indeed possible; once again, however, I will postpone discussion of the consequences of this assumption until section 2.5.[33]

Jackendoff also suggests that the feature COMP can be used in encoding the meaning of measure phrases such as *three inches of* or *for an hour*. This suggestion is tantamount to requiring these expressions to select for unbounded entities, in contrast to restrictive modifiers such as *three inch long* or *in an hour*, which presumably do not have COMP as part of their meaning. It is worth noting that Jackendoff's treatment of these phrases is reminiscent of Krifka's, where nominal and verbal measure phrases are also given a unitary analysis; however, in this case, and in the present account, the selectional restrictions are posited to be sortal constraints.

Turning now to the problem of aspectual composition, the inadequacy of Jackendoff's rather informal approach becomes even more problematic. Although Jackendoff does not directly address the problem of aspectual composition, it seems safe to assume he would endorse some sort of feature passing mechanism along the lines of Verkuyl's plus principle. This is sketched in the following examples:

(2.45) (a) A swan glided along the shore for thirty seconds.

[33] Motivation for this assumption is given in the next section, where Jackendoff's treatment of the imperfective paradox is reviewed.

(b)

[Material, +b]	A swan
[Space, −b]	along the shore
[Situation, −b]	A swan glides along the shore

(2.46) (a) Swans glided past the dock for ten minutes.

(b)

[Material, −b]	Swans
[Space, +b]	past the dock
[Situation, −b]	Swans glide past the dock

(2.47) (a) A swan glided past the dock in thirty seconds.

(b)

[Material, +b]	A swan
[Space, +b]	past the dock
[Situation, +b]	A swan glides past the dock

There are a couple of problems worth mentioning here, both stemming from the absence of any model-theoretic interpretation of the features [±b] and [±i]. First, the feature mechanism would have little explanatory value, insofar as it would not make clear why these features should combine according to the plus principle, rather than in some other way.[34] Second, such an approach could not account for the distributive readings of temporal adverbials, since these specify the durations of situations not appearing explicitly in the conceptual structures. While these readings have been largely ignored in the literature, examples are easy enough to construct. In (2.48), for instance, the first adverbial of each sentence is distributive, insofar as it specifies the duration of each basic situation, i.e. each situation of a swan gliding either past the dock or along the shore; the second adverbial, in contrast, simply specifies the duration of the entire meta-situation:

(2.48) (a) Swans glided past the dock in 30 seconds for 10 minutes.

(b) Twenty swans glided past the dock in 30 seconds in 10 minutes.

(c) Twenty swans glided along the shore for 30 seconds in 10 minutes.

(d) Swans glided along the shore for 30 seconds for 10 minutes.

As we have just seen, Jackendoff's analysis differs from the previous ones insofar as it makes use of a cross-categorial sortal distinction which measure phrases are sensitive to. While the approach seems promising, it suffers from an absence of a model-theoretic basis, an inadequacy that the present account seeks to resolve.

[34] Cf. also (Verkuyl and Zwarts, 1992).

2.2 The Imperfective Paradox

The imperfective paradox is the observation that the entailments from a progressive verb phrase to its non-progressive counterpart differ according to whether the verb phrase is an activity one or an accomplishment one. This is shown in the contrast below:[35]

(2.49) (a) $\models$ Jack was pushing a cart. / Jack pushed a cart.

(b) $\not\models$ Jack was drawing a circle. / Jack drew a circle.

Intuitively, we judge (2.49a) valid because the antecedent entails that a process of Jack pushing a cart went on at least until the time of evaluation, which is enough to make the consequent true;[36] in contrast, we do not judge (2.49b) valid, since the process of Jack drawing a circle entailed to have gone on by the antecedent need only constitute part of an event of Jack drawing the circle—in other words, nothing guarantees that this process ever finishes.

The imperfective paradox is intimately related to aspectual composition, since the differing entailments are not solely determined by the verb. This is shown in the following examples adapted from the previous section:[37]

(2.50) (a) $\models$ Jack was pouring wort into the carboy. / Jack poured wort into the carboy.

(b) $\not\models$ Jack was filling the carboy with wort. / Jack filled the carboy with wort.

(2.51) $\not\models$ Jack was pouring twenty liters of wort into the carboy. / Jack poured twenty liters of wort into the carboy.

(2.52) $\models$ Jack was filling carboys with wort. / Jack filled carboys with wort.

(2.53) (a) $\models$ Jack was running along the river. / Jack ran along the river.

[35] Note that the futurate progressive (as in *Jack was VP-ing the following week*) will be ignored throughout this section.

[36] The reader might object that we should not judge (2.49a) valid, since *Jack pushed a cart* seems to suggest that the activity has been completed; with *Jack was pushing a cart*, on the other hand, the pushing could still be in progress. It appears that this is only an implicature, however, since the suggestion of completion can be cancelled—cf. *Jack was pushing a cart. I know he pushed it past Independence Hall, but I have no idea whether he's still pushing it now.*

[37] Example (2.52) exhibits a peculiar case of the minimal parts problem, insofar as the antecedent but not the consequent seems applicable in a situation where Jack is interrupted not long after starting. As such, this entailment must be subject to the caveat that the time of evaluation in the antecedent cannot be in the middle of the first filling event (at least).

(b) $\not\models$ Jack was running to the museum.
Jack ran to the museum.

Having briefly introduced the problem, we turn now to prior approaches.[38]

2.2.1 Dowty (1979)

Dowty (1979) presents a modal analysis of the progressive relying on **inertia worlds**, which are those worlds which are just like the current world up to the present moment, but after which nothing unexpected or abnormal happens. Inertia worlds allow Dowty (p. 149) to define the truth of [PROG ϕ] in the current world in terms of the truth of ϕ in the inertia worlds (irrespective of the ϕ's truth in the current world) and thus avoid the imperfective paradox:[39]

(2.54) [PROG ϕ] is true at $\langle I, w\rangle$ iff for some interval I' such that $I \subset I'$ and I is not a final subinterval for I', and for all w' such that $w' \varepsilon\, Inr(\langle I, w\rangle)$, ϕ is true at $\langle I', w'\rangle$.

Dowty relies on the homogeneous reference of activity predicates to account for the entailments above. This may be sketched as follows. Suppose *Jack be pushing a cart* is true in w at I. By the semantics of the progressive, *Jack push a cart* is true in every inertia world w' for $\langle I, w\rangle$ at some I' surrounding I. Since *Jack push a cart* is an activity, it is also true at I in each w'. But then it is also true at I in w, since the inertia worlds are the same up to I.

Reformulated in terms of events, Dowty's analysis states roughly that an event is in progress iff it continues on to its conclusion in all the worlds where nothing unusual happens. Seen in this way, there are three components of his analysis which come to the fore: first, the progressive relates an event in progress and a corresponding complete event; second, since the complete event need not be actual, the progressive creates an intensional context; and third, the progressive presupposes some normality conditions. We will return to some technical problems with Dowty's notion of normality when we review (Landman, 1992); but first, let us consider (Parsons, 1990), where the first two components are challenged.

2.2.2 Parsons (1990)

In contrast to Dowty, Parsons (1990, pp. 170-178) presents an analysis of the progressive in terms of actual events and two aspectual relations between events and intervals: an event can **culminate** at an interval, meaning it gets completed there, or it can merely **hold** at an interval, possibly incompletely. Thus for

[38] The following review owes much to (Landman, 1992).

[39] Note that *Inr* is the function which returns the set of inertia worlds for each world-time pair.

Parsons, the antecedent and consequent of (2.49b) are simply translated as follows:

(2.55) (a) $\exists x[\text{Circle}(x) \wedge \exists e \exists i[i < now \wedge \text{Drawing}(e) \wedge \text{Agent}(e) = j \wedge \text{Theme}(e) = x \wedge \text{Hold}(e, i)]]$

(b) $\exists x[\text{Circle}(x) \wedge \exists e \exists i[i < now \wedge \text{Drawing}(e) \wedge \text{Agent}(e) = j \wedge \text{Theme}(e) = x \wedge \text{Cul}(e, i)]]$

The logical form in (2.55a) corresponds to *Jack was drawing a circle*, the one in (2.55b) to *Jack drew a circle*. This example reveals Parsons' analysis of the progressive, which is to change the relation Cul to the relation Hold, in the case of accomplishments; in the case of activities, it does nothing at all. Since Cul entails Hold but not vice-versa, he successfully accounts for both (2.49a) and (2.49b).[40]

Because Parsons' analysis is so simple it is worth considering carefully. Since his analysis only makes reference to actual events, it does not involve an intensional context. This is quite clear in (2.55a), which requires there to be an actual circle and an actual event of Jack drawing that circle which holds (but does not necessarily culminate) at some past interval. Note that in order for this to make sense, Parsons must therefore commit himself to the existence of actual but incomplete objects—which he in fact does. Parsons justifies this rather unusual move as follows (pp. 173-174):

> If Mary is building a house, then her building event has an object that is a house, and so there is a house that she is building. Now suppose that Mary is struck down by lightning with the house only one-fourth finished. The objector takes me to the location and demands, "Where is the house? All I see is a foundation and portions of some wall-framing!" My answer is that we are looking at the house—it is merely an incomplete or unfinished one. [...]
>
> In northern California one can visit Jack London State Park and see the house that Jack London was building when he died. At least this is what the tourist guides say. It isn't much of a house—only a foundation and parts of some walls. But native speakers of English call it a house. What evidence could there be that they are wrong?

Even if we accept Parsons' justification for accepting actual but incomplete objects, this need not convince us that the progressive does not create an intensional context, but only that we should look further for more convincing evidence. This matter is taken up by Landman (1992, pp. 8-9), who considers the following example:

(2.56) God was creating a unicorn when He changed his mind.

[40] At a less explicit level, Moens (1987, ch. 8) seems to propose much the same thing with his pairs start-cul and start-stop. This matter is discussed further in chapter 5.

First Landman notes that *create* is extensional, insofar as *God created a unicorn* entails the existence of a unicorn, so that if we find evidence of intensionality it must be due to the progressive. Then he observes that unlike the case of the partially drawn circle, or Jack London's house, it is plausible enough that the unicorn does not come into existence until the very end, after all the appropriate incantations, and thus that if the process is interrupted there is no partial unicorn at all. For this reason Parsons' appeal to incomplete objects will not work in this case.

Landman's argument that the progressive does indeed create an intensional context appears to be conclusive.[41] As it turns out, Parsons' claim that the progressive need not relate events in progress to corresponding complete events does not seem to be tenable either; for expository purposes, however, I will postpone discussion of this matter until chapter 5.

2.2.3 Bach (1986)

Bach (1986, p. 12) observes that the imperfective paradox has a parallel problem in the nominal domain, which he calls the **partitive puzzle**:

(2.57) $\not\models$ (a) We found part of a Roman aqueduct.
(b) There was a Roman aqueduct.

Bach notes that (2.57a) does not entail that there ever existed an actual Roman aqueduct which had as a part whatever it was they found; (2.57a) could be true even if the construction of the aqueduct was interrupted forever by hordes of barbarians from the north.

Bach does not go on to formalize this notion. Landman (1992, pp. 13-14) suggests that one way to do so would be to introduce a realization relation between actual entities and types of entities:

(2.58) *This is part of a Roman aqueduct* is true iff this (actual entity) partially realizes the type of Roman aqueducts.

Noting that not every actual stone is part of a Roman aqueduct, Landman argues that some (perhaps necessarily vague) sufficiency requirement should be added:

(2.59) x partially realizes type X iff x realizes sufficiently much of X.

Extending this idea to the progressive, Landman comes up with the following proposal:

[41] Landman also considers some further evidence concerning the parallel anaphora possibilities with intensional verbs and verbs appearing in the progressive.

(2.60) **The Part-of Proposal:**
Mary is crossing the street is true iff some actual event realizes sufficiently much of the type of events of Mary's crossing the street.

Landman observes that such a proposal does address the problem of modality, since something that partially realizes the event type of God's creating a unicorn need not necessarily realize the unicorn. However, he (pp. 14-17) then points out that some notion of normality is still necessary if we are to account for the problem of 'non-interruptions'. The problem with non-interruptions is that an actual completed event, no matter how unusual, should make the progressive true in retrospect. To illustrate the problem, he considers the following example under two different scenarios:

(2.61) Mary was crossing the Atlantic.

Scenario 1: Mary gets into the water in France; she is not a very good swimmer; she swims for an hour and sinks.

Scenario 2: Same scenario, except that instead of drowning, she manages to make it across through divine intervention.

Landman observes that it is quite possible to judge (2.61) as false under the first scenario and true under the second. However, under both scenarios, equally much of the event type of Mary's crossing the Atlantic would seem to be realized at the time of drowning, and thus the part-of proposal by itself appears to be insufficient to distinguish the two. We will return to discuss these subtleties further after examining Landman's own proposal.

2.2.4 Landman (1992)

At this point we have established that an adequate analysis of the progressive must have a modal component, and that the part-of proposal appears to be missing a notion of normality, at least. Landman (1992) discusses several proposals which attempt to formalize the requisite notion of normality and finds that all suffer from one problem or another. Noting that these problems pull in different directions, he comments that "dealing with them all is a bit like trying to perform a juggling act while sailing between Scylla and Charybdis"; undaunted, he nevertheless goes on to present an analysis of the progressive which is intended to combine the insights of these prior proposals.

We begin by returning to Dowty's analysis, which Landman (p. 3) paraphrases as follows:[42]

(2.62) **Dowty's Proposal:**
Mary is crossing the street is true in a world w at an interval i iff in

[42] Landman omits the condition that i should be a non-final subinterval. The significance of this omission is unclear to me.

every inertia world v for w at i this interval i is a subinterval of a larger interval where *Mary cross the street* is true.

On Dowty's analysis, *Mary is crossing the street* is true if Mary crosses the street in all the worlds where nothing unexpected happens. This analysis successfully deals with examples like the following one:

(2.63) Mary was crossing the street, when a thunderbolt from heaven struck her down.

However, as Vlach (1981) points out, Dowty's analysis does fare not so well with more mundane examples:

(2.64) Mary was crossing the street, when a truck hit her.

The problem here is that if we consider an interval moments from impact, then in all the worlds in which nothing unexpected happens Mary gets hit rather than managing to cross the street; as a result, Dowty's analysis incorrectly predicts that (2.64) should be false.

What seems to go wrong in Dowty's theory is that it requires everything to take its normal course, including the truck. This leads Landman (p. 11) to suggest requiring only that Mary's crossing proceed normally, as well as everything else that doesn't interfere:

(2.65) **The Normality Proposal:**
Mary is crossing the street is true in w at i iff some process of crossing by Mary, e, is going on in w at i and in every inertia world for w and e at i, i.e., in every world where e is allowed to follow its normal course, there is an interval surrounding i where *Mary cross the street* is true.

This proposal handles Vlach's problem: although Mary's crossing is interrupted in the our world, it needn't be in those worlds where it is allowed to proceed normally.

Vlach's proposal, in contrast, does not rely on normality. Landman (p. 12) restates it as follows:

(2.66) **Vlach's Proposal:**
Mary is crossing the street is true in w at i iff there is a process going on in w at i such that if it were to continue, it would eventually cause *Mary cross the street* to become true.

Taken literally, Vlach's proposal fares no better than Dowty's: if we take an interval fifteen seconds before impact, whatever is going on at that interval does continue beyond that interval—up until impact—but it does not eventually cause *Mary cross the street* to become true.

Landman observes that the subjunctive in Vlach's proposal should not be applied at the interval of evaluation but rather at the interval of interruption, and thus suggests the following correction:

(2.67) **The Subjunctive Proposal:**
Mary is crossing the street is true in w at i iff there is a process going on in w at i such that if it were to continue beyond the interval where it stops in w, it would eventually cause *Mary cross the street* to become true.

Assuming a standard analysis of counterfactuals, e.g. (Stalnaker, 1968), the subjunctive proposal states that *Mary is crossing the street is true* if in the closest world where her crossing is not interrupted *Mary cross the street becomes true*. Since the closest such world is plausibly the one with the truck somewhere else, (2.64) turns out true.

As we have just seen, the normality proposal and the subjunctive proposal fare equally well on Vlach's problem. Returning to the problem of non-interruptions, however, we find that the subjunctive proposal has a decided advantage. Recall that in the second scenario, against all odds and through divine intervention Mary manages to cross the Atlantic. In this case the real world is unlike the inertia worlds where she drowns; because the normality proposal looks to the inertia worlds to see whether Mary crosses, it incorrectly predicts that (2.61) should be false under this scenario. In contrast, the subjunctive proposal correctly predicts that (2.61) should be true in this case, since the process of Mary's crossing continues on to completion; that it did so through divine intervention is irrelevant, as normality plays no role in this.

Lest we think the issue settled, Landman introduces another problem, the 'problem of continuations'. Adding a twist to Vlach's story, he posits a second truck in a second lane, just behind the first, with an equally inattentive driver. In this situation both sentences below are intuitively true:

(2.68) (a) Mary was crossing the street, when a truck hit her.
(b) If the first truck hadn't hit her, the second truck would have.

If (2.68b) is true, then in the closest world where Mary's crossing is not stopped by the first truck it is stopped by the second; consequently, the subjunctive proposal incorrectly predicts (2.68a) to be false.

To correct this problem, Landman (p. 18) considers changing the subjunctive proposal so that we look not just at what would have happened had the event not been interrupted, but at what would have happened had there been no interruption of danger at all. However, he notes that this modification is too strong, as shown by the following example:

(2.69) Mary was wiping out the Roman army.

Assuming Mary is a person of moderate physical capabilities, she has no chance of succeeding, and thus we judge (2.69) false. However, if we suppose that Mary kills a few soldiers before getting killed herself, then if we remove not just this soldier but all danger of interruption we remove the rest of the Roman army; in

that world, Mary has indeed wiped out the Roman army, and thus the modified subjunctive proposal incorrectly predicts (2.69) to be true.

As advertised, Landman proposes a blend of the above proposals, specifically the part-of, normality and subjunctive ones. Let us consider the contribution of each in turn.

Landman (pp. 22-24) assumes a set of events partially ordered by the relations **part-of** and **stage-of**. The part-of relation is the one that holds, say, between Hanny Schaft's acts of resistance and World War II. The stage-of relation, which Landman suggests is reminiscent of Carlson's (1977a) notion of stages of an individual, is a more intimate one: an event is a stage of another event if the second can be regarded as a more developed version of the first.[43] This more constrained relation, Landman argues, is necessary to determine when event continuations stop. He observes that it is not sufficient to say, as we might at first be inclined, that the continuation of an event e stops at interval i in world w iff there are no events of which e is a part that are realized in w extending beyond i. To see this, note that Hanny Schaft's acts of resistance stopped well before the end of World War II; however, since her acts of resistance are part of WWII, this suggestion would entail that they continued on. This is where stages come in: rather than using larger parts, Landman suggests using larger stages.[44] This results in the following definition:

(2.70) e **stops** at i in w iff no event of which e is a stage goes on beyond i in w.

In a manner reminiscent of the event-indexed inertia worlds, Landman (pp. 24-26) assumes that the model assigns to event e in world w a set of worlds $R(e, w)$, the set of **reasonable options** for e in w:

(2.71) $v \, \varepsilon \, R(e, w)$ iff there is a reasonable chance on the basis of what is internal to e in w that e continues in w as far as it does in v.

As we saw earlier, non-interruptions are a problem for the normality proposal: if an event continues to completion, even against all odds, we may judge the progressive true; the fact that the real world is not a reasonable option is irrelevant. This leads Landman to suggest that if normality is to play a part in the semantics of the progressive, it must do so within the confines of the subjunctive proposal.

Landman (pp. 26-27) incorporates reasonable chance into the subjunctive proposal by defining the **continuation branch** of an event e in w, $C(e, w)$, as follows:

[43] Note that Landman does not assume that events and stages are ontologically distinct entities; instead he just assumes there are two distinct relations (part-of and stage-of) that hold between events.

[44] It is worth noting that Krifka (1989) seems to have both part-of notions in mind, but in different parts of the paper. While conceptually problematic, this conflation appears to have only minor technical repercussions.

(2.72) $C(e, w)$ is the smallest set of pairs of events and worlds such that

1. for every event f in w such that e is a stage of f, $\langle f, w\rangle \, \varepsilon \, C(e, w)$; the continuation stretch of e in w;
2. if the continuation stretch of e in w stops in w, it has a maximal element f and f stops in w. Consider the closest world v where f does not stop:
   - if v is not in $R(e, w)$, the continuation branch stops.
   - if v is in $R(e, w)$, then $\langle f, v\rangle \, \varepsilon \, C(e, w)$. In this case, we repeat the construction.

He then provides the following explanation:

> So the idea is that you follow e in our world: if its continuation stops, you follow it in the closest world where it doesn't stop, if that world is a reasonable option for e in w; if the continuation stops in that world, you go to the closest world again, if it's reasonable, and you continue until either in some world it doesn't stop (and then you stay in that world) or, the more normal case, you reach a point where going to the closest world is no longer a reasonable option and you stop there.

Having defined this notion, Landman provides at long last the following semantics for the progressive:

(2.73) $[\![\text{PROG}(e, P)]\!]_{w,g} = 1$ iff $\exists f \exists v : \langle f, v\rangle \, \varepsilon \, C(g(e), w)$ and $[\![P]\!]_{v,g}(f) = 1$

Let us now revisit the most difficult problems, beginning with the problem of non-interruptions. On the first scenario of (2.61) we follow Mary until she sinks. Now we look at the first world where she continues; on the basis of her capacities, she gets a bit further before sinking again. We look again at the closest world where she continues; maybe we can still conceive of her getting a bit further, but soon it is no longer reasonable to think she has a chance, and thus we judge (2.61) false. On the second scenario, miracle of miracles, she crosses. In this case we follow Mary in the real world until the event is completed, and judge (2.61) true. Note that example (2.69) is essentially the same as the first scenario; while Mary might plausibly kill a few more soldiers, soon she weakens and it becomes implausible to think of her lasting further. Turning now to the problem of continuations, we follow Mary until she gets hit by the first truck, then zap away the first truck and let her continue; this time she gets hit by the second truck, so we zap away the second truck and again let her continue; this time it is plausible enough that Mary finishes crossing the street, and thus we judge (2.68) true.

While decidedly clever, Landman's theory is unfortunately somewhat fragile: if we try to get him to juggle one more problem, he appears to slip and fall

directly into the clutches of Scylla. Let us suppose, adding a wrinkle to (2.69), that Mary is a tireless robot, who has a 90% chance of defeating any given Roman soldier. Let us also suppose that for some reason (machismo?) the soldiers are constrained to fight individually, in succession. On this scenario, whenever Mary gets killed, we turn to the next closest world where it is always equally plausible that she continues. Assuming the Roman army to be finite, it is soon wiped out. Nevertheless, from elementary probability theory (and even our naive intuitions) we know this to be highly implausible, and judge (2.69) false, contra Landman's proposal.

Having seen the perils that awaited Landman, I will steer clear of the entire zone of Scylla and Charybdis. Rather than trying to resolve when the progressive may be used to describe an event in progress, I will instead focus on how a sortal distinction between events and processes can be used to account for the pattern of entailments revealed by the imperfective paradox. To do so, I will by and large remain within the confines of the part-of proposal, which of course requires responding to Landman's criticisms of this particular approach.

Recall that Landman invokes the problem of non-interruptions, illustrated by example (2.61), to demonstrate the inadequacy of the part-of proposal. In so doing, he implicitly assumes it is a simple matter to determine whether an event is part of a larger, possibly incomplete event: that is, under both scenarios, he implicitly assumes that the same event of Mary swimming away from France for an hour is part of a larger event of Mary crossing the Atlantic, regardless of whether God (or some other divine being) intends to help her along. In contrast, I will assume that it is a difficult matter to determine whether an event is part of a larger, possibly incomplete event, in hopes of localizing the perils that befell Landman and his predecessors to this classification.[45]

Let us now examine why it might make sense to do so. As Richard Oehrle has pointed out to me, it is important to consider what the speaker is assumed to know about a situation when asserting the truth of a description. To illustrate, he considers the following example. Suppose that a couple steps into the street and starts to waltz, round and round. Are they crossing the street? At this point it might seem silly to say so. But, if we see them reach the other side, we might in retrospect describe the situation by saying they were waltzing their way across the street. On the other hand, if we left the scene before the reach the other side, we would still have no reason to assert that they were crossing the street, irrespective of what happened afterwards. This example also brings up the issue of intentions: if someone told us that the couple meant to waltz across the street, then we would have reason to assert they were crossing it.

In an appendix (pp. 30-31), Landman acknowledges a similar point in his discussion of **perspectives**: noting that someone might object to the assertion that Mary was crossing the Atlantic before the divine intervention, he suggests

[45] In other words, in an attempt to abstract away from the difficulties inherent in formalizing continuations, I will assume that the truth of the progressive can be reduced to the truth of the part-of relation, which it is up to each particular model to decide.

that whether one is happy to assert the truth of her crossing depends on one's perspective of the context. What Landman does not consider is how this issue affects his evaluation of the part-of proposal. In the case at hand at least, the part-of proposal seems to fare quite well: if we see the couple cross the street, or if we are told that they intend to do so, then we have reason to judge their waltzing to be part of an event of crossing the street, and thus we have reason to assert in retrospect that they were crossing the street; on the other hand, if we leave the scene before they reach the other side, and without anyone informing us of their intentions, then we do not have sufficient knowledge to judge their waltzing to be part of an event of crossing the street—for all we know, they may have been intending to return to the same side the whole time (before they got hit by a truck).

While it is perhaps unlikely that the part-of proposal will ultimately suffice on its own, I take these examples to show the utility of pushing the approach as far as it will go.

2.2.5 Jackendoff (1991)

To conclude our discussion of the imperfective paradox, I will now briefly review Jackendoff's (1991) analysis. Recall that Jackendoff assumes processes and events are two different types of situations, distinguished by the feature [±b] for 'bounded'. Extracting from his discussion of aspectual verbs, we may safely assume that his encoding of the progressive would select for [−b] situations, as sketched below:

(2.74) [PROG([$_{-\mathrm{b}}$ Jack push a cart])]

This raises an interesting question, namely what happens when the progressive is used with an event expression. To resolve the resulting feature clash, Jackendoff suggests that a **rule of construal** is invoked. For Jackendoff, rules of construal are general principles which license the addition of non-lexical material to the conceptual structure of a sentence. In this case, a rule of construal could be used to insert the feature GR, for 'ground from',[46] the approximate inverse of COMP:

(2.75) [PROG([$_{-\mathrm{b}}$ GR([$_{+\mathrm{b}}$ Jack draw a circle])])]

In order to account for the imperfective paradox, Jackendoff suggests that GR be treated as an 'extracting' function, i.e. one that does not convey existential entailments. Because GR appears in (2.75) but not (2.74), this could plausibly account for their differing entailments.

Let us examine how this is supposed to work in the case of the aspectual verb *stop*, which Jackendoff considers in some detail:[47]

[46] Jackendoff likens GR to the Universal Grinder discussed in (Pelletier and Schubert, 1989), whence the name.

[47] For simplicity, I have replaced his feature BD for boundary event with the feature STOP.

(2.76) (a) [STOP([$_{+b}$ COMP([$_{-b}$ GR([$_{+b}$ RUN TO THE STORE])])])]

(b) "This unpacks as follows: the bounded Event *run to the store* is ground up by GR into a process; some of this process is gathered up into a unit by COMP; the end of this unit is picked out by STOP. It is this boundary Event that is expressed by *stop running to the store*. In turn, since *run to the store* has been ground up, there is no inference to completion."

For present purposes, there are two important observations to be made about Jackendoff's explanation in (2.76b). The first one concerns the idea of COMP 'gathering up' some of a process: if we are to take this suggestion literally, then we must assume that distinct events can be in the composed-of relation to the same process.[48] The second one concerns how we are supposed to understand GR: in the case of events, at least, the composition of COMP and GR appears to be equivalent to the part-of relation employed by Hinrichs and Krifka. We will return to this matter in section 2.5.

While Jackendoff's approach is an interesting one, it is worth reiterating that in the absence of a model-theoretic interpretation his analysis leaves something to be desired. For example, take the case of (2.75): although we don't want *Jack was drawing a circle* to entail *Jack drew a circle*, we do want it to entail *Jack drew something* and other such sentences. This problem is yet another that an uninterpreted sortal approach cannot adequately address.

2.3 Aspectual Type Coercion

We have just seen that Jackendoff's (1991) analysis of the imperfective paradox makes use of a rule of construal which operates in some cases but not others. Jackendoff likewise makes use of an opportunistic rule of construal to account for the iterative reading of *The light flashed for hours*. A similar, independently developed account is found in (Moens, 1987), where Moens (pp. 55-68, 50-52) invokes the the notion of **type coercion** to accomplish the same ends. Both of these more informal approaches differ from the model-theoretic ones discussed in the previous sections, wherein the progressive and *for*-adverbials are generally given more unitary analyses. For this reason these coercion-based approaches may, at first glance, appear overly complex and less explanatory. In this section I will attempt to counter this conception; in the process, I will also try to diffuse some of the arguments that have been made in favor of a quantificational analysis of *for*-adverbials.

Type coercion becomes less peculiar to the extent that it is seen as a general principle of interpretation. Indeed, Moens suggests that coercion is involved not only in the semantics of the progressive and *for*-adverbials, but also in

[48] Note that this answers our question of whether we should assume that distinct material entities can be composed of the same substance, as promised in the last section.

the perfect, *in*-adverbials, tense, *when*-clauses, and so forth. Jackendoff even suggests that his rules of construal extend to instances of metonymy such as Nunberg's (1979) *The ham sandwich in the corner wants another cup of coffee*, where *the ham sandwich* refers to the customer who ordered a ham sandwich.

Of course, the generality of type coercion can also become a burden, since care must be taken to avoid overgeneration. For example, Moens and Steedman (1988, p. 20) point out that their culminated process to process coercion, which is meant to cancel the entailment of completion, must be licensed by the progressive rather than freely applying. To see this, consider the following pair:

(2.77) (a) Jack was solving the puzzle when he fell asleep.

(b) ? Jack solved the puzzle for a while.

Example (2.77b) does not seem to have a reading in which Jack engages in a process of puzzle-solving for some time without ever solving the puzzle; this is in stark contrast to (2.77a), which has this reading as its most natural one. If type coercions were to apply completely freely, there would be no way to explain this contrast.

While Moens and Steedman recognize the need for grammaticizing some coercions, they do not propose a mechanism for doing so. Such a mechanism is discussed in (Pustejovsky, 1991a), where Pustejovsky suggests assigning a set of coercion operators Σ_α to each expression α. These operators are incorporated into an extended version of function application as shown below:

(2.78) **Function Application with Coercion:**
If α is of type $\langle b, a\rangle$, and β is of type c, then

(a) if type $c = b$, then $\alpha(\beta)$ is of type a.

(b) if there is a $\sigma \, \varepsilon \, \Sigma_\beta$ such that $\sigma(\beta)$ results in an expression of type b, then $\alpha(\sigma(\beta))$ is of type a.

(c) otherwise a type error is produced.

Note that Pustejovsky's method may be used to resolve licensing issues by simply assigning different operators to different expressions.

Having established that type coercion is a reasonable option to pursue, let us now consider a couple of its advantages. First, type coercion provides an elegant way to capture certain ambiguities. Consider the following example and schematic interpretations from (Jackendoff, 1991):

(2.79) Bill stopped running to the store.

(a) [STOP([COMP([GR([Bill run to the store])])])].

(b) [STOP([COMP([PL([Bill run to the store])])])].

The interpretation in (2.79a) should be familiar from our discussion of the imperfective paradox: the operator GR maps the event of Bill running to the store to the process which makes it up; it is this process which is said to stop, potentially before completion. The interpretation in (2.79b), in contrast, contains the plural operator PL, which yields an iterative process of Bill running to the store again and again; it is this sequence which is said to stop under this reading. Note that the ambiguity in (2.79) is not due to the syntax, but rather to which rule of construal is invoked.[49]

A second advantage of type coercion is that it enables one to account for readings that could not plausibly be derived from the lexical meanings alone. An excellent example of this from (Moens and Steedman, 1988, p. 21) appears below:

(2.80) It took me two days to play the "Minute Waltz" in less than sixty seconds for more than an hour.

Moens and Steedman suggest that *it took me two days* adds to the interpretation (by way of two coercions) the process leading up to the feat of repeated "Minute Waltz" playing, which world knowledge suggests is practicing; it is this process which is said to take two days. They also suggest that the repetition is a result of a coercion induced by the *for*-adverbial, along the same lines as Jackendoff. Now, while this repetition could be argued instead to be due to the *for*-adverbial, as have Hinrichs (1985) and more recently Moltmann (1991), to my knowledge no one has proposed an analysis of *take two days* that would account for the above preparatory process interpretation; such an analysis would be highly implausible indeed.

Interestingly, this last example suggests that type coercion could be used to diffuse some of the arguments that have been made in favor of a quantificational analysis of *for*-adverbials. These arguments have been made most recently and adamantly by Moltmann (1991); of her numerous arguments, I will only directly address the two most relevant for present purposes. First, Moltmann argues (pp. 646-647) that since *for*-adverbials exhibit scope ambiguities with adverbial quantifiers, this provides evidence against treating them as simple eventuality predicate modifiers. This argument does not seem to be a particularly strong one, however, since *in*-adverbials (which are assumed to be eventuality predicate modifiers) also exhibit scope ambiguities, as evidenced by (2.80); cf. also the difference between *Jack awoke twice in ten minutes* and *Jack awoke in ten minutes twice*.

A second and related argument concerns the following contrast:

(2.81) (a) For several years a lot of students complained about the requirements.

[49]Or, in the spirit of Pustejovsky's proposal, the ambiguity could be said to reside in the set of operators assigned to *stop*.

(b) A lot of students complained about the requirements for several years.

Moltmann observes that for many speakers the preferred reading of (2.81a) need not involve a single group of students, in contrast to (2.81b); furthermore, she asserts that an eventuality predicate modifier approach (such as Krifka's) could only account for the latter of these two readings. On her account, this difference comes out as an everyday quantifier scope ambiguity. Note, however, that type coercion may provide a way out of this particular impasse, insofar as an iterative or perhaps generic operator introduced by coercion could allow the existentially quantified participants to vary in the successive situations. Of course, adding an iterative operator would not by itself account for the differing preferred interpretations in (2.81), but it would at least open up a number of viable options, none requiring *for*-adverbials to be analyzed quantificationally.[50]

2.4 Individuation

We return now to the problem of aspectual composition in order to focus on some subtle problems concerning individuation and the property of homogeneous reference.

In their discussion of mass expressions, Pelletier and Schubert (1989, p. 329) point out that some count expressions are like mass expressions in that they fail to individuate, i.e. they fail to mark off one instance of a count expression from another. Examples of such expressions include *thing*, *object*, *entity*, and so forth. In Krifka's terminology, non-individuating expressions are not quantized: a book is a *thing*, but so are its proper subparts, such as its cover, its binding, etc.

The problem with such non-individuating expressions is that they make tests for quantized or homogeneous reference inexact as tests for whether a nominal expression is count or mass. As we shall see below, there are analogous non-individuating verbal expressions; these likewise make tests for reference properties inexact as tests for whether a verbal expression is an activity one or an accomplishment one. While this problem has been recognized by several authors focusing on the semantics of aspect, including Mittwoch (1982), Moens (1987), Verkuyl (1989) and Krifka (1989), only Verkuyl has systematically addressed the problem.[51]

Mittwoch discusses non-individuating verbal expressions containing the vague quantifier *something*, as in (2.82) below:[52]

[50] Although we will not consider the problems Moltmann raises with binding and indexicals here, it seems that coercion-induced quantificational operators could be used to address these issues as well.

[51] Recall from section 2.1 that Verkuyl does not make use of shared reference properties, relying instead on his notion of 'specified quantity'.

[52] For consistency, I have taken the liberty of changing *John* to *Jack* in this example.

(2.82) Jack wrote something in 10 minutes which it took me half an hour to translate.

Much like the nominal expressions above, the verbal expression *Jack write something* fails to individuate, i.e. it fails to mark off one instance of such an event from another—if there is an event of Jack writing something, then all the subevents of that event (down to a certain limit in size) will also be events of Jack writing something. In this respect, the expression *Jack write something* behaves like an activity expression; nevertheless, its cooccurrence with *in*-adverbials shows that it should be considered an accomplishment one.

To take another example, consider the following sentence:

(2.83) Jack ran more than a mile in less than five minutes.

Example (2.83) shows how vagueness can be a problem: two events of Jack running more than a mile put together is of course another event of Jack running more than a mile; furthermore, all subevents of an event of Jack running more than a mile are also events of Jack running more than a mile, if we take our limit in size to be one-mile long events.

More insidious examples may even be constructed, involving reference to objects which are 'self-similar'. Consider (2.84):

(2.84) Jack typed a sequence of characters in thirty seconds (which it took me two minutes to write by hand).

The problem here is that subsequences of characters are also sequences of characters; for this reason the expression *Jack type a sequence of characters* also fails to individuate.

These non-individuating accomplishment expressions are problematic for the accounts of Dowty, Hinrichs and Krifka, since their treatments of *for*-adverbials rely upon tests for homogeneous reference in one way or another. Let us now examine in some detail why such expressions make these tests inexact; since Krifka's account is perhaps the most straightforward, I will look at his as our case in point.

Extrapolating from (Krifka, 1989, p. 93), we may assume the following translations for *Jack write a letter* and *Jack write something*:

(2.85) (a) *Jack write a letter*:
$\lambda e \exists x[\text{write}'(e) \wedge \text{letter}'(x, 1) \wedge \text{PAT}(e, x) \wedge \text{AG}(e, j)]$

(b) *Jack write something*:
$\lambda e \exists x[\text{write}'(e) \wedge \text{thing}'(x, 1) \wedge \text{PAT}(e, x) \wedge \text{AG}(e, j)]$

Krifka's translation of *for ten minutes* is shown below:

(2.86) *for ten minutes* translates as
$\lambda P \lambda e[P(e) \wedge \text{minutes}'(e) =$
$10 \ / \ \text{QMOD}(P, \lambda P \lambda e[P(e) \wedge \text{minutes}'(e) = 10])]$

Recall from section 2.1 that QMOD enforces a well-formedness condition requiring the modified event predicate P to be non-quantized. This condition correctly rules out the cooccurrence of *for ten minutes* with (2.85a), as long as we assume $\lambda x[\text{letter}'(x, 1)]$ is quantized; however, it fails to rule out the cooccurrence of *for ten minutes* with (2.85b): if we assume that $\lambda x[\text{thing}'(x, 1)]$ is non-quantized, then the event predicate in (2.85b) turns out non-quantized too, and thus the well-formedness condition is mistakenly satisfied.

There is an obvious potential remedy for this problem worth considering. One might be tempted to think that the problem with the event predicate in (2.85b) is the existential quantification over the object x playing the patient role; a simple remedy would then be to translate such event predicates using a free variable and imposing an existential closure condition higher up, à la (Heim, 1983):

(2.87) *Jack write something*:
$\lambda e[\text{write}'(e) \wedge \text{thing}'(x, 1) \wedge \text{PAT}(e, x) \wedge \text{AG}(e, j)]$

Note that this event predicate does turn out quantized, since the object x written by Jack is not allowed to vary from events to subevents.

Unfortunately, there are two serious problems with this potential remedy. First, to correctly account for mass terms and bare plurals, the problematic existential quantifier must remain:[53]

(2.88) *Jack write letters*:
$\lambda e \exists x[\text{write}'(e) \wedge \text{letters}'(x) \wedge \text{PAT}(e, x) \wedge \text{AG}(e, j)]$

For this event predicate to be non-quantized (as desired), the letters x written by Jack must be allowed to vary from events to subevents. This precludes a uniform treatment of NPs, and undermines the explanatory power of the theory.

The second problem is even more troublesome. As was discussed in the last section, sometimes the participants do vary from situation to situation; recall Moltmann's example, repeated below:

(2.89) For several years a lot of students complained about the requirements.

Accounting for this reading would seem to require the insertion of the problematic existential quantifier before the application of the *for*-adverbial. This shows the obvious remedy to be untenable as it stands.

As mentioned in section 2.1, Krifka does briefly sketch how Link and Bach's sortal distinction between individuals and quantities of matter might be used to address the issue of non-individuating count nouns such as *sequence*. Let us now examine this matter in detail.

Link (1983) and Bach (1986) both propose to distinguish between ordinary individuals and quantities of matter. To motivate this distinction, they consider examples such as the following one:

[53] Note that Krifka uses the term $\text{letters}'(x)$ as shorthand for $\exists n[\text{letter}'(x, n)]$.

(2.90) The gold making up Terry's ring is old but the ring itself is new (not old).

The fact that (2.90) is not contradictory suggests that the ring and the quantity of gold making it up should be considered distinct individuals. In a similar vein, Bach (p. 8) and Link (1987) also propose to distinguish between events and 'bounded processes' ('bits of process') analogous to the quantities of matter that constitute the 'material extensions' of (physical) objects. As a motivating example in this domain, Bach (pp. 9-10) considers two distinct but materially ('processually') equivalent events, one of Jones pouring poison into the water main—say, in order to rid waterbeds of bedfish—and one of Jones poisoning the populace. By letting these be distinct individuals, their proposal enables the first event to be an intentional one yet the second an unintentional one, without leading to a contradiction.

In Link and Bach's framework, individuals and quantities are sortally distinguished. For each sort, there is a separate part-of relation which induces a lattice structure on these entities; additionally, there is a 'materialization' function mapping from individuals to quantities which preserves this relation. The difference between these lattices is that only the former one is an atomic lattice, i.e. has minimal elements; this makes the individual lattice appropriate for modeling pluralities, and the quantity lattice appropriate for modeling portions of matter (which seem to lack such minimal elements, conceptually).

An advantage of Link and Bach's setup is that Jack and his arm can be both two things and one at the same time. This is because both Jack's arm and Jack himself can be minimal elements of the individual lattice, making the pair a plurality of cardinality two. At the same time, Jack's arm can be part of Jack in the quantity lattice, making Jack one again (whew!).

Returning now to the troublesome count noun *sequence*, Krifka (1989, p. 87) notes that Link and Bach's sortal distinction can be employed to ensure that its translation turns out quantized, since a sequence could then have proper subsequences standing in the quantity part-of relation to it whilst maintaining its status as a minimal element in the individual lattice.[54] As mentioned in section 2.1, however, Krifka does not consider whether this same sortal distinction might be useful in analyzing complex verbal expressions; moreover, he glosses over this distinction in the rest of (Krifka, 1989), and does not mention it at all in (Krifka, 1992). As it turns out, this move does not really help to resolve the problem of non-individuating accomplishment expressions. To see why, consider the following example:

(2.91) *Jack type a sequence*:
$\lambda e \exists x [\text{type}'(e) \wedge \text{sequence}'(x, 1) \wedge \text{PAT}(e, x) \wedge \text{AG}(e, j)]$

If we continue to use the quantity part-of relation in the eventuality domain, then an event of Jack typing a (non-trivial) sequence will still have subevents of

[54] This is extrapolated from Krifka's discussion of twigs.

Jack typing its subsequences, and thus the problem remains. Moreover, adding Link and Bach's distinction to the object domain does not even begin to address the problem with *Jack ran more than a mile*.

At this point, one naturally wonders whether the same sortal distinction might work in the eventuality domain as well. Note that one cannot simply use the individual part-of relation with all eventuality predicates; that would mistakenly turn the translation of *Jack ran* into a quantized predicate. Instead, one must appropriately link the sortal requirements of the eventuality predicate to those of the verbal arguments and modifiers. As Manfred Krifka has pointed out to me, the difficulty is how to do so in an explanatory fashion, as otherwise one has done nothing more than convert syntactic stipulations into semantic ones.

2.5 Interim Summary and Thumbnail Sketch

We have just seen that the problem of non-individuating accomplishment expressions is a troublesome one for the analyses of Dowty, Hinrichs and Krifka, where tests for individuation are assumed to correctly identify the class of accomplishment expressions. To my thinking, this should really come as no surprise, for the following reason. It has long been recognized that the meaning of a noun does not completely determine whether it is classified as count or mass in a language (such as English) which grammaticizes this distinction: besides the cases of non-individuating count nouns, there are the well-known idiosyncracies concerning, say, the count nouns *table*, *desk* and *chair* on the one hand and the mass noun *furniture* on the other. If the event/process distinction in the verbal domain is indeed the analogue of the count/mass distinction in the nominal domain, then we should expect some degree of idiosyncracy in both cases.

I take these facts to suggest that we cannot do without some stipulation at the lexical level. As such, the challenge is thus to account for the behavior of complex verbal expressions making as few additional stipulations as possible. By and large, it is this challenge which has motivated the development of the account to be presented in the next two chapters.

As mentioned in the introduction, the present account may be viewed as a synthesis of proposals by Jackendoff (1991), Hinrichs (1985) and Krifka (1989; 1992). At this point we are ready to flesh out this statement.

From Jackendoff, I will borrow the main idea of reducing the problem of aspectual composition to the problem of detecting derived sortal incompatibilities. As noted in section 2.1.5, since Jackendoff is rather vague on how this is supposed to work, doing so will require some interpretation of his approach. In particular, there is the thorny question of how we are to understand the sortal difference between events and processes on the one hand and objects and substances on the other. As we saw in the previous section, the distinction that Link and Bach suggest, and that Krifka briefly considers, does not seem to be

of much help.

Fortunately, there is another such distinction we might try, namely Carlson's (1977a) distinction between individuals (and/or stages) and kinds. In section 2.1.2, as the reader may recall, we saw that Hinrichs makes use of this distinction in his treatment of the nominal system but does not do so in the verbal system; instead, he opts to follow Dowty (1979) in relying on an indirect test for homogeneous reference derived from a quantificational analysis of *for*-adverbials. Interestingly, Hinrichs does not examine nominal measure phrases, and thus the issue of whether *for*-adverbials should receive an analogous treatment does not even arise.

Adding another piece to the puzzle, we noted in sections 2.1.5 and 2.1.3 that both Jackendoff and Krifka propose to analyze nominal and verbal measure phrases in a uniform fashion. In his discussion of the former ones, Krifka (1989, p. 82) states the following intuition:

> The measure phrase serves to 'cut out' entities of a certain size from a continuum of entities which fall under the head noun. If the head noun is quantized, then there is no such continuum, and the application of the measure phrase therefore should be infelicitous.

In section 2.1.3 we saw how Krifka goes on to formalize this intuition in terms of quantized modification; in the previous section, however, we saw that this approach to measure phrases ultimately falls prey to the problem of non-individuating accomplishment expressions.

To complete the puzzle, we need only observe that Krifka's intuitive description of the role of measure phrases can be reconstructed in a much more direct fashion, along the lines of the present conception of Jackendoff's approach. The idea is simply to let the more abstract entities in our ontological pairs—that is, the processes, substances, and their analogues in the path domain—be the continua, and to let Jackendoff's 'composed-of' mapping do the 'cutting';[55] as we shall see, this suggestion is tantamount to viewing Jackendoff's 'composed-of' mapping as an extended version of Carlson's 'realization' relation. Naturally, this approach will successfully avoid the problem of non-individuating accomplishment expressions, since tests for homogeneous reference will no longer play a part in detecting semantic anomalies in such cases—sortal clashes will be utilized instead.

Now that the puzzle is complete, the reader will no doubt be wondering just what he or she is looking at, exactly. At this point I will only consider the five most pressing questions which naturally arise, and, even so, only to mention which ones will receive answers in the next two chapters.

First, one should rightly ask whether and how the sortal requirements of eventuality predicates can be derived in a principled fashion: as hinted at above,

[55]Recall that this mapping is what distinguishes measure phrases from ordinary restrictive modifiers in Jackendoff's approach.

these requirements will be shown to be derivable using only lexical stipulations.

Second, one is entitled to wonder how seriously to consider the idea of taking processes and their analogues in the path domain to be essentially like substances, understood as not unlike Carlson's kinds: reasonable though this doubt may be, I will argue nonetheless that these more abstract entities may be given a natural and uniform treatment in terms of **equivalence classes**, a notion that we will also find useful in formalizing paths along the lines suggested in Habel (1990).

The remaining questions (alas) will not be answered in the present study, but are nevertheless important enough to be explicitly acknowledged. These questions concern: third, how the present account relates to those of Carlson and his sucessors, where issues pertaining to genericity are of central concern; fourth, what role remains for Link and Bach's individual/quantity distinction and Carlson's individual/stage distinction; and fifth, where dynamic notions of definiteness à la Kamp and Heim fit into the picture. To keep matters manageable, I will say very little about these issues in the ensuing chapters.

To wrap up this section, it remains only to mention the aspects of the approach which do not fit neatly within this picture. These are as follows.

First, there is the imperfective paradox. As mentioned in section 2.2.4, I will focus on how the present sortal approach to aspectual composition can be made to mesh with Bach's part-of proposal, which was argued to be worth pursuing further. What I will show is that the distinction between events and processes suggested above, coupled with a simple-minded treatment of modality, suffices to account for the pattern of entailments revealed by the imperfective paradox. By and large, however, the analysis will not help to resolve the thorny issue of when the progressive may be used to describe an event in progress.

Second, there is the interaction of aspect with tense. Here again I will make do with a simple-minded and uncontroversial analysis, focusing only on how to capture the sortal restrictions which may be seen to arise from the present tense's reference to the moment of utterance. As such, I will leave aside not only the perfect but also the relevance of tense and aspect to discourse interpretation.[56]

Third, there is the issue of plurals. In this case, I will by and large just extend Hinrichs' adaptation of Link's (1983) analysis. As we shall see, this may be done without imposing a distinction between individuals and quantities of matter (or between individuals and stages); for the sake of simplicity then, I will not include these distinctions in the ontology, as hinted above. Moreover, I will also remain silent on the issues of cumulative, collective and group predication, which likewise appear to be largely orthogonal to the present concerns.

Finally, there is the issue of aspectual type coercion. As we saw in section 2.3, aspectual type coercion has the potential to significantly increase the coverage of a sortally based analysis, but care must be taken to avoid overgen-

[56] Note, however, that I will address this latter issue in chapter 6, at least in a preliminary way.

eration. One way to do so would be to assign a set of coercion operators to each expression, as Pustejovsky suggests. While this would appear to solve the problem, he does not indicate how this set of operators should be compositionally specified. For this reason, I will employ the perhaps less elegant solution of lexicalizing these operators, leaving open the issue of how best to capture the obvious generalizations which exist across the resulting lexical entries.

2.6 Syntactic Desiderata

This section and the next list the syntactic (cooccurrence) and semantic (entailments) tests which the present approach sets out to explain.

2.6.1 *For*-Adverbials

For-adverbials cooccur with activity expressions and stative expressions:

(2.92) (a) Jack poured wort into the carboy for ten seconds.

(b) Jack filled carboys with wort for ten minutes.

(c) Jack ran along the river for ten minutes.

(2.93) (a) The carboy was full for ten minutes.

(b) Jack didn't fill the carboy for ten minutes.

(c) Jack was filling the carboy for ten seconds.

For-adverbials do not cooccur with accomplishment expressions under single event readings; as discussed in section 2.4, this is the case even if the accomplishment expressions are non-individuating:[57]

(2.94) (a) * Jack poured twenty liters of wort into the carboy for ten seconds.

(b) * Jack filled twenty carboys with wort for ten minutes.

(b′) * Jack filled the carboy with wort for ten seconds.

(c) * Jack ran to the museum for ten minutes.

(c′) * Jack ran two miles for ten minutes.

(2.95) (a) * Jack poured some amount of wort into the carboy for ten seconds.

(b) * Jack filled some number of carboys with wort for ten minutes.

(c) * Jack ran somewhere for ten minutes.

(c′) * Jack ran some distance for ten minutes.

[57]Note that some expressions appear to be ambiguous: *Jack read the newspaper* $\left\{ \begin{array}{c} for \\ in \end{array} \right\}$ *ten minutes*. I will assume that this ambiguity is lexical in nature. For an opposing viewpoint, cf. (Zucchi, 1993).

However, *for*-adverbials do cooccur with both accomplishment and achievement expressions under iterated interpretations.[58] Note that these readings often require some imagination; the examples below, for instance, become less strange if one imagines Jack is practicing for some contest:

(2.96) (a) Jack poured twenty liters of wort into the carboy for forty minutes.

(b) Jack filled the carboy with wort for forty minutes.

(c) Jack ran to the museum for forty minutes.

(d) Jack winked for ten seconds.

2.6.2 *In*-Adverbials

In-adverbials, in contrast to *for*-adverbials, do cooccur with accomplishment expressions, even the non-individuating ones:

(2.97) (a) Jack poured twenty liters of wort into the carboy in ten seconds.

(b) Jack filled twenty carboys with wort in ten minutes.

(b′) Jack filled the carboy with wort in ten seconds.

(c) Jack ran to the museum in ten minutes.

(c′) Jack ran two miles in ten minutes.

(2.98) (a) Jack poured some amount of wort into the carboy in ten seconds.

(b) Jack filled some number of carboys with wort in ten minutes.

(c) Jack ran somewhere in ten minutes.

(c′) Jack ran some distance in ten minutes.

Again in contrast to *for*-adverbials, *in*-adverbials do not ordinarily cooccur with activity expressions:

(2.99) (a) * Jack poured wort into the carboy in ten seconds.

(b) * Jack filled carboys with wort in ten minutes.

(c) * Jack ran along the river in ten minutes.

However, as discussed by Moens and Steedman (1988), *in*-adverbials do cooccur with activity expressions if there is some contextually understood amount, quantity or distance; note that this is generally the case when we speak of routine activities (though some imagination is still required):

(2.100) (a) (This time) Jack poured wort into the carboy in ten seconds.

(b) (This time) Jack filled carboys with wort in ten minutes.

(c) (This time) Jack ran along the river in ten minutes.

[58] There is considerable disagreement in the literature concerning the nature of the achievement / accomplishment distinction. We will return to this point in chapter 4.

Finally, as mentioned in section 2.3 and (Moens and Steedman, 1988), *in*-adverbials also cooccur with achievement expressions under preparatory process readings:[59]

(2.101) (a) Jack won the race in thirty minutes.
(b) The wort reached the top in ten seconds.

2.6.3 *At*-Adverbials

At-adverbials cooccur with stative expressions and achievement expressions:

(2.102) (a) The carboy was full at that moment.
(b) Jack winked at that moment.

As noted by Steedman (1982), *at*-adverbials also cooccur with activity expressions under **inchoative** (onset) readings; though accomplishment expressions appear to behave similarly, note that (2.103b) is somewhat degraded:

(2.103) (a) Jack ran at that moment.
(b) ? Jack ran to the museum at that moment.

Stative expressions also cooccur with *at*-adverbials under inchoative readings; as Kent (1993) observes, these seem to be predominant with those expressions that Vendler deemed to have secondary achievement uses:

(2.104) (a) Jack understood the answer at that moment.
(b) Jack knew the results at that moment.

2.6.4 Present Tense

The present tense also cooccurs with stative and achievement expressions; however, as Verkuyl points out, the latter are restricted to reports of ongoing events:

(2.105) (a) The carboy is full.
(b) (Now) Jack winks.

Achievement expressions often cooccur with the present tense under habitual readings, which we will not discuss. Likewise, activity and accomplishment expressions usually cooccur with the present tense under habitual readings; however, these expressions have inchoative readings as well (in reports):

(2.106) (a) (Now) Jack runs.
(b) (Now) Jack runs to the museum.

Note that we will not discuss cases in which events are described outside the natural temporal order, e.g. historical present cases, exemplified cooking instructions, etc.

[59] The preparatory process reading induced by the *in*-adverbial should not be confused with the one where *in n minutes* is paraphraseable by *n minutes from now/then*; cf. (White, 1992) for further discussion of this distinction.

2.6.5 The Progressive

In section 2.2 we saw that the progressive cooccurs with activity and accomplishment expressions under **in-progress** readings. It does not generally cooccur with stative expressions, though there are exceptional readings we will not discuss. With achievement expressions, the progressive displays **futurate** readings:[60]

(2.107) (a) Jack was winning the race.

(b) The wort was reaching the top.

Note that the progressive can also cooccur with these expressions under iterative readings:

(2.108) (a) Jack was winning the race (each time).

(b) The wort was reaching the top (each time).

Finally, the in-progress and futurate readings of the progressive can lead to ambiguities with temporal adverbials:

(2.109) (a) Jack was running at noon (and had been for some time).

(b) Jack was running at noon (so he started loosening up).

2.6.6 Aspectual Verbs

Aspectual verbs are like the progressive in that they cooccur with activity and accomplishment expressions; as noted in section 2.1, they also have iterative readings. However, aspectual verbs do not seem to exhibit futurate readings:

(2.110) (a) ? Jack stopped winning the race.

(b) ? The wort stopped reaching the top.

Here the only available readings appear to be the iterative ones.

2.7 Semantic Desiderata

2.7.1 Downward Entailments

In section 2.1 we observed that activity expressions are downward entailing:

(2.111) (a) $\models \dfrac{\text{Jack poured wort into the carboy for ten seconds.}}{\text{Jack poured wort into the carboy for nine seconds.}}$

(b) $\models \dfrac{\text{Jack filled carboys with wort for ten minutes.}}{\text{Jack filled carboys with wort for nine minutes.}}$

[60] These readings do not appear to be as uniform as has been supposed; we will return to this point in chapter 4.

(c) $\models$ Jack ran along the river for ten minutes. / Jack ran along the river for nine minutes.

Recall that these entailments only hold down to some limit in size (the minimal parts problem); note that this problem does not arise with stative expressions, which are also downward entailing:

(2.112) (a) $\models$ The carboy was full for ten minutes. / The carboy was full for nine minutes.

(b) $\models$ Jack didn't fill the carboy for ten minutes. / Jack didn't fill the carboy for nine minutes.

(c) $\models$ Jack was filling the carboy for ten seconds. / Jack was filling the carboy for nine seconds.

Unlike activity expressions, accomplishment expressions (at least, the individuating ones) are not downward entailing:

(2.113) (a) $\not\models$ Jack poured twenty liters of wort into the carboy in ten seconds. / Jack poured twenty liters of wort into the carboy in nine seconds.

(b) $\not\models$ Jack filled twenty carboys with wort in ten minutes. / Jack filled twenty carboys with wort in nine minutes.

(b′) $\not\models$ Jack filled the carboy with wort in ten seconds. / Jack filled the carboy with wort in nine seconds.

(c) $\not\models$ Jack ran to the museum in ten minutes. / Jack ran to the museum in nine minutes.

(c′) $\not\models$ Jack ran two miles in ten minutes. / Jack ran two miles in nine minutes.

However, as discussed in section 2.4, some non-individuating accomplishment expressions do turn out downward entailing:

(2.114) (a) $\models$ Jack poured some amount of wort into the carboy in ten seconds. / Jack poured some amount of wort into the carboy in nine seconds.

(b) $\models$ Jack filled some carboys with wort in ten minutes. / Jack filled some carboys with wort in nine minutes.

(c) $\models$ Jack ran somewhere in ten minutes. / Jack ran somewhere in nine minutes.

(c′) $\models$ Jack ran some distance in ten minutes. / Jack ran some distance in nine minutes.

Finally, we may observe that while accomplishment expressions are not (normally) downward entailing temporally, they are downward entailing on the relevant quantity or distance:

(2.115) (a) $\models$ Jack poured twenty liters of wort into the carboy. / Jack poured ten liters of wort into the carboy.

(b) $\models$ Jack filled twenty carboys with wort. / Jack filled ten carboys with wort.

(c′) $\models$ Jack ran two miles. / Jack ran one mile.

2.7.2 Existential Entailments

The relation between activity expressions and the accomplishment expressions derived from them by adding amount or destination phrases is further illuminated by the following logical equivalences:[61]

(2.116) (a) $\models$ Jack poured wort into the carboy. / Jack poured some amount of wort into the carboy.

$\models$ Jack poured some amount of wort into the carboy. / Jack poured wort into the carboy.

(b) $\models$ Jack filled carboys with wort. / Jack filled some carboys with wort.

$\models$ Jack filled some carboys with wort. / Jack filled carboys with wort.

(c) $\models$ Jack ran. / Jack ran somewhere.

$\models$ Jack ran somewhere. / Jack ran.

(c′) $\models$ Jack ran. / Jack ran some distance.

$\models$ Jack ran some distance. / Jack ran.

2.7.3 The Imperfective Paradox

As we saw in section 2.2, the imperfective paradox distinguishes activity expressions from accomplishment ones:[62]

(2.117) (a) $\models$ Jack was pouring wort into the carboy. / Jack poured wort into the carboy.

(b) $\models$ Jack was filling carboys with wort. / Jack filled carboys with wort.

[61] In (2.116c), the first entailment must be subject to the caveat that Jack does not run in place.

[62] Note that the negative judgements no longer hold if the progressive is interpreted iteratively.

(c) $\models$ Jack was running along the river. / Jack ran along the river.

(2.118)

(a) $\not\models$ Jack was pouring twenty liters of wort into the carboy. / Jack poured twenty liters of wort into the carboy.

(b) $\not\models$ Jack was filling twenty carboys with wort. / Jack filled twenty carboys with wort.

(b′) $\not\models$ Jack was filling the carboy with wort. / Jack filled the carboy with wort.

(c) $\not\models$ Jack was running to the museum. / Jack ran to the museum.

(c′) $\not\models$ Jack was running two miles. / Jack ran two miles.

Recall, however, that (2.117b) exhibits a peculiar case of the minimal parts problem, insofar as this entailment must be subject to the caveat that the time of evaluation is not in the middle of the first filling event.

2.7.4 Aspectual Verbs

The preceding examples of the imperfective paradox work equally well if the progressive is replaced by either of the aspectual verbs *start* or *stop*; I will consider just two if cases here:

(2.119)

(a) $\not\models$ Jack stopped pouring twenty liters of wort into the carboy. / Jack poured twenty liters of wort into the carboy.

(b′) $\not\models$ Jack stopped filling the carboy with wort. / Jack filled the carboy with wort.

Naturally, these judgements switch if we replace *stop* by *finish*:

(2.120)

(a) $\models$ Jack finished pouring twenty liters of wort into the carboy. / Jack poured twenty liters of wort into the carboy.

(b′) $\models$ Jack finished filling the carboy with wort. / Jack filled the carboy with wort.

2.7.5 Distributive Temporal Adverbials

In section 2.1 we observed that temporal adverbials exhibit distributive readings. These give rise to the following entailments:

(2.121)

(a) $\models$ Swans glided past the dock in 30 seconds for 10 minutes. / A swan glided past the dock in 30 seconds.

(b) $\models$ Twenty swans glided along the shore for 30 seconds in 10 minutes. / A swan glided along the shore for 30 seconds.

Chapter 3

Preliminaries

In this chapter I present the technical preliminaries which underly the present approach. In the first two sections, I show how a classical categorial grammar may be used to translate English expressions into a typed, order-sorted translation language with a well-defined semantics. In the third section, I use this logic to formalize the present treatment of part structures, amounts, and times. This will then set the stage for the next chapter, where the core of the analysis is developed.

3.1 The Translation Language $\mathcal{L}_\sigma$

$\mathcal{L}_\sigma$ is a typed, order-sorted language whose formulation draws from (Partee, ter Meulen, and Wall, 1990; Gunter, 1992; Hedtstück and Schmitt, 1990; Walther, 1990). Since typed logics are well known in linguistics, this choice needs no explanation; order-sorted logics, being less well know, merit some discussion.

Many-sorted logic generalizes ordinary predicate logic by dividing up the domain of entities into various sorts. Strict many-sorted logic requires the sorts to form a partition; order-sorted logic relaxes this requirement, allowing the sorts to be partially-ordered.

Adding sorts to a logic has both semantic and syntactic ramifications. Semantically, sorts enable us to distinguish formulas that are merely false from those that are nonsensical, i.e. not **well-sorted**. For example, whereas (3.1a) may or may not be true in a particular model, (3.1b) does not make any sense whatsoever:

(3.1) (a) $\forall x^{\mathsf{line}}.\, \exists y^{\mathsf{line}}.\, \mathsf{parallel}(x, y)$

(b) $\forall x^{\mathsf{point}}.\, \exists y^{\mathsf{point}}.\, \mathsf{parallel}(x, y)$

Not surprisingly, formulas in order-sorted logic may be equivalently expressed in ordinary predicate logic using **sort axioms** and **relativizations**.

Sort axioms enforce the lexically specified sortal restrictions using unary **sort predicates**; relativizations are formulas where the sortal restrictions on bound variables become implications and conjunctions. For example, the relativization of (3.1a) appears below:

(3.2) $\forall x.\, \mathsf{line}(x) \rightarrow \exists y.\, \mathsf{line}(y) \wedge \mathsf{parallel}(x, y)$

Of course, such translations eliminate the ability to distinguish nonsensical terms, a distinction which is crucial to the present approach.

The use of sort axioms and relativizations has syntactic consequences as well. As can be seen from (3.2), relativization leads to longer formulas; moreover, sort axioms add to the available hypotheses. In practice, this results in longer and more complicated proofs than in the equivalent many-sorted logic, as Walther (1990) points out. Put the other way around, many-sorted logic enables shorter deductions with smaller formulas from smaller sets of hypotheses—which is clearly quite desirable from a computational standpoint.

Many-sorted logic is normally formulated as a generalization of first-order logic. In contrast, $\mathcal{L}_\sigma$ is a higher-order logic employing both sorts and types, where the sorts correspond to the partially-ordered **base types**. The difference between sorts and types in the present approach is best illustrated by examining the following two degenerate terms:

(3.3) (a) $x = 0 + \text{`}\mathsf{cosmo}\text{'}$
(b) $x = 0(0)$

Formula (3.3a) is not well-sorted, since addition is not defined for pairs of numbers and strings; in contrast, formula (3.3b) is not well-typed, since zero is of base type and thus cannot apply to itself.[1]

In many-sorted logic, a **signature** is generally employed to assign a unique sort to each argument position of a function symbol, as well as to its result. However, this is occasionally too restrictive. For example, suppose that we want to assign the appropriate sorts to the plus symbol, where we have the sorts natural number, integer, real number and complex number partially ordered as expected. Since any two numbers can be summed, we may assign + the following type:

(3.4) $+ : \mathbf{C} \times \mathbf{C} \rightarrow \mathbf{C}$

This type assignment lets + take as input pairs of natural numbers, pairs of reals, mixed pairs of natural numbers and integers, and so forth. Note, however, that regardless of the input types, + always returns a complex number. Clearly it would be preferable to let the sum of two natural numbers be a natural number, and so on; more generally, we would like the result type to be any sort which encompasses the two input sorts, as suggested below:

[1] Strictly speaking, sort errors should be thought of as type errors involving a clash of first-order types.

(3.5) $+ : \Pi\alpha^S.\, \alpha \times \alpha \rightarrow \alpha$, where $S = \{\mathbf{N}, \mathbf{Z}, \mathbf{R}, \mathbf{C}\}$

This type assignment involves a form of **parametric subtype polymorphism** which allows the result category to be more specific than would otherwise be possible. As we shall see in the next chapter, this form of polymorphism plays an important role in the present approach; unfortunately though, the desired polymorphism is non-trivial to define, since the partially-ordered sorts pose a significant challenge for the parametrization.

While devising an adequate treatment of this kind of polymorphism is an interesting problem, it is not particularly germane to the issues at hand. For this reason, I will let $\mathcal{L}_\sigma$ contain terms which are well-typed only up to sort compatibility. Of course, this means that we will lose the syntactic advantages of order-sorted logic pointed out by Walther; nevertheless, by employing appropriate sort axioms, we will retain the ability to semantically distinguish the well-sorted terms, as desired.

Before continuing, it is worth pointing out an analogy between this aspect of $\mathcal{L}_\sigma$ and the distinction between **static** (compile-time) and **dynamic** (run-time) type-checking in programming languages. While static type-checking is generally to be preferred, since it enables the early and efficient detection of a certain class of programming errors, it occasionally proves to be too restrictive; for this reason, increasing the flexibility of type-checking systems is the focus of much research. In the meanwhile, however, one is sometimes forced to resort to dynamic type-checking. Viewed in these terms, $\mathcal{L}_\sigma$ is a hybrid system employing static type-checking augmented with dynamic sort-checking, where the static component guarantees well-typed terms up to sort compatibility.

3.1.1 Syntax of $\mathcal{L}_\sigma$

We proceed now to the formal definition of the syntax of $\mathcal{L}_\sigma$, beginning with its signature and types:

(3.6) **Definition:** The **signature** of $\mathcal{L}_\sigma$ is a tuple $\langle(\mathsf{Sort}, \leq), \mathsf{Con}, \Sigma, \mathcal{S}\rangle$, where

(i) $(\mathsf{Sort}, \leq)$ is a lattice of **sorts** with top element $\top$ and bottom element $\bot$ (and not containing the special symbol t),

(ii) Con is an at most countably infinite set of non-logical constants,

(iii) $\Sigma : \mathsf{Con} \rightarrow \mathsf{Type}$ is a **lexical type assignment**, i.e. a function mapping each constant in Con to a type in Type, defined below, and

(iv) $\mathcal{S}$ is a set of sort axioms.

To specify the relationships between sorts, I will assume the following lattice-theoretic notions:

(3.7) **Definition:** Sorts s_1 and s_2 **overlap**, written $s_1 \circ s_2$, iff they have a common subsort s_3, i.e., iff there is some sort s_3 such that $s_3 \leq s_1$ and $s_3 \leq s_2$; sorts s_1 and s_2 are **disjoint**, written $s_1 \diamond s_2$, iff they do not overlap. The sort $s_1 \sqcap s_2$ is the **meet** of the sorts s_1 and s_2 if it is the greatest lower bound of s_1 and s_2 according to $\leq$; conversely, the sort $s_1 \sqcup s_2$ is the **join** of the sorts s_1 and s_2 if it is their least upper bound.

The types of $\mathcal{L}_\sigma$ are given below:

(3.8) **Definition:** The set Type of types T is defined according to the following grammar:

$$\begin{array}{lcl} T & ::= & \mathsf{e} \mid \mathsf{t} \\ T & ::= & T \times T \mid T \rightarrow T \end{array}$$

In order to keep disambiguating parentheses to a minimum, I will assume $\times$ and $\rightarrow$ to be right-associative and let $\times$ bind more tightly than $\rightarrow$.

Next we turn to the terms of $\mathcal{L}_\sigma$:

(3.9) **Definition:** Let Var be a countably infinite set of variables. The set of **raw terms** M is defined by the following grammar:

$$\begin{array}{l} x \,\varepsilon\, \mathsf{Var},\ c \,\varepsilon\, \mathsf{Con},\ s \,\varepsilon\, \mathsf{Sort},\ \tau \,\varepsilon\, \mathsf{Type} \\ \quad M \quad ::= \quad x \mid c \mid P_s \mid \mathsf{Defined} \\ \quad M \quad ::= \quad \lambda x^\tau.\, M \mid MM \mid \langle M, M \rangle \mid \forall x^\tau.\, M \mid \exists x^\tau.\, M \\ \quad M \quad ::= \quad \neg M \mid M \wedge M \mid M \vee M \mid M \rightarrow M \mid M \leftrightarrow M \mid M = M \end{array}$$

Note that $\mathcal{L}_\sigma$ includes a sortal predicate P_s for each sort s in Sort; I will often abbreviate this predicate as S. $\mathcal{L}_\sigma$ also includes a special predicate $\mathsf{Defined}$, which I will use to select the defined terms. To further minimize parentheses, I will assume application is left-associative and that application binds more tightly than abstraction and quantification. I will also assume that pairing is right-associative, abbreviating consecutive pairings as $x_1, \ldots, x_n$. Finally, I will often drop the type tags on bound variables, and occasionally add superfluous parentheses for clarity (especially in the case of function application).

(3.10) **Definition:** The set Term of **terms** is the set of raw terms M for which there is a **typing judgement** $H \vdash M : \tau$, as defined below.

(3.11) **Definition:** A **type assignment** H is a list $x_1 : \tau_1, \ldots, x_n : \tau_n$ such that the x_i are distinct. The type assigned to x_i by H is written $H(x_i)$. Note that the empty list will be abbreviated by simply omitting H.

(3.12) **Definition:** A **typing judgement** is a relation $\vdash$ between a type assignment H, a term M and a type τ. It is the least relation satisfying the rules shown in Figure 3.1. Comments on particular rules appear below:

- [Lookup] assigns non-logical constants their lexically specified type.
- [Sort] assigns sortal predicates the type $\mathsf{e} \rightarrow \mathsf{t}$, so that they may apply to any term of type e.
- [Defined] assigns the special predicate $\mathsf{Defined}$ the type $\tau \rightarrow \mathsf{t}$, so that it may apply to a term of any type.
- In the [Abstraction] and [Quantification] rules, the variable x must not appear in H, as otherwise $H, x : \rho$ would not be a well-formed type assignment.
- The additional cases of the rules [Quantification] and [Boolean] are omitted.

(3.13) **Remark:** I will assume the usual equational theory for the simply-typed λ-calculus,[2] suitably extended for the logical constructs.

Before continuing to the semantics of $\mathcal{L}_\sigma$ let us first consider a simple example of a **type derivation**. Let the signature of $\mathcal{L}_\sigma$ contain the following elements:

(i) the lattice $(\{\bot, \mathsf{Agent}, \mathsf{Object}, \mathsf{Event}, \top\}, \leq)$, where the sorts Agent, Object and Event are disjoint

(ii) the non-logical constants j, c and fill

(iii) the lexical type assignment

- $\Sigma(\mathsf{j}) = \Sigma(\mathsf{c}) = \mathsf{e}$
- $\Sigma(\mathsf{fill}) = \mathsf{e} \times \mathsf{e} \times \mathsf{e} \rightarrow \mathsf{t}$

(iv) the set $\mathcal{S}$ of sort axioms containing

- $\mathsf{Agent}(\mathsf{j})$
- $\mathsf{Object}(\mathsf{c})$
- $\forall x.\forall y.\forall e.\,[\mathsf{Defined}(\mathsf{fill}(x, y, e)) \leftrightarrow [\mathsf{Agent}(x) \wedge \mathsf{Object}(y) \wedge \mathsf{Event}(e)]]$

[2] Cf. (Gunter, 1992), for example.

[Projection]	$H, x : \tau, H' \vdash x : \tau$
[Lookup]	$H \vdash c : \Sigma(c)$
[Sort]	$H \vdash P_s : \mathsf{e} \rightarrow \mathsf{t}$
[Defined]	$H \vdash \mathsf{Defined} : \tau \rightarrow \mathsf{t}$
[Abstraction]	$\dfrac{H, x : \rho \vdash M : \tau}{H \vdash \lambda x^{\rho}.\, M : \rho \rightarrow \tau}$
[Application]	$\dfrac{H \vdash M : \rho \rightarrow \tau \qquad H \vdash N : \rho}{H \vdash M(N) : \tau}$
[Pairing]	$\dfrac{H \vdash M : \rho \qquad H \vdash N : \tau}{H \vdash \langle M, N \rangle : \rho \times \tau}$
[Quantification]	$\dfrac{H, x : \rho \vdash M : \mathsf{t}}{H \vdash \forall x^{\rho}.\, M : \mathsf{t}}$
[Negation]	$\dfrac{H \vdash M : \mathsf{t}}{H \vdash \neg M : \mathsf{t}}$
[Boolean]	$\dfrac{H \vdash M : \mathsf{t} \qquad H \vdash N : \mathsf{t}}{H \vdash M \wedge N : \mathsf{t}}$
[Equality]	$\dfrac{H \vdash M : \tau \qquad H \vdash N : \tau}{H \vdash M = N : \mathsf{t}}$

Figure 3.1: Typing rules for $\mathcal{L}_{\sigma}$.

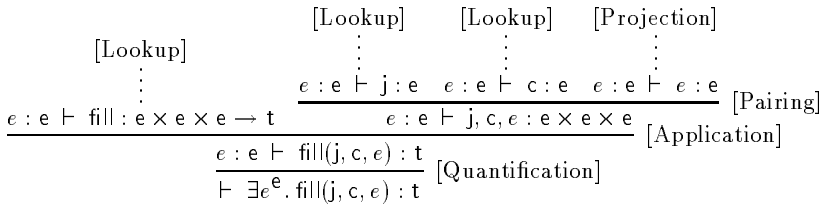


Figure 3.2: A simple type derivation.

Figure 3.2 shows how the typing judgement $\vdash \exists e^{\mathsf{e}}.\, \mathsf{fill}(\mathsf{j}, \mathsf{c}, e) : \mathsf{t}$ is derived. First the existential quantifier is removed using [Quantification], enabling the rule [Application] to apply; two applications of [Pairing] then isolates each argument term. The proof terminates with three applications of [Lookup] and one of [Projection].

Note that unlike the preceding term, $\mathsf{fill}(\mathsf{c}, \mathsf{j}, e)$ should not be well-sorted; however, as discussed above, nothing in the typing rules prevents this term from receiving a typing judgement. This is in contrast to the non- well-typed term $\mathsf{fill}(\mathsf{fill})$, which cannot be derived by the typing rules.

3.1.2 Semantics of $\mathcal{L}_\sigma$

The semantics of $\mathcal{L}_\sigma$ is largely straightforward. Naturally enough, the partial order on the sorts is interpreted as the subset relation on the domain of entities. The only complication is the presence of the sort axioms: if these are to have any content, we must allow functions to be partial—otherwise, predicates would be forced to be defined for all entities.

We begin with the models of $\mathcal{L}_\sigma$:

(3.14) **Definition:** A **model** $\mathcal{M}$ for $\mathcal{L}_\sigma$ is a pair $\langle (A_s)_{s \,\varepsilon\, \mathsf{Sort}}, I \rangle$, where $(A_s)_{s \,\varepsilon\, \mathsf{Sort}}$ is an indexed family of sets forming the **domain of entities** and I is the **lexical interpretation function**. The domain of entities is required to satisfy the following two conditions:

(i) $A_{s'} \subseteq A_s$ iff $s' \leq s$.

(ii) $A_\top$ is $\bigcup (A_s)_{s \,\varepsilon\, \mathsf{Sort}}$, and $A_\bot$ is $\emptyset$.

The lexical interpretation function I assigns each non-logical constant c in Con of type $\tau = \Sigma(c)$ an element of the set D_τ of possible denotations of type τ, defined below. Finally, the sort axioms $\mathcal{S}$ will be required to be true in $\mathcal{M}$, as defined below.

(3.15) **Definition:** The set D_τ of **possible denotations** of type τ is defined recursively as follows:

(i) $D_{\mathsf{e}} = A_\top$.

(ii) $D_{\mathsf{t}} = \{1, 0\}$.

(iii) $D_{\rho \times \tau} = D_\rho \times D_\tau$, the Cartesian product of D_ρ and D_τ.

(iv) $D_{\rho \to \tau}$ is the set of all partial functions $f : D_\rho \rightharpoonup D_\tau$.

To get a better feel for the possible denotations, consider the following example. Let the signature of $\mathcal{L}_\sigma$ contain the following elements:

(i) the lattice $(\{\bot, \mathsf{Dog}, \mathsf{Cat}, \top\}, \leq)$, where $\mathsf{Dog} \diamond \mathsf{Cat}$

(ii) the non-logical constants Astro, Cosmo and R

(iii) the lexical type assignment

- $\Sigma(\mathsf{Astro}) = \Sigma(\mathsf{Cosmo}) = \mathsf{e}$
- $\Sigma(\mathsf{R}) = \mathsf{e} \times \mathsf{e} \to \mathsf{t}$

(iv) the set $\mathcal{S}$ of sort axioms containing

- $\mathsf{Dog}(\mathsf{Astro})$
- $\mathsf{Cat}(\mathsf{Cosmo})$
- $\forall x. \forall y. [\mathsf{Defined}(\mathsf{R}(x, y)) \leftrightarrow [[\mathsf{Dog}(x) \wedge \mathsf{Dog}(y)] \vee [\mathsf{Cat}(x) \wedge \mathsf{Cat}(y)]]]$

Let $\mathcal{M}$ be the following model for $\mathcal{L}_\sigma$:

(i) The domain of entities consists of a set A_{Dog} of dogs and a set A_{Cat} of cats, as well as the empty set $A_\bot$ and the set of all entities $A_\top$. Note that the sets A_{Dog} and A_{Cat} must be disjoint to satisfy the partial order on the sorts.

(ii) The interpretation function I assigns Astro some particular dog, Cosmo some particular cat, and R the kinship relation between dogs and between cats, that is the union of the two total functions of type $A_{\mathsf{Dog}} \times A_{\mathsf{Dog}} \to \mathsf{t}$ and $A_{\mathsf{Cat}} \times A_{\mathsf{Cat}} \to \mathsf{t}$ which pick out the related dogs and the related cats, respectively.

Note that the union of the two functions making up R is only a partial function on $A_\top \times A_\top$, since $A_{\mathsf{Dog}} \times A_{\mathsf{Dog}} \cup A_{\mathsf{Cat}} \times A_{\mathsf{Cat}}$ is a proper subset of $A_\top \times A_\top$; in other words, $I(\mathsf{R})$ is not defined for the mixed pairs in $A_{\mathsf{Dog}} \times A_{\mathsf{Cat}} \cup A_{\mathsf{Cat}} \times A_{\mathsf{Dog}}$. For this reason, the non- well-sorted term $\mathsf{R}(\mathsf{Astro}, \mathsf{Cosmo})$ is meaningless in $\mathcal{M}$.[3]

[3]This intuitive notion of well-sortedness will be given a precise definition shortly.

We turn now to the formal definition of the meaning of a $\mathcal{L}_\sigma$-term in a model $\mathcal{M}$, which is given by the denotation partial function $[\![\cdot]\!]$. The definition of $[\![\cdot]\!]$ appears below, following two auxiliary definitions for variable management:

(3.16) **Definition:** Let H be a type assignment. An H-**environment** g is a function mapping each variable x_i in H to an element of $D_{H(x_i)}$.

(3.17) **Definition:** If g is an H-environment and d an element of $D_{H(x)}$, then $g[d/x]$ is the unique function f such that $f(x) = d$ and $f(y) = g(y)$ for y distinct from x.

(3.18) **Definition:** The **denotation** $[\![H \rhd M : \tau]\!]^{\mathcal{M},g}$ of a term M of type τ in H is defined relative to a model $\mathcal{M}$ and an H-environment g in Figures 3.3 and 3.4. Comments on particular cases appear below:

- (Defined) makes $\mathsf{Defined}(M)$ true of just those terms whose denotations are defined.
- (Sort) assigns P_s the predicate true of those entities which are in the subdomain A_s.
- (Abstraction) assigns the λ-term $\lambda x^\rho.\, M$ of type $\rho \to \tau$ in H the unique partial function f which yields the value $[\![H, x : \rho \rhd M : \tau]\!]^{\mathcal{M},g[d/x]}$ when applied to an element d of D_ρ, if such a value exists. If no such value exists for any d, then the λ-term is left undefined.
- (Application) leaves $H \rhd M(N) : \tau$ undefined if $[\![H \rhd N : \rho]\!]^{\mathcal{M},g}$ is not in the domain of the function $[\![H \rhd M : \rho \to \tau]\!]^{\mathcal{M},g}$.
- In (Application), (Pairing), (Negation), (Boolean), and (Equality), the denotation is left undefined if any of the subexpressions lack defined denotations.
- In (Quantification), the quantified term $Qx.\, \phi$ is left undefined iff ϕ is undefined for all possible substitutions for x.

The usual logical notions of model, consistency, validity and entailment can now be defined as follows:

(3.19) **Definition:** A **sentence** of $\mathcal{L}_\sigma$ is a term ϕ with typing judgement $\vdash \phi : \mathsf{t}$.

(3.20) **Definition:** A model $\mathcal{M}$ is a **model** of an $\mathcal{L}_\sigma$-sentence ϕ, or equivalently, ϕ is **true** in $\mathcal{M}$, iff $[\![\rhd \phi : \mathsf{t}]\!]^{\mathcal{M}} = 1$. This is symbolized $\mathcal{M} \models \phi$. If Γ is a set of $\mathcal{L}_\sigma$-sentences, then $\mathcal{M} \models \Gamma$ iff $\mathcal{M} \models \phi$ for each ϕ in Γ.

(3.21) **Definition:** An $\mathcal{L}_\sigma$-sentence ϕ is **consistent** iff it has a model.

(Projection)	$[\![H \triangleright x : \tau]\!]^{\mathcal{M},g} = g(x)$
(Lookup)	$[\![H \triangleright c : \tau]\!]^{\mathcal{M},g} = I(c)$
(Defined)	$[\![H \triangleright \mathsf{Defined}(M) : \mathsf{t}]\!]^{\mathcal{M},g} = 1$ iff $[\![H \triangleright M : \tau]\!]^{\mathcal{M},g}$ is defined; otherwise $[\![H \triangleright \mathsf{Defined}(M) : \mathsf{t}]\!]^{\mathcal{M},g} = 0$
(Sort)	$[\![H \triangleright P_s : \mathsf{e} \rightarrow \mathsf{t}]\!]^{\mathcal{M},g}$ is the function $f : D_{\mathsf{e}} \rightarrow D_{\mathsf{t}}$ such that $f(d) = 1$ iff $d \,\varepsilon\, A_s$
(Abstraction)	$[\![H \triangleright \lambda x^{\rho}.\, M : \rho \rightarrow \tau]\!]^{\mathcal{M},g}$ is that partial function $f : D_{\rho} \rightharpoonup D_{\tau}$ such that $f(d) = [\![H, x : \rho \triangleright M : \tau]\!]^{\mathcal{M},g[d/x]}$, if such a value exists; if no such f exists, $[\![H \triangleright \lambda x^{\rho}.\, M : \rho \rightarrow \tau]\!]^{\mathcal{M},g}$ is undefined
(Application)	$[\![H \triangleright M(N) : \tau]\!]^{\mathcal{M},g} = [\![H \triangleright M : \rho \rightarrow \tau]\!]^{\mathcal{M},g}([\![H \triangleright N : \rho]\!]^{\mathcal{M},g})$, if such a value exists; otherwise $[\![H \triangleright M(N) : \tau]\!]^{\mathcal{M},g}$ is undefined
(Pairing)	$[\![H \triangleright \langle M, N\rangle : \rho \times \tau]\!]^{\mathcal{M},g} = \langle [\![H \triangleright M : \rho]\!]^{\mathcal{M},g}, [\![H \triangleright N : \tau]\!]^{\mathcal{M},g}\rangle$, if such a value exists; otherwise $[\![H \triangleright \langle M, N\rangle : \rho \times \tau]\!]^{\mathcal{M},g}$ is undefined

Figure 3.3: Semantics of $\mathcal{L}_{\sigma}$, Part I.

(Quantification) (∀) $[\![H \triangleright \forall x^{\rho}.\, M : \mathsf{t}]\!]^{\mathcal{M},g}$ is undefined iff
for all $x \,\varepsilon\, D_{\rho}$, $[\![H, x : \rho \triangleright M : \tau]\!]^{\mathcal{M},g[d/x]}$ is undefined;
otherwise, $[\![H \triangleright \forall x^{\rho}.\, M : \mathsf{t}]\!]^{\mathcal{M},g} = 1$ iff
for no $x \,\varepsilon\, D_{\rho}$, $[\![H, x : \rho \triangleright M : \tau]\!]^{\mathcal{M},g[d/x]} = 0$,
and $[\![H \triangleright \forall x^{\rho}.\, M : \mathsf{t}]\!]^{\mathcal{M},g} = 0$ iff
for some $x \,\varepsilon\, D_{\rho}$, $[\![H, x : \rho \triangleright M : \tau]\!]^{\mathcal{M},g[d/x]} = 0$

(∃) $[\![H \triangleright \exists x^{\rho}.\, M : \mathsf{t}]\!]^{\mathcal{M},g}$ is analogous

(Negation) $[\![H \triangleright \neg M : \mathsf{t}]\!]^{\mathcal{M},g} = 1$ iff $[\![H \triangleright M : \mathsf{t}]\!]^{\mathcal{M},g} = 0$;
$[\![H \triangleright \neg M : \mathsf{t}]\!]^{\mathcal{M},g} = 0$ iff $[\![H \triangleright M : \mathsf{t}]\!]^{\mathcal{M},g} = 1$;
otherwise $[\![H \triangleright \neg M : \mathsf{t}]\!]^{\mathcal{M},g}$ is undefined

(Boolean) (∧) $[\![H \triangleright M \wedge N : \mathsf{t}]\!]^{\mathcal{M},g}$ is undefined iff
$[\![H \triangleright M : \mathsf{t}]\!]^{\mathcal{M},g}$ is undefined or $[\![H \triangleright N : \mathsf{t}]\!]^{\mathcal{M},g}$ is undefined;
otherwise, $[\![H \triangleright M \wedge N : \mathsf{t}]\!]^{\mathcal{M},g} = 1$ iff
$[\![H \triangleright M : \mathsf{t}]\!]^{\mathcal{M},g} = 1$ and $[\![H \triangleright N : \mathsf{t}]\!]^{\mathcal{M},g} = 1$,
and $[\![H \triangleright M \wedge N : \mathsf{t}]\!]^{\mathcal{M},g} = 0$ iff
$[\![H \triangleright M : \mathsf{t}]\!]^{\mathcal{M},g} = 0$ or $[\![H \triangleright N : \mathsf{t}]\!]^{\mathcal{M},g} = 0$

(∨) $[\![H \triangleright M \vee N : \mathsf{t}]\!]^{\mathcal{M},g}$ is analogous

(→) $[\![H \triangleright M \rightarrow N : \mathsf{t}]\!]^{\mathcal{M},g} = [\![H \triangleright \neg M \vee N : \mathsf{t}]\!]^{\mathcal{M},g}$

(↔) $[\![H \triangleright M \leftrightarrow N : \mathsf{t}]\!]^{\mathcal{M},g} = [\![H \triangleright (M \rightarrow N) \wedge (N \rightarrow M) : \mathsf{t}]\!]^{\mathcal{M},g}$

(Equality) $[\![H \triangleright M = N : \tau]\!]^{\mathcal{M},g} = 1$ iff
$[\![H \triangleright M : \tau]\!]^{\mathcal{M},g} = [\![H \triangleright N : \tau]\!]^{\mathcal{M},g}$;
$[\![H \triangleright M = N : \tau]\!]^{\mathcal{M},g} = 0$ iff
$[\![H \triangleright M : \tau]\!]^{\mathcal{M},g} \neq [\![H \triangleright N : \tau]\!]^{\mathcal{M},g}$;
otherwise $[\![H \triangleright M = N : \tau]\!]^{\mathcal{M},g}$ is undefined

Figure 3.4: Semantics of $\mathcal{L}_{\sigma}$, Part II.

(3.22) **Definition:** An $\mathcal{L}_\sigma$-sentence ϕ is **valid** iff it is true in every model. In symbols, this is written write $\models \phi$. A set Γ of $\mathcal{L}_\sigma$-sentences is valid iff every sentence ϕ in Γ is valid.

(3.23) **Definition:** A set Γ of $\mathcal{L}_\sigma$-sentences **entails** an $\mathcal{L}_\sigma$-sentence ϕ iff ϕ is true in every model in which Γ is true. Overloading the $\models$ symbol, this is also written $\Gamma \models \phi$.

Finally, well-sortedness can now be defined in terms of consistency as follows:

(3.24) **Definition:** A $\mathcal{L}_\sigma$-term M is **well-sorted** iff $\mathsf{Defined}(M)$ is consistent.

Note that a term M may be undefined in a particular model yet still be well-sorted, since well-sortedness is defined in terms of consistency; for a term M to be non- well-sorted, its denotation must be undefined in every model.

3.2 A Classical Categorial Grammar

For concreteness and ease of exposition I will make use of a classical categorial grammar, as defined below.

3.2.1 Formulation

The formulation of the grammar is based upon the one given in (Partee, ter Meulen, and Wall, 1990).[4]

(3.25) **Definition:** A **categorial grammar** $\mathcal{G}$ with target translations in $\mathcal{L}_\sigma$ is a triple $\langle \mathsf{Lex}, \mathsf{Alph}, F \rangle$ such that

(i) Lex is a finite set of **lexical items**,

(ii) Alph is a finite set of **basic category symbols**, and

(iii) F is a **lexical category assignment** function mapping each lexical item in Lex to a set of **category-meaning pairs** $X : M$, where X is in the set Cat of category symbols defined below and M is in the set of well-sorted $\mathcal{L}_\sigma$-terms.

(3.26) **Definition:** The set Cat of **category symbols** C is defined by the following grammar:

$$c \; \varepsilon \; \mathsf{Alph}$$
$$C \; ::= \; c \mid C/C \mid C \backslash C$$

Note that I will take both $/$ and $\backslash$ to be left-associative.

[4] Cf. also (Dowty, 1988) and (Steedman, 1991a) and references therein.

(3.27) **Definition:** The **one-step reduction** binary relation $\Longrightarrow$ is defined on sequences of category-meaning pairs as follows. For any categories X, Y in Cat, and any well-sorted $\mathcal{L}_\sigma$-terms M, N and $M(N)$,

(Forward Application) $X/Y : M \quad Y : N \quad \Longrightarrow \quad X : M(N)$,
and

(Backward Application) $Y : N \quad X \backslash Y : M \quad \Longrightarrow \quad X : M(N)$.

Note that we do not wish to limit ourselves to sequences of category-meaning pairs of length two. To that end, I will define $\Longrightarrow'$ to be the binary relation extending $\Longrightarrow$ to arbitrary length sequences of category-meaning pairs as follows: let α, β, η be any three category-meaning pairs, and let Γ, Δ be any two finite sequences of category-meaning pairs; then

$$\Gamma\ \alpha\ \beta\ \Delta \Longrightarrow' \Gamma\ \eta\ \Delta \quad \text{iff} \quad \alpha\ \beta \Longrightarrow \eta.$$

Using this extension, the **many-step reduction** binary relation $\overset{*}{\Longrightarrow}$ may be defined as the reflexive-transitive closure of $\Longrightarrow'$.

(3.28) **Definition:** A string $w_1 \ldots w_n$ in Lex^* is assigned a **reading** M by $\mathcal{G}$ iff

(i) there is some sequence of category-meaning pairs $\alpha_1 \ldots \alpha_n$ such that $F(w_i) = \alpha_i\ (1 \leq i \leq n)$, and

(ii) $\alpha_1 \ldots \alpha_n \overset{*}{\Longrightarrow} X : M$ for some category X.

In other words, $\mathcal{G}$ assigns $w_1 \ldots w_n$ a reading M iff there is some choice of lexical assignments α_i such that $\alpha_1 \ldots \alpha_n$ reduces to $X : M$ in zero or more steps. A **derivation** is a tree indicating the reduction steps. To symbolize the set of readings assigned to a string w, I will write $\mathcal{G}(w)$.

Let us now examine a sample derivation. Consider the following signature:

(i) the lattice $(\{\bot, \mathsf{Agent}, \mathsf{Event}, \mathsf{Time}, \top\}, \leq)$, with Agent, Event and Time disjoint

(ii) the non-logical constants Jack, Cosmo, noon, see, and PAST

(iii) the lexical type assignment

- $\Sigma(\mathsf{Jack}) = \Sigma(\mathsf{Cosmo}) = \Sigma(\mathsf{noon}) = \mathsf{e}$,
- $\Sigma(\mathsf{see}) = \mathsf{e} \times \mathsf{e} \times \mathsf{e} \rightarrow \mathsf{t}$,
- $\Sigma(\mathsf{PAST}) = \mathsf{e} \rightarrow \mathsf{t}$

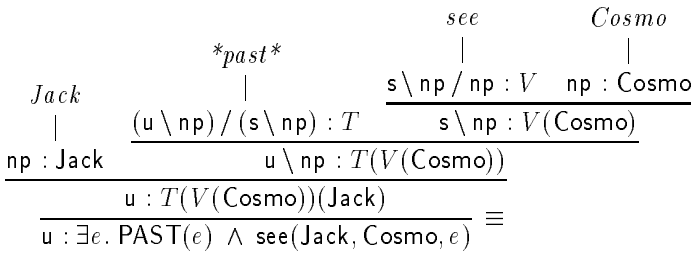


Figure 3.5: A categorial grammar derivation.

(iv) the set $\mathcal{S}$ of sort axioms containing

- $\mathsf{Agent}(\mathsf{Jack}) \wedge \mathsf{Agent}(\mathsf{Cosmo})$
- $\mathsf{Time}(\mathsf{noon})$
- $\forall x.\forall y.\forall e.\,[\mathsf{Defined}(\mathsf{see}(x,y,e)) \leftrightarrow [\mathsf{Agent}(x) \wedge \mathsf{Agent}(y) \wedge \mathsf{Event}(e)]]$

Let $\mathcal{G}$ contain the following elements:

(i) the lexical items *Jack, Cosmo, noon, see* and **past**

(ii) the basic category symbols np, s and u

(iii) the lexical category assignment F such that

Jack	$\mapsto$	np	:	Jack	
Cosmo	$\mapsto$	np	:	Cosmo	
noon	$\mapsto$	np	:	noon	
see	$\mapsto$	s \ np / np	:	$\lambda y.\,\lambda x.\,\lambda e.\,\mathsf{see}(x,y,e)$	$(\equiv V)$
past	$\mapsto$	(u \ np) / (s \ np)	:	$\lambda P.\,\lambda x.\,\exists e.\,\mathsf{PAST}(e) \wedge P(x)(e)$	$(\equiv T)$

Figure 3.5 shows a derivation of the meaning assigned to the string *Jack saw Cosmo*, assuming that *saw* is divided into **past** plus *see*. Note that the last step just replaces the previous $\mathcal{L}_\sigma$-term with its β-reduced equivalent.

Now consider the string *Cosmo saw noon*. Unlike the previous one, this string fails to have a reading, since the $\mathcal{L}_\sigma$-term $\exists e.\mathsf{PAST}(e) \wedge \mathsf{see}(\mathsf{Cosmo}, \mathsf{noon}, e)$ is not well-sorted, as required in definition (3.27). This illustrates how semantic anomalies due to sortal clashes are formally captured in the present account.

3.2.2 Truth Revisited

Since $\mathcal{G}$ does not unambiguously assign readings to strings, it is not possible to extend our logical notions of model, consistency, validity and entailment directly; instead, we must limit ourself to particular readings:

(3.29) **Definition:** A string w **under reading** ϕ is a **sentence** iff ϕ is in $\mathcal{G}(w)$ and $\vdash \phi : \mathsf{t}$.

(3.30) **Definition:** A sentence w under reading ϕ is **true** in a model $\mathcal{M}$ iff $\mathcal{M} \models \phi$. If Γ is a set of sentences (w_i) under readings (ϕ_i), then $\mathcal{M} \models \Gamma$ iff $\mathcal{M} \models \phi_i$ for each ϕ_i in Γ.

Consistency, validity and entailment are defined similarly.

At this point it is worth noting that in developing our fragment of English we will not be interested in all possible models of $\mathcal{L}_\sigma$; instead, we will prefer to focus on a restricted class of **permissible models** which are 'natural' in some sense. To specify this class, we will simply identify the non-logical constants of particular interest and the set of axioms or **meaning postulates** $\mathcal{A}$ that these must satisfy. Formally, this leads us to relativize the definitions of the logical notions of validity and entailment as follows:

(3.31) **Definition:** A model $\mathcal{M}$ is a **permissible model**, or equivalently, an $\mathcal{A}$-**model**, iff $\mathcal{M} \models \mathcal{A}$.

(3.32) **Definition:** An $\mathcal{L}_\sigma$-sentence ϕ is $\mathcal{A}$-**valid** iff it is true in every $\mathcal{A}$-model. In symbols, we write $\models_{\mathcal{A}} \phi$. A set Γ of $\mathcal{L}_\sigma$-sentences is $\mathcal{A}$-valid iff every sentence ϕ in Γ is $\mathcal{A}$-valid.

(3.33) **Definition:** A set Γ of $\mathcal{L}_\sigma$-sentences $\mathcal{A}$-**entails** an $\mathcal{L}_\sigma$-sentence ϕ iff ϕ is true in every $\mathcal{A}$-model in which Γ is true. This is written $\Gamma \models_{\mathcal{A}} \phi$.

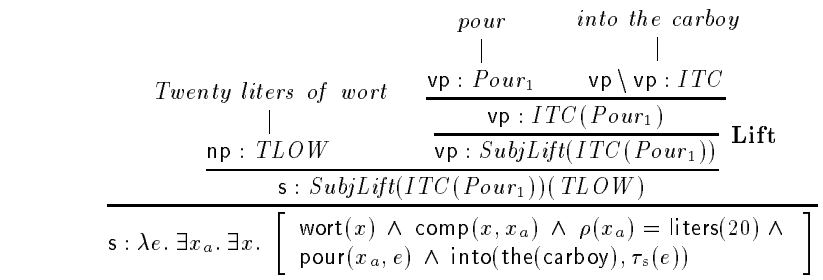
The natural extension of these notions to strings under readings is omitted. Having made these relativized definitions, I will henceforth shorten '$\mathcal{A}$-model' to 'model' and so on.

3.2.3 Sentence Radicals and Type-Lifting

To complete this section, I will briefly mention a couple of technical issues which arise in specifying a grammar for a fragment of English.

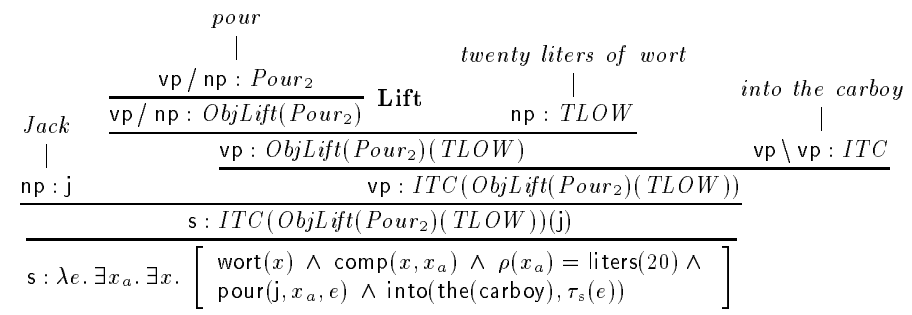
The first issue concerns how to handle eventuality variables, which I will assume appear in the translations of verbal expressions. Following Krifka (1989), I will translate untensed sentential expressions as eventuality predicates, called **sentence radicals**, and assume that tense and mood operators apply to sentence radicals to produce sentential expressions in $\mathcal{L}_\sigma$. Note that this has already been illustrated in Figure 3.5.

The second issue concerns how to manage the interaction of eventuality predicates with quantificational NPs. Because these NPs are of higher type than simple NPs, the grammar rules defined thus far will not allow them to combine with the λ-terms assigned to the verbs. To circumvent this problem, I will follow Landman (1991) in employing unary **type-lifting** operators, which



$$
\begin{array}{rcl}
TLOW & \equiv & \lambda P.\, \exists x_a.\, \exists x.\, [\mathsf{wort}(x) \wedge \mathsf{comp}(x, x_a) \wedge \rho(x_a) = \mathsf{liters}(20) \wedge P(x_a)] \\
Pour_1 & \equiv & \lambda x.\, \lambda e.\, \mathsf{pour}(x, e) \\
ITC & \equiv & \lambda P.\, \lambda x.\, \lambda e.\, [P(x)(e) \wedge \mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_s(e))] \\
SubjLift & \equiv & \lambda P.\, \lambda M.\, \lambda e.\, [M(\lambda x.\, Pxe)]
\end{array}
$$

Figure 3.6: Subject Type-Lifting.



$$
\begin{array}{rcl}
TLOW & \equiv & \lambda P.\, \exists x_a.\, \exists x.\, [\mathsf{wort}(x) \wedge \mathsf{comp}(x, x_a) \wedge \rho(x_a) = \mathsf{liters}(20) \wedge P(x_a)] \\
Pour_2 & \equiv & \lambda y.\, \lambda x.\, \lambda e.\, \mathsf{pour}(x, y, e) \\
ITC & \equiv & \lambda P.\, \lambda x.\, \lambda e.\, [P(x)(e) \wedge \mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_s(e))] \\
ObjLift & \equiv & \lambda Q.\, \lambda M.\, \lambda x.\, \lambda e.\, [M(\lambda y.\, Pyxe)]
\end{array}
$$

Figure 3.7: Object Type-Lifting.

we may add to the grammar as additional cases of the one-step reduction rule. Two such operators appear below:[5]

(3.34)
$$\begin{array}{rrcrcl} - & \mathsf{vp} & : & P & \Longrightarrow & \mathsf{vp} \quad : \quad \mathit{SubjLift}(P) \\ & \mathsf{vp}\,/\,\mathsf{np} & : & Q & \Longrightarrow & \mathsf{vp}\,/\,\mathsf{np} \quad : \quad \mathit{ObjLift}(Q) \end{array}$$

$$\begin{array}{ll} - & \mathit{SubjLift} \equiv \lambda P.\,\lambda M.\,\lambda e.\,[M(\lambda x.\,Pxe)] \\ - & \mathit{ObjLift} \equiv \lambda Q.\,\lambda M.\,\lambda x.\,\lambda e.\,[M(\lambda y.\,Qyxe)] \end{array}$$

These operators allow the λ-terms to combine in the desired way while maintaining the same syntactic structure.[6] To illustrate, Figures 3.6 and 3.7 show how these operators are used to derive the sentence radicals for *Twenty liters of wort poured into the carboy* and *Jack poured twenty liters of wort into the carboy*, respectively.

3.3 Using $\mathcal{L}_\sigma$

In this section I show how $\mathcal{L}_\sigma$ meaning postulates may be used to formalize the present treatment of part structures, amounts, and times.

3.3.1 Part Structures

To capture the desired part-of relations, I will employ the partial orders $\sqsubseteq_{\mathrm{i}}$, $\sqsubseteq_{\mathrm{q}}$ and $\trianglelefteq$. The relation $\sqsubseteq_{\mathrm{i}}$ corresponds to Link and Bach's partial order on plural individuals, and $\sqsubseteq_{\mathrm{q}}$ to their partial order on quantities. As Landman (1989a, p. 560) points out, these relations should not be conflated with the more involved (and more immediate) notion of part-of that holds between, say, Landman's hand and Landman himself: in the first case, the plural individuals include a singular subset, characterized by the lack of proper parts, which is clearly at odds with this latter notion; in the second case, the quantities are intended to be rather arbitrary portions of matter, which is again contrary to the latter notion's more structured quality. Interestingly, the distinction that Landman draws in (Landman, 1989a) seems to be very much the same as the one he draws in (Landman, 1992) between 'stages' of an event and 'parts' of an event—as noted in (cf. section 2.2.4), Landman (1992) finds it useful to distinguish the relation that holds between 'successive stages' of (say) Hanny Schaft's acts of resistance from the relation that holds between these acts and (say) World War II. Thus, with this parallel distinction in mind, I will let $\trianglelefteq$ symbolize this more complex structural relation on the various domains.

As we saw in section 2.1, both Hinrichs and Krifka follow Link and Bach in using their part-of relations to axiomatize complete join semilattices. It seems

[5] Here and in the sequel, I will often use vp as shorthand for $\mathsf{s} \setminus \mathsf{np}$.

[6] Note that additional operators are necessary for other syntactic positions; also, I will remain neutral as to whether similar operators should be used to account for scope ambiguity.

to me, however, that complete join semilattices do not accurately characterize the intended part structures—in some cases the axiomatization is too general, and in other cases it is too restrictive. Before considering this point further, though, let us review the appeal of these part structures in the first place.

Below we will see how the individual part-of relation $\sqsubseteq_{\mathrm{i}}$ can be used to effectively treat pluralities as reified sets. The attraction of this move is that it permits an intuitive and formally parallel treatment of the count and mass domains through a similar axiomatization of the quantity part-of relation $\sqsubseteq_{\mathrm{q}}$. The difference between the two is that $\sqsubseteq_{\mathrm{i}}$ is required to induce an **atomic** structure, i.e. one in which all elements contain an **atom**, or minimal element. Since this requirement does not hold of $\sqsubseteq_{\mathrm{q}}$, the intuition that portions of matter need not be divisible into minimal elements (unlike collections of individuals) is successfully captured.

At this point the technical questions arise. The first question is whether complete join semilattices are too general to capture the intended structures. In considering this question, Landman observes that in order to adequately formalize the notion of reified sets, the individual part-of relation $\sqsubseteq_{\mathrm{i}}$ should be required to form a **freely generated** complete join semilattice: otherwise, the semilattice could have unwanted contractions or 'mysterious' elements at the top—or in other words, sums not uniquely determined by their atoms. Interestingly, Landman also shows that similar concerns hold for the quantity part-of relation $\sqsubseteq_{\mathrm{q}}$.[7] Rather than pursue this line, however, I will simply focus on the constraints arising from the axiomatization of measure functions, and leave Landman's concerns for future work.

The second question, which neither Landman nor his predecessors consider, is whether complete join semilattices are too specific. In particular, we are entitled to wonder whether a natural join (or sum) operation can be defined in all the domains of interest. As we shall see shortly, defining joins for intervals of time is quite unnatural; moreover, in the next chapter we will find that a join operation cannot be defined for paths, conceived of as purely spatial entities. For this reason, I will not require $\sqsubseteq_{\mathrm{q}}$ to induce complete join semilattices below. Before doing so, however, let us reconsider part of the attraction that sums have held to this point.

In presenting his treatment of kinds, Hinrichs (1985, pp. 80-81) suggests that rather than taking the stages realizing a kind to be closed under subparts, we should assume the weaker condition of closure under joins, in order to sidestep the minimal parts problem. As indicated above, I will not follow Hinrichs' lead here, since doing so will considerably simplify the ensuing formalization. Moreover, it is not entirely clear whether any accuracy is lost in this move. Recall that when the minimal parts problem was first introduced in section 2.1, we noted that the expression *Jack run along the river* seems to apply to sucessively smaller subevents, at least until we reach a lower limit in size—subevents of Jack

[7] In this case though, the elements should be thought of as generated by their proper parts.

lifting his leg seem too small to qualify. However, careful reflection upon our structural part-of relation $\trianglelefteq$ reveals that in some sense the divisive reference of *Jack run along the river* can be made to fail long beforehand; consider, for example, the subevent of Jack's hand moving back and forth as he runs along: clearly this event does not qualify, for it does not even pertain to all of Jack. This observation suggests that neither this subevent nor the one of Jack lifting his leg should be considered to be part-of the event of Jack running along the river according to quantity part-of relation $\sqsubseteq_{\mathrm{q}}$; indeed, if this relation is to be of any use at all in analyzing events, it must have some notion of homogeneity 'built-in.' Consequently, there appears to be no reason not to require the stronger condition of downward closure on $\sqsubseteq_{\mathrm{q}}$. Note that doing so does not completely eliminate the minimal parts problem, since the matter of determining how small of a temporal interval may be successfully applied still remains—Jack's running for ten minutes should entail his running for five minutes, but perhaps not his running for 100 milliseconds.

With these remarks in mind, let us now turn to the axiomatization of the part structures. First, there is the requirement that $\sqsubseteq_{\mathrm{i}}$, $\sqsubseteq_{\mathrm{q}}$ and $\trianglelefteq$ each form a **partial order**:[8]

(3.35) **Definition (Partial Order):** A relation $\preceq$ is a **partial order** iff

(Reflexivity)	$\forall x.\,[x \preceq x]$
(Transitivity)	$\forall x.\,\forall y.\,\forall z.\,[[x \preceq y \wedge y \preceq z] \rightarrow x \preceq z]$
(Antisymmetry)	$\forall x.\,\forall y.\,[[x \preceq y \wedge y \preceq x] \rightarrow x = y]$

In the next chapter we shall see how the structural part-of relation $\trianglelefteq$ plays an important part in the present account of the imperfective paradox as well as negation and quantification—but for the moment, we will suspend further discussion of $\trianglelefteq$ and continue with the individual and quantiy part-of relations $\sqsubseteq_{\mathrm{i}}$ and $\sqsubseteq_{\mathrm{q}}$.

To partially characterize the sortal requirements of $\sqsubseteq_{\mathrm{i}}$ and $\sqsubseteq_{\mathrm{q}}$,[9] I will employ the sorts Plural^{+} and Plural^{-}, as shown below:[10]

(3.36) (a) $\forall x.\,\forall y.\,[\mathsf{Defined}(x \sqsubseteq_{\mathrm{i}} y) \rightarrow \mathsf{Plural}^{+}(x, y)]$

(b) $\forall x.\,\forall y.\,[\mathsf{Defined}(x \sqsubseteq_{\mathrm{q}} y) \rightarrow \mathsf{Plural}^{-}(x, y)]$

(c) $\mathsf{Plural}^{-} \leq \mathsf{Plural}^{+}$

[8] This definition may be considered an axiom schema for each of these relations.

[9] Note that if these sort axioms were formulated using the biconditional, then this would force the part-of relations to be defined across the disjoint domains of eventualities, space, time, etc.

[10] Both here and in the sequel, I will use $S(x_1, \ldots, x_n)$ as shorthand for the conjunction of the sortal propositions $S(x_i)$.

Along the lines of Link and Bach, I will take the ordinary, 'singular' individuals (Plural^-), the domain of the $\sqsubseteq_{\mathrm{q}}$ relation, to be a subset of the pluralities (Plural^+), the domain of the $\sqsubseteq_{\mathrm{i}}$ relation, as shown in (3.36c). Note that this is a slight simplification of their approach, where the domain of the quantity part-of relation (the 'quantities') is assumed to be a subset of the domain of ordinary individuals; as mentioned in section 2.5, since this distinction plays no role in the present account, I will omit it here.

To formalize the notion of 'singular' individual, I will again follow Link and Bach in identifying these with the $\sqsubseteq_{\mathrm{i}}$-atoms, as shown below:

(3.37) (a) $\mathsf{Plural}^- \;=\; \mathsf{Atom}_{\mathrm{i}}$

(b) $\forall y.\,[\mathsf{Atom}(y) \;\leftrightarrow\; \forall x.\,[x \sqsubseteq y \;\rightarrow\; x = y]]$

(c) $\forall x.\,\forall y.\,[x \sqsubseteq_{\mathrm{a}} y \;\leftrightarrow\; [\mathsf{Atom}(x) \wedge x \sqsubseteq y]]$

Note that the definition of the sort Atom in (3.37b) is schematized over both part-of relations. For convenience, I have also defined the schematized atomic part-of relation $\sqsubseteq_{\mathrm{a}}$ at this point, in (3.37c).[11]

In the case of the individual part-of relation, the join operation $\sqcup_{\mathrm{i}}$ may be defined as follows:[12]

(3.38) (a) $\forall x.\,\forall y.\,[\mathsf{Defined}(x \sqcup_{\mathrm{i}} y) \;\leftrightarrow\; \mathsf{Defined}(x \sqsubseteq_{\mathrm{i}} y)]$

(b) $\forall x.\,\forall y.\,\forall z.\,[x \sqcup_{\mathrm{i}} y = z \;\leftrightarrow\; \mathsf{LUB}_{\mathrm{i}}(x, y, z)]$

(3.39) (a) $\forall x.\,\forall y.\,\forall z.\,[\mathsf{LUB}(x, y, z) \;\leftrightarrow\; \mathsf{LUB}(\lambda w.\,[w = x \vee w = y], z)]$

(b) $\forall P.\,\forall z.\,[\mathsf{LUB}(P, z) \;\leftrightarrow\; \mathsf{Least}(\lambda v.\,[\mathsf{UpperBound}(P, v)], z)]$

(c) $\forall Q.\,\forall z.\,[\mathsf{Least}(Q, z) \;\leftrightarrow\; [Q(z) \wedge \forall v.\,[Q(v) \;\rightarrow\; z \sqsubseteq v]]]$

(d) $\forall P.\,\forall v.\,[\mathsf{UpperBound}(P, v) \;\leftrightarrow\; \forall w.\,[P(w) \;\rightarrow\; w \sqsubseteq v]]$

For the reasons discussed above, I will not require the quantity join operation to be defined in all cases:

(3.40) (a) $\forall x.\,\forall y.\,[\mathsf{Defined}(x \sqcup_{\mathrm{q}} y) \;\rightarrow\; \mathsf{Defined}(x \sqsubseteq_{\mathrm{q}} y)]$

(b) $\forall x.\,\forall y.\,\forall z.\,[x \sqcup_{\mathrm{q}} y = z \;\leftrightarrow\; \mathsf{LUB}_{\mathrm{q}}(x, y, z)]$

Since quantity least upper bounds will not always be defined, the less restrictive condition of minimal upper bound will prove useful in formalizing amounts:

(3.41) (a) $\forall x.\,\forall y.\,\forall z.\,[\mathsf{MUB}(x, y, z) \;\leftrightarrow\; \mathsf{MUB}(\lambda w.\,[w = x \vee w = y], z)]$

(b) $\forall P.\,\forall z.\,[\mathsf{MUB}(P, z) \;\leftrightarrow\; \mathsf{Min}(\lambda v.\,[\mathsf{UpperBound}(P, v)], z)]$

[11] In general, I will not make explicit the obvious sortal requirments on such schematized symbols.

[12] Note that in addition to schematizing the symbols LUB, Least, and UpperBound, I have also overloaded the symbol LUB in (3.39), since it is used with two different arities.

(c) $\forall Q.\, \forall z.\, [\mathsf{Min}(Q, z) \leftrightarrow [Q(z) \wedge \forall v.\, [[Q(v) \wedge v \sqsubseteq z] \rightarrow v = z]]]$

Now, as Landman points out, since the individual and quantity part structures are supposed to consist of entities that can be referred to linguistically, it makes no sense for them to contain a bottom element that is part of all the rest; this leads us to the following schematized postulate:

(3.42) $\neg\exists x.\, \forall y.\, [x \sqsubseteq y]$

Finally, to complete our characterization of the partial orders $\sqsubseteq_{\mathrm{i}}$ and $\sqsubseteq_{\mathrm{q}}$, it remains only to define the schematized relations **overlap** ($\circ$) and **disjoint** ($\diamond$):

(3.43) (a) $\forall y.\, \forall z.\, [y \circ z \leftrightarrow \exists x.\, [x \sqsubseteq y \wedge x \sqsubseteq z]]$
(b) $\forall y.\, \forall z.\, [y \diamond z \leftrightarrow \neg[y \circ z]]$

3.3.2 Amounts

To further characterize our part structures, I will employ a straightforward treatment of **amounts**, conceived of as abstract entities preserving numeric addition.

Let $\preceq_{\mathrm{a}}$ and $+_{\mathrm{a}}$ be a linear order and sum operation on amounts, meeting the following sortal requirements:

(3.44) (a) $\forall \alpha_1.\, \forall \alpha_2.\, [\mathsf{Defined}(\alpha_1 \preceq_{\mathrm{a}} \alpha_2) \rightarrow \mathsf{Amount}(\alpha_1, \alpha_2)]$
(b) $\forall \alpha_1.\, \forall \alpha_2.\, [\mathsf{Defined}(\alpha_1 +_{\mathrm{a}} \alpha_2) \rightarrow \mathsf{Amount}(\alpha_1, \alpha_2, \alpha_1 +_{\mathrm{a}} \alpha_2)]$

To attach numbers to these abstract amounts, I will employ **unit measures** such as liters, kilometers, minutes, and so forth. I will assume these unit measures ν are injective functions defined over the positive reals:

(3.45) (a) $\forall m.\, [\mathsf{Defined}(\nu(m)) \leftrightarrow [\mathbf{R}(m) \wedge m > 0]]$
(b) $\forall m.\, [\mathsf{Defined}(\nu(m)) \rightarrow \mathsf{Amount}(\nu(m))]$
(c) $\forall m.\, \forall n.\, [\nu(m) = \nu(n) \rightarrow m = n]$

Naturally, these unit measures will be required to preserve the arithmetic counterparts of $\preceq_{\mathrm{a}}$ and $+_{\mathrm{a}}$:

(3.46) (a) $\forall m.\, \forall n.\, [0 < m \leq n \rightarrow \nu(m) \preceq_{\mathrm{a}} \nu(n)]$
(b) $\forall m.\, \forall n.\, [\nu(m + n) = \nu(m) +_{\mathrm{a}} \nu(n)]$

Next, let ρ be a measurement function mapping quantities to amounts:

(3.47) $\forall x.\, [\mathsf{Defined}(\rho(x)) \rightarrow \mathsf{Amount}(\rho(x))]$

The measurement function ρ will be required to preserve the part-of relation $\sqsubseteq_{\mathrm{q}}$:

(3.48) $\forall x.\, \forall y.\, [x \sqsubseteq_{\mathrm{q}} y \;\rightarrow\; \rho(x) \preceq_{\mathrm{a}} \rho(y)]$

This function will also be required to preserve sums of minimal upper bounds; note, however, that in order to avoid counting the same part twice, the quantities must overlap in no more than a point (or zero-dimensional entity, more generally):

(3.49) $$\forall x.\, \forall y.\, \forall z.\, \left[\begin{array}{l} \left[\begin{array}{l} \mathsf{MUB}_{\mathrm{q}}(x, y, z) \wedge \\ \forall w.\, \left[x \circ_{\mathrm{q}} w \circ_{\mathrm{q}} y \;\rightarrow\; \mathsf{Dimension}^{0}(w) \right] \end{array} \right] \rightarrow \\ \rho(x) +_{\mathrm{a}} \rho(y) \;=\; \rho(z) \end{array} \right]$$

The amount function ρ can be extended to non-overlapping pluralities similarly:

(3.50) $\forall x.\, \forall y.\, [x \diamond_{\mathrm{i}} y \;\rightarrow\; \rho(x) +_{\mathrm{a}} \rho(y) \;=\; \rho(x \sqcup_{\mathrm{i}} y)]$

Finally, let $|\cdot|$ be the cardinality function mapping pluralities to natural numbers:

(3.51) $\forall x.\, [\mathsf{Defined}(|\, x\,|) \;\rightarrow\; [\mathsf{Plural}^{+}(x) \;\wedge\; \mathbf{N}(|\, x\,|)]]$

The cardinality function will likewise be required to preserve sums of disjoint pluralities, where atoms are assigned a cardinality of one:

(3.52) (a) $\forall x.\, [\mathsf{Atom}_{\mathrm{i}}(x) \;\rightarrow |\, x\,| = 1]$

(b) $\forall x.\, \forall y.\, [x \diamond_{\mathrm{i}} y \;\rightarrow |\, x\,| + |\, y\,| = |\, x \sqcup_{\mathrm{i}} y\,|]$

3.3.3 Times

In axiomatizing time, I will assume that instants form a dense linear order, and construct intervals out of instants using the quantity part-of relation $\sqsubseteq_{\mathrm{q}}$; the individual part-of relation $\sqsubseteq_{\mathrm{i}}$ will then be used to construct collections of intervals.

To establish these structures, I will first introduce the sorts Time, Interval and Instant, as shown below:

(3.53) (a) $\mathsf{Time} \leq \mathsf{Plural}^{+}$

(b) $\mathsf{Interval} \;=\; \mathsf{Time} \sqcap \mathsf{Plural}^{-}$

(c) $\mathsf{Instant} \;=\; \mathsf{Interval} \sqcap \mathsf{Atom}_{\mathrm{q}}$

Note that the times form a subset of the pluralities, the intervals a subset of the times, and the instants a subset of the intervals; additionally, the intervals correspond to the singular times (by $\sqsubseteq_{\mathrm{i}}$), and the instants to the atomic intervals (by $\sqsubseteq_{\mathrm{q}}$).

First let us consider the instants. Let $\preceq_{\mathrm{t}}$ be the following dense linear:

(3.54) $\forall t_1.\, \forall t_2.\, [\mathsf{Defined}(t_1 \preceq_{\mathrm{t}} t_2) \;\leftrightarrow\; \mathsf{Instant}(t_1, t_2)]$

(3.55) (Reflexivity) $\forall t.\,[t \preceq_{\mathrm{t}} t]$

(Transitivity) $\forall t_1.\,\forall t_2.\,\forall t_3.\,[[t_1 \preceq_{\mathrm{t}} t_2 \wedge t_2 \preceq_{\mathrm{t}} t_3] \rightarrow t_1 \preceq_{\mathrm{t}} t_3]$

(Antisymmetry) $\forall t_1.\,\forall t_2.\,[[t_1 \preceq_{\mathrm{t}} t_2 \wedge t_2 \preceq_{\mathrm{t}} t_1] \rightarrow t_1 = t_2]$

(Comparability) $\forall t_1.\,\forall t_2.\,[t_1 \preceq_{\mathrm{t}} t_2 \vee t_2 \preceq_{\mathrm{t}} t_1]$

(Density) $\forall t_1.\,\forall t_2.\,[t_1 \prec_{\mathrm{t}} t_2 \rightarrow \exists t_3.\,[t_1 \prec_{\mathrm{t}} t_3 \wedge t_3 \prec_{\mathrm{t}} t_2]]$

Using the ordering $\preceq_{\mathrm{t}}$ and the atomic quantity part-of relation $\sqsubseteq_{\mathrm{aq}}$, the intervals can be defined as convex collections of instants, as shown below:

(3.56) (a) $\forall i.\,[\mathsf{Interval}(i) \leftrightarrow$

$$\forall t_1.\,\forall t_2.\,\forall t_3.\,\left[\left[\begin{array}{c} t_1 \sqsubseteq_{\mathrm{aq}} i \wedge t_3 \sqsubseteq_{\mathrm{aq}} i \\ \bigwedge \\ t_1 \preceq_{\mathrm{t}} t_2 \wedge t_2 \preceq_{\mathrm{t}} t_3 \end{array}\right] \rightarrow t_2 \sqsubseteq_{\mathrm{aq}} i\right]]$$

(b) $\forall i_1.\,\forall i_2.\,[\mathsf{Interval}(i_1, i_2) \rightarrow [i_1 = i_2 \leftrightarrow \forall t.\,[t \sqsubseteq_{\mathrm{aq}} i_1 \leftrightarrow t \sqsubseteq_{\mathrm{aq}} i_2]]]$

Naturally, the linear ordering $\preceq_{\mathrm{t}}$ on the instants can be used to define a partial ordering $\preceq_{\mathrm{i}}$ on the intervals as follows:

(3.57) $\forall i_1.\,\forall i_2.\,[\mathsf{Defined}(i_1 \preceq_{\mathrm{i}} i_2) \leftrightarrow \mathsf{Interval}(i_1, i_2)]$

(3.58) $\forall i_1.\,\forall i_2.\,[i_1 \preceq_{\mathrm{i}} i_2 \leftrightarrow \forall t_1.\,\forall t_2.\,[[t_1 \sqsubseteq_{\mathrm{aq}} i_1 \wedge t_2 \sqsubseteq_{\mathrm{aq}} i_2] \rightarrow t_1 \preceq_{\mathrm{t}} t_2]]$

At this point let us consider durations, i.e. the amounts assigned to temporal intervals:

(3.59) (a) $\mathsf{Duration} \leq \mathsf{Amount}$

(b) $\forall i.\,[\mathsf{Interval}(i) \rightarrow \mathsf{Duration}(\rho(i))]$

Since we are only considering non-zero amounts, it does not make sense for atomic intervals to have defined durations. Accordingly, I will introduce the disjoint sorts $\mathsf{Dimension}^0$ and $\mathsf{Dimension}^1$ to partition the intervals into the instants and the rest:

(3.60) (a) $\mathsf{Dimension}^0 \diamond \mathsf{Dimension}^1$

(b) $\mathsf{Instant} = \mathsf{Interval} \sqcap \mathsf{Dimension}^0$

(c) $\mathsf{Interval} = \mathsf{Interval} \sqcap (\mathsf{Dimension}^0 \sqcup \mathsf{Dimension}^1)$

Of the remaining intervals, I will only require the closed ones to have defined durations; to identify these, I will introduce a subsort ClosedInterval of Interval, as follows:

(3.61) (a) $\mathsf{ClosedInterval} \leq \mathsf{Interval}$

(b) $\forall i.\,[\mathsf{ClosedInterval}(i) \leftrightarrow [\mathsf{Defined}(\mathsf{min}(i)) \wedge \mathsf{Defined}(\mathsf{max}(i))]]$

$$\text{(c)}\ \forall i.\,\forall t_0.\left[\begin{array}{c}[\mathsf{Defined}(\mathsf{min}(i))\ \wedge\ \mathsf{min}(i) = t_0]\\ \longleftrightarrow\\ [t_0 \sqsubseteq_{\mathrm{aq}} i\ \wedge\ \forall t.\,[t \sqsubseteq_{\mathrm{aq}} i\ \rightarrow\ t_0 \preceq_{\mathrm{t}} t]]\end{array}\right]$$

$$\text{(d)}\ \forall i.\,\forall t_1.\left[\begin{array}{c}[\mathsf{Defined}(\mathsf{max}(i))\ \wedge\ \mathsf{max}(i) = t_1]\\ \longleftrightarrow\\ [t_1 \sqsubseteq_{\mathrm{aq}} i\ \wedge\ \forall t.\,[t \sqsubseteq_{\mathrm{aq}} i\ \rightarrow\ t \preceq_{\mathrm{t}} t_1]]\end{array}\right]$$

Given these additional sorts, the non-atomic closed intervals can be guaranteed to have defined durations as follows:

(3.62) $\forall i.\,[\mathsf{ClosedInterval} \sqcap \mathsf{Dimension}^1(i)\ \rightarrow\ \mathsf{Defined}(\rho(i))]$

Before continuing, let us pause to examine why requiring there to be a quantity join operation $\sqcup_{\mathrm{q}}$ defined over the times would significantly complicate the formalization. Consider first two intervals which overlap: in this case there is no problem defining their join to be their least upper bound, since this yields another interval. Now consider two disjoint intervals: in this case, the least collection of instants in one or the other interval is no longer an interval, since it is not convex; for this reason, defining their join in this way would force an unnatural expansion of the domain. Of course, one might consider defining their join to be the least interval including them both; however, this move is equally unattractive, because one would then have to relax the condition that the duration of the join equals the sum of the durations of the pair, due to the gap in the middle.

At this point one might wonder whether it would be worth expanding the domain of atomic times to include non-convex ones, despite the unnaturalness of this move, for the sake of generality.[13] For one, doing so might appear to simplify the calculation of the duration of a plurality of events: rather than taking the duration of a plurality to be the sum of the durations of each atom, one could use Link and Bach's 'materialization' function to map the plurality to the quantity join of its atoms, and then just take the duration of this atomic individual. Indeed, one might even think that this could help to analyze sentences like the following one:

(3.63) Twenty swans glided past the dock in five minutes.

If the swans proceed in succession, then it does not make a difference how the durations are calculated. But, suppose now that each swan does not wait for the preceding one to finish its journey before embarking; quite plausibly, the most natural reading in this case is where five minutes is the amount of time it takes for all of them to finish, as the hypothesized analysis would have it. However, it is also possible that the intended reading is the one specified by the phrase *a total of*: the swans could be part of a race, where each is individually

[13] Or, including these non-convex times to begin with, as is done in Krifka (1992).

timed and free to start at will; in this case, the hypothesized analysis would not work if some of the journeys overlap.

To capture these readings then, I will assume there are two options: one, to sum the durations of the atoms, and two, to take the duration of the smallest interval containing the atoms. Interestingly, note that in addition to capturing the above readings, the second option will also work in the case where there are gaps between the completion of one swan's journey and the onset of another's which need to be smoothed over.

Since carrying out this analysis will require the introduction of a **convex hull** function cvx defined over the times, I will do so here:

(3.64) (a) $\forall i.\,[\mathsf{Defined}(\mathsf{cvx}(i)) \leftrightarrow \mathsf{Time}(i)]$

(b) $\forall i.\,[\mathsf{LUB}_{\mathrm{q}}(\lambda i'.\,[i' \sqsubseteq_{\mathrm{ai}} i], \mathsf{cvx}(i))]$

Note that since $\mathsf{cvx}(i)$ is required to be the quantity least upper bound of the individual atoms of i, $\mathsf{cvx}(i)$ must be an interval, and thus convex.[14]

[14] Of course, to generalize the definition of the convex hull function, the topological ordering implicit here would have to be made explicit.

Chapter 4

The Approach

In this chapter I develop the core of the present analysis, building upon the technical apparatus set forth in the preceding one. In section 1, I discuss the ontological distinctions encoded in the sortal lattice. In sections 2, 3, and 4, I show how these ontological distinctions may be directly employed in an explanatory account of the problem of aspectual composition. In sections 5 and 6, I address the closely related imperfective paradox, and cursorily examine the further issues of tense, negation, and aspectual type coercion. In sections 7 and 8, I review the desiderata listed at the end of chapter 2. Finally, in section 9 I summarize the analysis, identifying its merits and its shortcomings.

4.1 Ontology

The sortal lattice I will employ is quite similar, at least superficially, to the ones proposed in (Eberle, 1990; Dale, 1992) and especially (Jackendoff, 1991). The top level of the sort hierarchy is shown in Figure 4.1. That these sorts partition the domain of entities may be encoded as follows:

(4.1) (a) $\top = \mathsf{Material} \sqcup \mathsf{Eventuality} \sqcup \mathsf{Space} \sqcup \mathsf{Time} \sqcup \mathsf{Abstract}$

(b) $\mathsf{Material} \diamond \mathsf{Eventuality},\ \mathsf{Material} \diamond \mathsf{Space},\ \ldots$

I will assume that the sorts $\mathsf{Plural}^{\pm}$ cut across the hierarchy, but that the individual and quantity part-of relations respect the top-level partition, at least:[1]

(4.2) (a) $\forall x.\forall y.[\mathsf{TopLevel}(x,y) \leftrightarrow \left[\begin{array}{c}\mathsf{Material}(x,y) \vee \mathsf{Eventuality}(x,y) \vee \\ \mathsf{Space}(x,y) \vee \mathsf{Time}(x,y) \vee \\ \mathsf{Abstract}(x,y)\end{array}\right]]$

[1] As in the preceding chapter, I will use $S(x_1,\ldots,x_n)$ as shorthand for the conjunction of the sortal propositions $S(x_i)$. Note, however, that $\mathsf{TopLevel}(x,y)$ should not be confused with $\top(x,y)$, which holds of all pairs of entities.

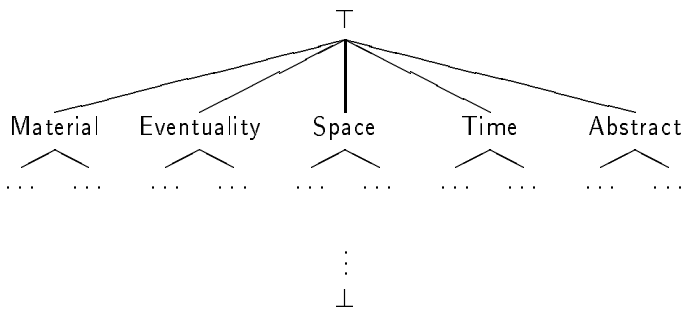


Figure 4.1: The top level of the sort hierarchy.

(b) $\forall x.\, \forall y.\, [\mathsf{Defined}(x \sqsubseteq y) \;\rightarrow\; \mathsf{TopLevel}(x, y)]$

I will also assume that the disjoint sorts $\mathsf{Delimited}^{\pm}$ cut across the hierarchy. These sorts correspond to the 'boundedness' feature [±b] in Jackendoff's system; the reasons behind the terminological change will be discussed in the next section. As shown below, I will require these sorts to partition the sorts Material, Eventuality, Space, and Time:

(4.3) $\mathsf{Delimited}^{+} \diamond \mathsf{Delimited}^{-}$

(4.4)
$$\begin{array}{lcll} \mathsf{Material} & = & \mathsf{Material} & \sqcap\ (\mathsf{Delimited}^{+} \sqcup \mathsf{Delimited}^{-}) \\ \mathsf{Eventuality} & = & \mathsf{Eventuality} & \sqcap\ (\mathsf{Delimited}^{+} \sqcup \mathsf{Delimited}^{-}) \\ \mathsf{Space} & = & \mathsf{Space} & \sqcap\ (\mathsf{Delimited}^{+} \sqcup \mathsf{Delimited}^{-}) \\ \mathsf{Time} & = & \mathsf{Time} & \sqcap\ (\mathsf{Delimited}^{+} \sqcup \mathsf{Delimited}^{-}) \end{array}$$

From this point onwards, I will focus on the singular entities in further developing the ontology. First, in the case of the material entities, the delimitedness distinction yields the following named subsorts:[2]

(4.5)
$$\begin{array}{lcl} \mathsf{Substance} & = & \mathsf{Material} \sqcap \mathsf{Delimited}^{-} \sqcap \mathsf{Plural}^{-} \\ \mathsf{Object} & = & \mathsf{Material} \sqcap \mathsf{Delimited}^{+} \sqcap \mathsf{Plural}^{-} \\ \mathsf{Agent} & = & \mathsf{Material} \sqcap \mathsf{Delimited}^{+} \sqcap \mathsf{Plural}^{-} \sqcap \mathsf{Agentive}^{+} \end{array}$$

Second, in the case of the spatial entities, our primary concern will be the formalization of paths; for this reason, I will treat the spatial entities analogously to the times:[3]

[2] The distinction between objects and agents is for mnemonic convenience only, i.e. I will have nothing to say about the sorts $\mathsf{Agentive}^{\pm}$.

[3] Note that places (or locations) are required to be points, i.e. zero-dimensional. To relax this requirement, one could assume an injection of the places into the atoms of the path structure; for simplicity though, I will not do so here.

(4.6) (a) $\mathsf{Path} = \mathsf{Space} \sqcap \mathsf{Plural}^-$

(b) $\mathsf{Place} = \mathsf{Path} \sqcap \mathsf{Atom}_{\mathrm{q}}$

(4.7) (a) $\mathsf{Place} = \mathsf{Path} \sqcap \mathsf{Dimension}^0$

(b) $\mathsf{Path} = \mathsf{Path} \sqcap (\mathsf{Dimension}^0 \sqcup \mathsf{Dimension}^1)$

Third, in the case of the eventualities, I will introduce the additional pair of partitioning sorts $\mathsf{Stative}^{\pm}$, to yield the following named subsorts:[4]

(4.8) (a) $\mathsf{Stative}^+ \diamond \mathsf{Stative}^-$

(b) $\mathsf{Eventuality} = \mathsf{Eventuality} \sqcap (\mathsf{Stative}^+ \sqcup \mathsf{Stative}^-)$

(4.9) (a)
$$\begin{array}{lcl} \mathsf{State} & = & \mathsf{Eventuality} \sqcap \mathsf{Stative}^+ \sqcap \mathsf{Plural}^- \\ \mathsf{NonState} & = & \mathsf{Eventuality} \sqcap \mathsf{Stative}^- \sqcap \mathsf{Plural}^- \end{array}$$

(b)
$$\begin{array}{lcl} \mathsf{Canonical} & = & \mathsf{State} \sqcap \mathsf{Delimited}^- \\ \mathsf{Moment} & = & \mathsf{State} \sqcap \mathsf{Dimension}^0 \sqcap \mathsf{Delimited}^+ \\ \mathsf{Period} & = & \mathsf{State} \sqcap \mathsf{Dimension}^1 \sqcap \mathsf{Delimited}^+ \end{array}$$

(c)
$$\begin{array}{lcl} \mathsf{Process} & = & \mathsf{NonState} \sqcap \mathsf{Delimited}^- \\ \mathsf{Event} & = & \mathsf{NonState} \sqcap \mathsf{Delimited}^+ \end{array}$$

(d)
$$\begin{array}{lcl} \mathsf{Momentaneous} & = & \mathsf{Event} \sqcap \mathsf{Dimension}^0 \\ \mathsf{Protracted} & = & \mathsf{Event} \sqcap \mathsf{Dimension}^1 \end{array}$$

Finally, since the times have already been introduced, it remains only to consider the abstract entities; the named subsorts of interest in this case are shown in Figure 4.2, along with the indicated sortal restrictions on the amount function ρ.

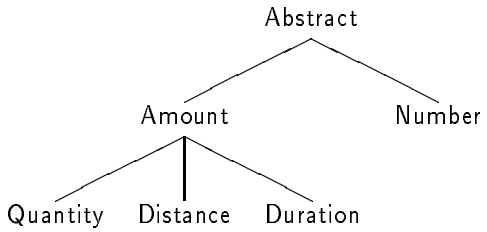
The sortal classification indicated above is quite rich; nonetheless, it is not meant to be exhaustive. There are of course related ontological distinctions in the literature, which are not the focus of attention here; these include, for example, the distinction between events which do and do not have inherent consequences,[5] and the distinction between groups and their members.[6] While the present framework is compatible with these distinctions, it does not depend on them.

To get a better feel for the classification, let us briefly compare it with some of its predecessors. The first point to make is that it posits sortal similarities that cut across the hierarchy of material entities, spatial entities, and so forth; in this regard it follows (Link, 1983; Hinrichs, 1985; Bach, 1986; Krifka, 1989; Eberle, 1990) and especially (Jackendoff, 1991). Focusing now on the eventualities, we

[4] As the reader may have noticed, the undelimited entities are never zero-dimensional, i.e. $\mathsf{Delimited}^- \diamond \mathsf{Dimension}^0$; the reason why this should be so will be explained in the next section.

[5] Cf. (Moens, 1987; Moens and Steedman, 1988).

[6] Cf. (Landman, 1989a; Landman, 1989b) and (Schwarzschild, 1992).



(a) $\forall x.\,[\mathsf{Material}(x) \rightarrow \mathsf{Quantity}(\rho(x))]$

(b) $\forall p.\,[\mathsf{Path}(p) \rightarrow \mathsf{Distance}(\rho(p))]$

(c) $\forall i.\,[\mathsf{Interval}(i) \rightarrow \mathsf{Duration}(\rho(i))]$

Figure 4.2: The subsorts of Abstract.

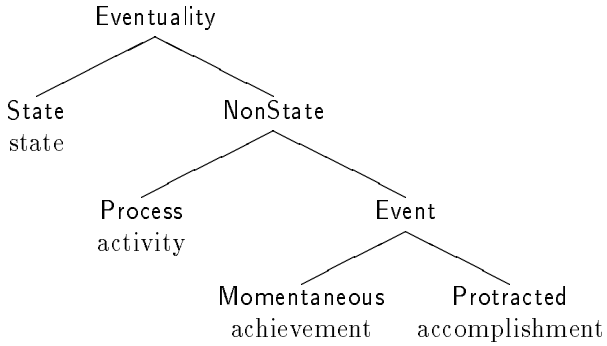


Figure 4.3: A comparison with Vendler's verbal classification.

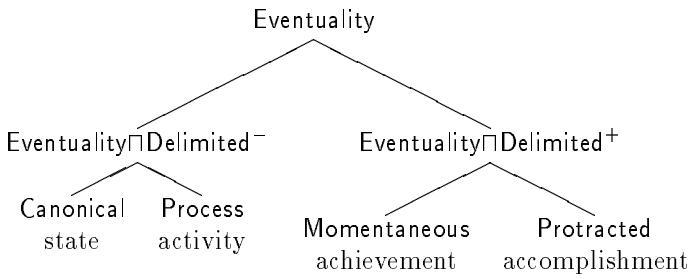
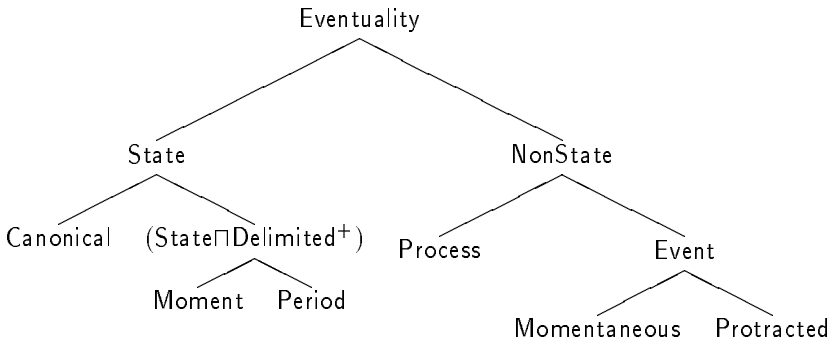


Figure 4.4: A second comparison with Vendler's verbal classification.

may observe that Vendler's verbal classification reappears in the sortal ordering as shown in Figure 4.3. Drawn in this way, it should be evident that the sortal classification incorporates the intuition that states are essentially different from the other eventualities, as in the classifications of (Bach, 1986), (Moens and Steedman, 1988), and their predecessors. However, the ordering shown in Figure 4.3 does not tell the whole story; in particular, one cannot tell whether states and activities form a coherent class, as they do in Vendler's classification. To explicitly indicate this aspect of the classification, we may redraw the sortal ordering as shown in Figure 4.4. To summarize, the orderings amongst the full set of named eventuality subsorts is suggestively drawn in Figure 4.5, with example (tenseless) expressions yielding predicates over the indicated sorts.

Of the sortal distinctions that play a role in the present analysis, there is one that warrants explicit discussion at this point. As Figure 4.3 indicates, Vendler's achievement/accomplishment distinction is treated here in terms of zero- or one-dimensionality. Interestingly, both Dowty (1986) and Verkuyl (1989) suggest that this aspect of Vendler's classification is not linguistically relevant; indeed, Verkuyl (p. 57) even goes back to Vendler's prototypical accomplishment expression *draw a circle* to point out that whether an event thus described is perceived as "momentary" or "protracted" may depend on whether one uses an old-fashioned protractor or a modern-day computer (where one might draw a circle with the touch of a button). However, what both Dowty and Verkuyl fail to point out is that giving up on this distinction means giving up on accounting for the remarkably different behavior that achievement and accomplishment expressions exhibit when combined with *at*-adverbials: as we saw in section 2.6, accomplishment expressions are somewhat degraded in combination with *at*-adverbials, in contrast to achievement expressions; moreover, to the extent that they do cooccur with *at*-adverbials, they do so only under inchoative readings, which achievement expressions seem to lack entirely.



Eventuality	Dimension^0 Delimited$^+$	Dimension^1 Delimited$^-$	Dimension^1 Delimited$^+$
Stative$^+$	Moment (a)	Canonical (b)	Period (c)
Stative$^-$	Momentaneous (d)	Process (e)	Protracted (f)

Example (tenseless) expressions:

(a) The carboy be full at 2:35 p.m

(b) The carboy be full

(c) The carboy be full for ten minutes

(d) Jack start running along the river at 2:35 p.m.

(e) Jack run along the river

(f) Jack run along the river for ten minutes

Figure 4.5: The named eventuality subsorts.

Ideally, one would like a system of aspectual categories to explain the behavior of the expressions of interest—here, the temporal adverbials and the phrases they modify. As we shall see in the ensuing sections, the distinctions of stativity, dimensionality and delimitedness illustrated in Figure 4.5 provide a basis for doing so, at least in part; in particular, we shall see that these cross-classifying distinctions enable a simple specification of the combinatory potential of the temporal adverbials *at*, *for*, and *in*—namely, that *at*-adverbials select for zero-dimensional eventualities, *for*-adverbials map undelimited eventualities to their delimited counterparts, and *in*-adverbials select for protracted events.

4.2 Delimitedness

In reviewing prior approaches to aspectual composition in (section 2.1), we saw that the accounts of Verkuyl and Jackendoff differed from those of Dowty, Hinrichs and Krifka in that the former ones employed the notions of 'specified quantity' and 'boundedness' (respectively) rather than appealing to the property of homogeneous reference; for this reason, their accounts did not appear to fall prey to the problem of non-individuating accomplishment expressions (section 2.4). However, as mentioned in sections 2.1.4 and 2.1.5, exactly how we are to interpret these two notions is unfortunately left rather vague. As a step towards improving upon their analyses then, I will develop in this section the related notion of **delimitedness**,[7] in a more fully formalized fashion.

After showing in section 2.4 that the problem of non-individuating accomplishment expressions was troublesome for the analyses of Dowty, Hinrichs and Krifka, I suggested in section 2.5 that this should really have come as no surprise: if we assume that the event/process distinction in the verbal domain is the analogue of the count/mass distinction in the nominal domain, and we know the count/mass distinction is subject to lexical idiosyncracy, then we should expect the event/process distinction to be subject to lexical idiosyncracy as well, at least indirectly; as such, the challenge then is to account for the behavior of complex verbal expressions making as few additional stipulations as possible.

In the present approach, the event/process distinction is encoded in terms of the partitioning sorts $\mathsf{Delimited}^{\pm}$, which likewise encode the object/substance distinction and analogous ones in the domains of space and time. At this point, the question arises as to how to interpret this cross-classifying distinction. As we saw in section 2.4, taking the relevant distinction between events and processes (and so on) to be the one that Link and Bach suggest does not seem to be of much help. For this reason, I suggested in section 2.5 that we try something along the lines of Carlson's (1977a) distinction between individuals (and/or stages) and kinds, as enriched by Hinrichs (1985); moreover, as our guide in doing so, I suggested we look to measure phrases.

[7] I have borrowed this term from (Mittwoch, 1982), where it is used informally.

As noted in sections 2.1.5 and 2.1.3, both Jackendoff and Krifka propose uniform analyses of nominal and verbal measure phrases.[8] In his discussion of the former ones, Krifka (1989, p. 82) states the following intuition:

> The measure phrase serves to 'cut out' entities of a certain size from a continuum of entities which fall under the head noun. If the head noun is quantized, then there is no such continuum, and the application of the measure phrase therefore should be infelicitous.

In section 2.5, I suggested that rather than following Krifka in formalizing this intuitive description of the role of measure phrases in terms of quantized modification, we could instead reconstruct the idea in a much more direct fashion, along the lines of the present conception of Jackendoff's approach. In present terms, the idea is simply to let the undelimited members of our ontological pairs—viz., the processes, substances, and so on—be the continua, and to let Jackendoff's 'composed-of' mapping do the 'cutting'—that is, the mapping to their delimited counterparts. As mentioned in section 2.5, and as we shall see below, this suggestion is tantamount to viewing Jackendoff's 'composed-of' mapping as an extended version of Carlson's 'realization' relation.

4.2.1 Homogeneous Predication

In further formalizing Carlson's ontology, Hinrichs (1985, pp. 80-81) suggests that we require the stages realizing a kind to be closed under joins; this he suggests is superior to taking them to be closed under subparts, which would seem to lead to difficulties with the minimal parts problem. In section 3.3.1, however, I argued that this latter, simpler option is in fact a viable one; for this reason, I will only require the composed-of mapping comp to be closed under subparts in reconstructing Hinrichs' idea here.

Roughly speaking, the relation comp should have the same sortal requirements as the quantity and individual part-of relations $\sqsubseteq_{\mathrm{q}}$ and $\sqsubseteq_{\mathrm{i}}$, except that comp maps undelimited entities to their delimited counterparts. As in the case of the part-of relations, I will not fully specify the sortal requirements on comp here, in order to remain non-committal on sortal issues that are not of present concern; I will, however, require comp to respect the top-level of the sort hierarchy, as well as plurality:

(4.10) (a) $$\forall x.\,\forall y.\,[\mathsf{Defined}(\mathsf{comp}(x,y)) \rightarrow \left[\begin{array}{c}\mathsf{Delimited}^{-}(x) \wedge \mathsf{Delimited}^{+}(y)\\ \bigwedge \\ \mathsf{TopLevel}(x,y)\end{array}\right]]$$

(b) $$\forall x.\,\forall y.\,[\mathsf{Defined}(\mathsf{comp}(x,y)) \rightarrow [\mathsf{Plural}^{-}(x) \rightarrow \mathsf{Plural}^{-}(y)]]$$

[8] As the reader may recall, this issue does not even arise for Hinrichs, since he does not examine nominal measure phrases.

To capture the intended connections to the part-of relations, I will also impose the following closure, identity and non-uniqueness[9] conditions:[10]

(4.11) (a) $\forall x.\,\forall x_a.\,\forall x_b.\,[\left[\begin{array}{l}\mathsf{Defined}(\mathsf{comp}(x,x_a))\wedge\\ \mathsf{Plural}^-(x)\wedge\\ \mathsf{Defined}(x_b\sqsubseteq_{\mathrm{q}}x_a)\end{array}\right]\rightarrow$
$\mathsf{Defined}(\mathsf{comp}(x,x_b))]$

(b) $\forall x.\,\forall x_a.\,\forall x_b.\,[\left[\begin{array}{l}\mathsf{Defined}(\mathsf{comp}(x,x_a))\wedge\\ \neg\mathsf{Plural}^-(x)\wedge\\ \mathsf{Defined}(x_b\sqsubseteq_{\mathrm{i}}x_a)\end{array}\right]\rightarrow$
$\mathsf{Defined}(\mathsf{comp}(x,x_b))]$

(c) $\forall x.\,\forall x_a.\,\forall x_b.\,[[\mathsf{comp}(x,x_a)\ \wedge\ x_b\sqsubseteq x_a]\ \rightarrow\ \mathsf{comp}(x,x_b)]$

(4.12) $\forall x.\forall y.[\mathsf{Delimited}^-(x,y)\ \rightarrow\ [x=y\ \leftrightarrow\ \forall z.[\mathsf{comp}(x,z)\ \leftrightarrow\ \mathsf{comp}(y,z)]]]$

(4.13) $\forall x.\,\forall x_a.\,[\mathsf{comp}(x,x_a)\ \rightarrow\ \exists x_b.\,[\mathsf{comp}(x,x_b)\ \wedge\ \neg[x_a=x_b]]]$

The relation comp has a natural quasi-inverse, namely the function which maps a delimited entity to the undelimited one containing it and all its subparts; we will name this function gr, after Jackendoff's 'ground-from' function:[11]

(4.14) (a) $\forall x.\,\forall x_a.\,[\mathsf{comp}(x,x_a)\ \rightarrow\ \mathsf{Defined}(\mathsf{gr}(x_a))]$

(b) $\forall x.\,[\mathsf{Defined}(\mathsf{gr}(x))\ \rightarrow\ \mathsf{Defined}(\mathsf{comp}(\mathsf{gr}(x),x))]$

(c) $\forall x.\,[\mathsf{Defined}(\mathsf{gr}(x))\ \rightarrow\ [\mathsf{Plural}^-(x)\ \leftrightarrow\ \mathsf{Plural}^-(\mathsf{gr}(x))]]$

(4.15) $\forall x_a.\,\forall x_b.\,[\mathsf{comp}(\mathsf{gr}(x_a),x_b)\ \leftrightarrow\ x_b\sqsubseteq x_a]$

To get a feel for the classification, let us now consider some examples. In the spirit of Pelletier and Schubert (1989), I will translate mass nouns such as *water* as predicates over both substances and objects (or equivalently, singular materials):[12]

(4.16) (a) $water\ \mapsto\ \mathsf{n}\ :\ \mathsf{water}$

(b) $\forall x.\,[\mathsf{Defined}(\mathsf{water}(x))\ \leftrightarrow\ \mathsf{Substance}\sqcup\mathsf{Object}(x)]$

(c) $\forall x.\,[\mathsf{Defined}(\mathsf{water}(x))\ \leftrightarrow\ \mathsf{Material}\sqcap\mathsf{Plural}^-(x)]$

[9] This requirement motivates the disjointness restriction on the undelimited and zero-dimensional entities, since the set of entities 'composed of' a certain undelimited entity cannot consist of just a single, atomic entity.

[10] Note that I will often use unsubscripted variables for undelimited entities, and subscripted variables for their delimited counterparts, when both appear in the same formula.

[11] As an alternative, one might consider instead letting gr map delimited entity predicates to undelimited entities; I will leave this issue open.

[12] Cf. (Pelletier and Schubert, 1989) for motivation, as well as an excellent review of prior treatments of mass expressions.

In contrast, I will translate count nouns such as *puddle* as predicates over objects only:

(4.17) (a) $\mathit{puddle} \mapsto \mathsf{n} : \mathsf{puddle}$
(b) $\forall x.\,[\mathsf{Defined}(\mathsf{puddle}(x)) \leftrightarrow \mathsf{Object}(x)]$

As Pelletier and Schubert note, some adjectives, such as *widespread*, do not make sense for ordinary objects; for expository purposes then,[13] I will translate *widespread* as follows:

(4.18) (a) $\mathit{widespread} \mapsto \mathsf{adj} : \mathsf{widespread}$
(b) $\forall x.\,[\mathsf{Defined}(\mathsf{widespread}(x)) \rightarrow \mathsf{Delimited}^{-}(x)]$

To illustrate the workings of the sort axioms, let us consider the following contrast:

(4.19) (a) This puddle is water.
(b) Water is widespread
(c) * This puddle is widespread.

The sort axioms given above are consistent with the acceptability of (4.19a-b) and the anomalous quality of (4.19c); to see why, observe that formulas (4.20a-b) below come out as theorems, whereas formula (4.20c) is not well-sorted (since $\mathsf{Object} \sqcap \mathsf{Delimited}^{-} = \bot$):

(4.20) (a) $\forall x.\,[[\mathsf{puddle}(x) \wedge \mathsf{water}(x)] \rightarrow \mathsf{Object}(x)]$
(b) $\forall x.\,[[\mathsf{water}(x) \wedge \mathsf{widespread}(x)] \rightarrow \mathsf{Substance}(x)]$
(c) * $\exists x.\,[\mathsf{puddle}(x) \wedge \mathsf{widespread}(x)]$

Up to this point, we have yet to address the interaction of predication and delimitedness. To do so, I will introduce the following notions:

(4.21) **Definition (Homogeneous Predication):**
$$\forall P.\,[\mathsf{Hom}(P) \leftrightarrow \left[\begin{array}{l} \forall x.\,\forall x_a.\,[[P(x) \wedge \mathsf{comp}(x, x_a)] \rightarrow P(x_a)] \wedge \\ \forall x_a.\,[P(x_a) \rightarrow P(\mathsf{gr}(x_a))] \end{array}\right]]$$

(4.22) **Definition (Divisive Reference):**
$$\forall P.\,[\mathsf{DIV}(P) \leftrightarrow \forall x_a.\,\forall x_b.\,[[P(x_a) \wedge x_b \sqsubseteq x_a] \rightarrow P(x_b)]]$$

Postulate (4.21) defines homogeneous predication in a novel way; according to this definition, a homogeneous predicate is one that is preserved by the mappings comp and gr. Postulate (4.22), on the other hand, schematically defines divisive reference in the usual way, i.e. a predicate refers divisively iff is is preserved by the part-of mappings $\sqsubseteq_{\mathrm{i}}$ or $\sqsubseteq_{\mathrm{q}}$.

The above definitions support the following schematized lemma:[14]

[13] Note that I will not address the temporal aspect of such predications.
[14] To be precise, $\mathsf{Hom}(P)$ entails either $\mathsf{DIV}_{\mathrm{i}}(P)$ or $\mathsf{DIV}_{\mathrm{q}}(P)$.

(4.23) **Lemma:** A homogeneous predicate refers divisively, i.e. (i) below is valid; however, a divisively referring predicate need not be a homogeneous one, i.e. (ii) below is not.

(i) $\forall P.\,[\mathsf{Hom}(P) \rightarrow \mathsf{DIV}(P)]$

(ii) $\forall P.\,[\mathsf{DIV}(P) \rightarrow \mathsf{Hom}(P)]$

Proof: In the first case, we have $\mathsf{comp}(\mathsf{gr}(x_a), x_b)$, by the definitions of comp and gr; since P satisfies Hom, P is preserved by both of these mappings, whence $P(x_b)$. In the second case, it suffices to note that divisively referring predicates, unlike homogeneous ones, may be undefined for all undelimited entities; if so, $P(\mathsf{gr}(x))$ will be undefined, and thus $\mathsf{Hom}(P)$ will be too.

Lemma (4.23) supports the present analysis of problematic count nouns such as *sequence* (cf. section 2.4). To illustrate, let us contrast its lexical requirements with those of the mass noun *wort*:

(4.24) (a) *sequence* $\mapsto$ n : sequence

(b) $\forall x.\,[\mathsf{Defined}(\mathsf{sequence}(x)) \leftrightarrow \mathsf{Object}(x)]$

(c) DIV(sequence)

(4.25) (a) *wort* $\mapsto$ n : wort

(b) $\forall x.\,[\mathsf{Defined}(\mathsf{wort}(x)) \leftrightarrow \mathsf{Substance} \sqcup \mathsf{Object}(x)]$

(c) Hom(wort)

By lemma (4.23), we may consistently require the translation of *sequence* to refer divisively, while still distinguishing it from count nouns such as *wort*; in other words, lemma (4.23) guarantees that the present framework has enough leeway to accommodate a lexically stipulated solution to this problem.

As might be expected, this same flexibility will support the present analysis of the problem of non-individuating accomplishment expressions. This idea is shown in caricature form below:

(4.26) (a) *Jack type a sequence* $\mapsto$ s : JTAS

(b) $\forall e.\,[\mathsf{Defined}(\mathsf{JTAS}(e)) \leftrightarrow \mathsf{Event}(e)]$

(c) DIV(JTAS)

(4.27) (a) *Jack type* $\mapsto$ s : JT

(b) $\forall e.\,[\mathsf{Defined}(\mathsf{JT}(e)) \leftrightarrow \mathsf{Process} \sqcup \mathsf{Event}(e)]$

(c) Hom(JT)

Of course, in the actual analysis, verbal expressions such as these will be translated as complex eventuality predicates; moreover, the sortal requirements of these predicates will not need to be stipulated, but will be made to follow from more basic assumptions. Nevertheless, this caricature should suffice to illustrate how a sortal approach could be used to resolve the problem of non-individuating accomplishment expressions; at the same time, it should also indicate the direction in which we are headed.

As a first step towards this analysis, consider the following lexical specifications for *fill* and *run*:

(4.28) (a) $\mathit{fill} \;\mapsto\; \mathsf{s} \setminus \mathsf{np} \,/\, \mathsf{np} \;:\; \lambda y.\, \lambda x.\, \lambda e.\, \mathsf{fill}(x, y, e)$

(b) $\forall x.\, \forall y.\, \forall e.\, [\mathsf{Defined}(\mathsf{fill}(x, y, e)) \rightarrow [\mathsf{Agent}(x) \wedge \mathsf{Object}(y) \wedge \mathsf{Protracted}(e)]]$

(4.29) (a) $\mathit{run} \;\mapsto\; \mathsf{s} \setminus \mathsf{np} \;:\; \lambda x.\, \lambda e.\, \mathsf{run}(x, e)$

(b) $\forall x.\, \forall e.\, [\mathsf{Defined}(\mathsf{run}(x, e)) \rightarrow [\mathsf{Agent}(x) \wedge \mathsf{Process} \sqcup \mathsf{Protracted}(e)]]$

(c) $\forall x.\, \mathsf{Hom}(\lambda e.\, \mathsf{run}(x, e))$

These examples show how the sortal requirements of simple state-change verbs such as *fill* can be distinguished from those of simple motion verbs such as *run*.

This ends our discussion of homogeneous predication for the moment, since I will delay the discussion of the more complex verbs, as well as plurals, until more of the present framework has been developed.

4.2.2 Measure Phrases

As mentioned at the beginning of this section, I will assume that measure phrases serve to 'cut out' a delimited entity of a certain size from an undelimited continuum of such entities. Technically, this assumption will lead us to include the composed-of mapping comp in the lexical specifications of the heads of measure phrases; in the case of ordinary restrictive modifiers, comp will of course left out.

Before turning to the formalization of this idea, let us consider why one would want to include the composed-of mapping in measure phrases in the first place. Since an undelimited entity is understood here as representing a continuum of entities of different sizes, there is no unique amount which can be sensibly assigned to it. For this reason, it makes sense to restrict the amount and cardinality functions to the delimited entities, as shown below:

(4.30) (a) $\forall x.\, [\mathsf{Defined}(\rho(x)) \leftrightarrow \mathsf{Delimited}^{+}(x)]$

(b) $\forall x.\, [\mathsf{Defined}(|\, x \,|) \leftrightarrow \mathsf{Delimited}^{+}(x)]$

Let us now compare and contrast the present treatment *for*- and *in*-adverbials. I will assume that the lexicon assigns both *for* and *in* functions from duration NPs (e.g. *ten minutes*) to VP-modifiers, as shown below:

(4.31)
$$\begin{array}{llrcl} \textit{for} & \mapsto & \mathsf{vp} \setminus \mathsf{vp} / \mathsf{np} & : & \textit{For} \\ \textit{in} & \mapsto & \mathsf{vp} \setminus \mathsf{vp} / \mathsf{np} & : & \textit{In} \\ \textit{ten} & \mapsto & \mathsf{nm} & : & 10 \\ \textit{minutes} & \mapsto & \mathsf{np} \setminus \mathsf{nm} & : & \mathsf{minutes} \end{array}$$

The terms *For* and *In* abbreviate the following complex terms:

(4.32)
$$\begin{array}{lcl} \textit{For} & \equiv & \lambda d.\, \lambda P.\, \lambda x.\, \lambda e_a.\, \exists e.\, [P(x)(e) \wedge \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = d] \\ \textit{In} & \equiv & \lambda d.\, \lambda P.\, \lambda x.\, \lambda e.\, [P(x)(e) \wedge \rho(\tau_{\mathrm{t}}(e)) \preceq_{\mathrm{a}} d] \end{array}$$

Unpacking the lambdas, (4.32) assigns to *in* a function which adds to the VP-meaning an upper bound d on the duration[15] $\rho(\tau_{\mathrm{t}}(e))$ of the event e; in contrast, as the head of a measure phrase, *for* is assigned a function which adds comp to the VP-meaning, so that the process e may be mapped to an event e_a of duration d.

To illustrate the effect of including comp in verbal measure phrases, consider the following tenseless expressions, together with their associated sentence radicals (cf. section 3.2):

(4.33) (a) Jack run for ten minutes

(b) $\lambda e_a.\, \exists e.\, [\mathsf{run}(\mathsf{j}, e) \wedge \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10)]$

(4.34) (a) * Jack fill the carboy for ten minutes

(b) *
$\lambda e_a.\exists e.[\mathsf{fill}(\mathsf{j}, \mathsf{the}(\mathsf{carboy}), e) \wedge \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10)]$

In example (4.33), the translation shown in (4.33b) is unproblematic, since run allows the eventuality e to be a process, as required by comp; in the anomalous example (4.34), on the other hand, the translation shown in (4.34b) is not well-sorted, since fill forces e to be an event.

Turning now to the nominal case, I will assume for the sake of simplicity that nominal measure phrases are headed by *of*, as shown below:

(4.35) (a)
$$\begin{array}{llrcl} \textit{twenty} & \mapsto & \mathsf{nm} & : & 20 \\ \textit{liters} & \mapsto & \mathsf{np} \setminus \mathsf{nm} & : & \mathsf{liters} \\ \textit{of} & \mapsto & \mathsf{np} / \mathsf{np} \setminus \mathsf{np} & : & \textit{Of} \end{array}$$

(b) $\textit{Of} \equiv \lambda q.\, \lambda x.\, \lambda P.\, \exists x_a.\, [P(x_a) \wedge \mathsf{comp}(x, x_a) \wedge \rho(x_a) = q]$

This lexical specification yields the following translation of the NP *twenty liters of wort*, assuming the appropriate type-lifting rules (again, cf. section 3.2):

(4.36) $\lambda P.\, \exists x_a.\, \exists x.\, [\mathsf{wort}(x) \wedge \mathsf{comp}(x, x_a) \wedge \rho(x_a) = \mathsf{liters}(20) \wedge P(x_a)]$

[15] Note that the duration of an event is equal to the amount of its associated temporal interval; this interval is the one returned by τ_{t}, the temporal trace function, which will be formalized in the next section.

Note that I have treated the bare NP *wort* as an indefinite expression here. As an alternative, one might also consider translating such NPs as names, following Carlson; to simplify matters, I will not do so here.

Before continuing on, it is worth pointing out that the measure phrases in the nominal domain again behave differently from restrictive modifiers, as shown below:

(4.37) (a) Ten feet of rope
(b) A ten foot rope
(c) A ten foot pole
(d) * Ten feet of (a) pole

In what follows, however, I will focus on the measure phrases, leaving their adjectival counterparts for another occasion.

4.2.3 A Note on Boundedness

At this point it should be evident that the present notion of delimitedness is quite different from the standard topological notion of boundedness. To clarify the difference, let us consider the case of intervals. Whereas an unbounded interval is one which has no lower or upper bounds, an undelimited interval represents a continuum of progressively larger delimited intervals; as such, nothing prevents an undelimited interval from having absolute limits. More precisely, given the definition of max-comp in (4.38), the existence of an undelimited entity x and a delimited entity x_a satisfying max-comp, as shown in (4.39), is perfectly consistent.

(4.38) $\forall x.\, \forall x_a.\, [\mathsf{max\text{-}comp}(x, x_a) \leftrightarrow [\mathsf{comp}(x, x_a) \wedge \forall x_b.\, [\mathsf{comp}(x, x_b) \rightarrow x_b \sqsubseteq x_a]]]$

(4.39) $\exists x.\, \exists x_a.\, [\mathsf{max\text{-}comp}(x, x_a)]$

As the reader may recall, in section 2.1.5 we observed that Jackendoff's notion of boundedness did not seem to fit with the standard topological notion, since his unbounded entities were allowed to have absolute limits; by conceiving of his distinction as one of delimitness, this particular problem with the notion can be resolved.

4.3 Space and Time

4.3.1 Spatio-Temporal Traces

In the spirit of (Krifka, 1989) and (Jackendoff, 1991), I will employ the following **spatio-temporal trace** functions τ to connect the eventuality domain to the spatial and temporal ones:

(4.40) (a) $\forall e.\,[\mathsf{Defined}(\tau_{\mathrm{s}}(e)) \rightarrow [\mathsf{Eventuality}(e) \wedge \mathsf{Space}(\tau_{\mathrm{s}}(e))]]$
(b) $\forall e.\,[\mathsf{Defined}(\tau_{\mathrm{t}}(e)) \rightarrow [\mathsf{Eventuality}(e) \wedge \mathsf{Time}(\tau_{\mathrm{t}}(e))]]$

As shown below, both traces will be required to preserve the part-of relations, as well as singularity:

(4.41) (a) $\forall e_a.\,\forall e_b.\,[e_b \sqsubseteq e_a \rightarrow \tau(e_b) \sqsubseteq \tau(e_a)]$
(b) $\forall e.\,[\mathsf{Plural}^{-}(e) \rightarrow \mathsf{Plural}^{-}(\tau(e))]$

Following Jackendoff, I will also assume that the spatio-temporal traces map moments and momentaneous events to places and instants; in the present framework, this is tantamount to requiring these functions to preserve zero-dimensionality:

(4.42) $\forall e.\,[\mathsf{Dimension}^{0}(e) \rightarrow \mathsf{Dimension}^{0}(\tau(e))]$

At an intuitive level, there is clearly a sense in which the eventuality domain is more tightly connected to the temporal domain than to the spatial domain. This intuition is reflected in the present treatment in two ways. First, only in the temporal case will I assume that traces are always defined:

(4.43) $\forall e.\,[\mathsf{Eventuality}(e) \rightarrow \mathsf{Defined}(\tau_{\mathrm{t}}(e))]$

Second, I will assume that the temporal trace function is an injective mapping over the quantity subparts of an eventuality:

(4.44) $\forall e_a.\,\forall e_b.\,[[e_b \sqsubseteq_{\mathrm{q}} e_a \wedge \tau_{\mathrm{t}}(e_b) = \tau_{\mathrm{t}}(e_a)] \rightarrow e_b = e_a]$

In contrast, I will not assume that the spatial trace function is injective in this way; this enables the spatial trace function to map states to places, as shown below:

(4.45) $\forall e.\,[\mathsf{State}(e) \rightarrow \mathsf{Place}(\tau_{\mathrm{s}}(e))]$

Note that since places are atomic, and since the spatial trace is required to preserve quantity part-of relation, this postulate requires the spatial location of states to be unchanging.

To aid in the formalization of paths, I will introduce the following (homogeneous) eventuality subsort:

(4.46) (a) $\mathsf{Locative} \leq \mathsf{Eventuality} \sqcap \mathsf{Plural}^{-}$
(b) $\mathsf{Hom}(\mathsf{Locative})$

The sort Locative identifies the singular eventualities which serve to keep track of the location of some object. As such, a locative eventuality could be a state, if the object is in one place, or an event, if it is moving. Naturally, I will require locative eventualities to always have defined spatial traces:

(4.47) $\forall e.\,[\mathsf{Locative}(e) \rightarrow \mathsf{Defined}(\tau_{\mathrm{s}}(e))]$

I will assume that locative events track continuous changes in location.[16] To ensure temporal continuity,[17] I will require the temporal trace function to be a bijective mapping over the quantity subparts of an eventuality; since the temporal trace function is already required to be injective in this way, it suffices to add the following surjectivity postulate:

(4.48) $\forall e_a.\,\forall i_b.\,[[\mathsf{Locative}(e_a) \wedge i_b \sqsubseteq_{\mathrm{q}} \tau_{\mathrm{t}}(e_a)] \rightarrow \exists e_b.\,[e_b \sqsubseteq_{\mathrm{q}} e_a \wedge \tau_{\mathrm{t}}(e_b) = i_b]]$

It is important to note that this bijectivity condition forces a locative event to have atomic elements, corresponding to the instants that are part of the event's temporal trace; as discussed in section 3.3.1, this condition is not one that will be required to hold for events in general. Following Jackendoff, I will assume that these atomic eventualities are moments (rather than momentaneous events):[18]

(4.49) $\forall e.\,[\mathsf{Locative} \sqcap \mathsf{Atomic}_{\mathrm{q}}(e) \rightarrow \mathsf{State}(e)]$

The idea here is simply to treat a locative event as a continuous sequence of moments which track the object's location; of course, a locative period will also consist of such a continuous sequence, the only difference being that the object's location must remain constant. Formally, the bijectivity condition on locative eventualities enables us to define following **time-slice** function κ:

(4.50) (a) $\forall e.\,\forall t.\,[[\mathsf{Locative}(e) \wedge \mathsf{Instant}(t) \wedge t \sqsubseteq_{\mathrm{aq}} \tau_{\mathrm{t}}(e)] \rightarrow \mathsf{Defined}(\kappa(e,t))]$

(b) $\forall e_a.\,\forall e_b.\,\forall t_b.\,[\kappa(e_a,t_b) = e_b \leftrightarrow [e_b \sqsubseteq_{\mathrm{aq}} e_a \wedge \tau_{\mathrm{t}}(e_b) = t_b]]$

To illustrate, Figure 4.6 shows how treating locative eventualities in this way can be seen to extend Hinrichs' (1985) notion of space-time 'sausages' by letting these consist of successive space-time 'slices' (cf. section 2.1.2).

4.3.2 Paths as Equivalence Classes

Paths are challenging to formalize. In the first place, it is not sufficient to let paths consist of an unordered collection of locations, for paths have distinguished origins and destinations. Moreover, one cannot simply rely upon an intrinsic ordering of locations, as one does with times, since the origin of one path may be the same as the destination of another, and vice-versa. Furthermore, it is

[16] Cf. Jackendoff (1993) for a somewhat different formalization of this idea, which is certainly implicit in (Jackendoff, 1991) and earlier works.

[17] Ensuring spatial continuity will be left for another occasion.

[18] Although I will not attempt to formalize the difference here, it is worth mentioning that momentaneous events, unlike moments, are supposed to mark second-order property changes, such as the onset of motion.

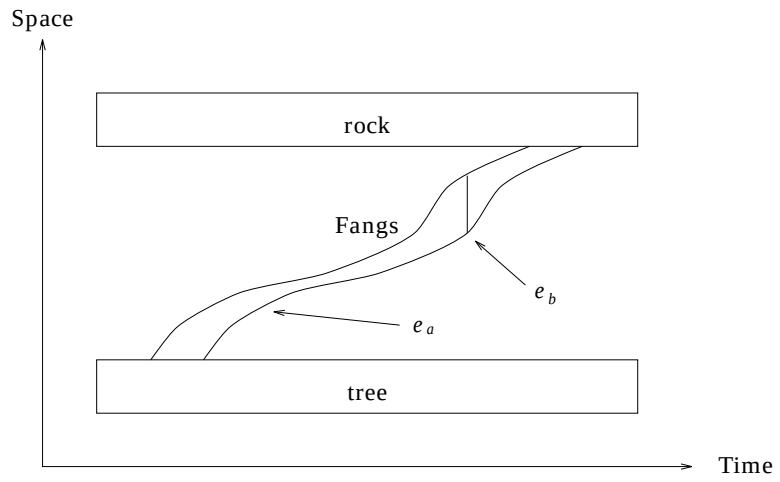


Figure 4.6: An event e_a of Fangs slithering from the tree to the rock (cf. Figure 2.1), together with a particular moment e_b of e_a.

not sufficient to let a path just consist of an ordered pair of locations, since this excludes paths with more complex shapes, such as figure-eights.

To account for this complexity, paths in mathematics are often taken to be functions from times (or some other linearly ordered set) to locations; note that Hinrichs' treatment of paths as chunks of space-time (cf. section 2.1.2) is quite similar to this idea. Interestingly, as Habel (1990) points out, there is a sense in which this treatment is too rich: if we model paths as functions from times to locations, then we distinguish two paths not only on the basis of distance, direction, and so forth, but also on the basis of duration, velocity, and other temporal properties—which clearly runs contra to the intuition that paths are purely spatial. To flatten these unwanted temporal distinctions, Habel suggests modeling paths as **equivalence classes** of such functions, where all the functions in an equivalence class are required to traverse the same locations in the same order.

Let us examine how Habel's notion of path can be reconstructed here. First, we should note that a locative eventuality e implicitly determines the following function from instants to locations, via the time-slice function κ and the spatial trace function τ_{s}:

(4.51) $\lambda t.\, \tau_{\mathrm{s}}(\kappa(e,t))$

Habel terms such functions **parametric paths**. If we were content with this notion of path, we might consider simply identifying the paths with the singular

locative eventualities:

(4.52) $\mathsf{Path} = \mathsf{Locative}$

Of course, this move is not consistent with our earlier assumption that the sorts Space and Eventuality are disjoint. To get around this problem, we could instead require the spatial trace function τ_{s} to form a bijection between these sorts:

(4.53) (a) $\forall p.\, [\mathsf{Path}(p) \rightarrow \exists e.\, [\mathsf{Locative}(e) \wedge \tau_{\mathrm{s}}(e) = p]]$
(b) $\forall e_1.\, \forall e_2.\, [[\mathsf{Locative}(e_1, e_2) \wedge \tau_{\mathrm{s}}(e_1) = \tau_{\mathrm{s}}(e_2)] \rightarrow e_1 = e_2]$

Now, while we could axiomatize paths in this way, doing so would not capture the notion of parametric path. The reason why is that this treatment of paths makes too many distinctions: if the paths are taken to be isomorphic to the locative eventualities, then irrelevant distinctions such as which object is in motion can force two paths to be different.

This suggests that we should relax the injectivity requirement on the spatial trace function, in order to flatten such unwanted distinctions. Note that this move is tantamount to treating paths as equivalence classes, as Habel would have us do, since each path can be viewed as a reified partition of the locative eventualities.

To capture the notion of parametric path then, we could impose the following condition on the equality of paths:

(4.54) $$\forall e_1.\, \forall e_2.\, \left[\begin{array}{l} \mathsf{Locative}(e_1, e_2) \rightarrow \\ \left[\begin{array}{l} \tau_{\mathrm{s}}(e_1) = \tau_{\mathrm{s}}(e_2) \leftrightarrow \end{array} \right. \left[\begin{array}{c} \tau_{\mathrm{t}}(e_1) = \tau_{\mathrm{t}}(e_2) \\ \bigwedge \\ \forall t.\, \left[\begin{array}{l} t \sqsubseteq_{\mathrm{aq}} \tau_{\mathrm{t}}(e_1) \rightarrow \\ \tau_{\mathrm{s}}(\kappa(e_1, t)) = \tau_{\mathrm{s}}(\kappa(e_2, t)) \end{array} \right] \end{array} \right] \end{array} \right]$$

This condition simply equates the paths associated with two locative eventualities iff they determine the same function from times to locations, as described above.

At this point it remains only to flatten the unwanted temporal distinctions in order to capture the desired path notion. To do so, we first need to consider the acceptable **parameter transformations**; along the lines of Habel's approach, I will map two locative eventualities to the same path if there is a bijective order homomorphism σ between their temporal traces which can appropriately adjust for their temporal differences, as shown below:

(4.55) **Definition:** $\mathsf{Hom}_{\preceq_{\mathrm{t}}}(\sigma, e_1, e_2)$ iff (a)-(d)

(a) $\forall t_1.\, [t_1 \sqsubseteq_{\mathrm{aq}} \tau_{\mathrm{t}}(e_1) \rightarrow \mathsf{Defined}(\sigma(t_1))]$
(b) $\forall t_2.\, [t_2 \sqsubseteq_{\mathrm{aq}} \tau_{\mathrm{t}}(e_2) \rightarrow \exists t_1.\, [t_1 \sqsubseteq_{\mathrm{aq}} \tau_{\mathrm{t}}(e_1) \wedge \sigma(t_1) = t_2]]$
(c) $\forall t_1.\, \forall t_2.\, [\sigma(t_1) = \sigma(t_2) \rightarrow t_1 = t_2]$
(d) $\forall t_1.\, \forall t_2.\, [t_1 \preceq_{\mathrm{t}} t_2 \rightarrow \sigma(t_1) \preceq_{\mathrm{t}} \sigma(t_2)]$

$$(4.56)\quad \forall e_1.\,\forall e_2.\left[\begin{array}{l}\mathsf{Locative}(e_1,e_2)\ \rightarrow\\ \left[\begin{array}{l}\\ \tau_{\mathrm{s}}(e_1)=\tau_{\mathrm{s}}(e_2)\ \leftrightarrow\ \exists\sigma.\left[\begin{array}{c}\mathsf{Hom}_{\preceq_{\mathrm{t}}}(\sigma,e_1,e_2)\\ \bigwedge\\ \forall t.\left[\begin{array}{l}t\sqsubseteq_{\mathrm{aq}}\tau_{\mathrm{t}}(e_1)\ \rightarrow\\ \tau_{\mathrm{s}}(\kappa(e_1,t))=\tau_{\mathrm{s}}(\kappa(e_2,\sigma(t)))\end{array}\right]\end{array}\right]\\ \\ \end{array}\right]\end{array}\right]$$

Given this treatment of paths, its associated quantity part-of relation can be defined derivatively, as follows:[19]

$$(4.57)\quad \forall e_1.\,\forall e_2.\left[\begin{array}{l}\mathsf{Locative}(e_1,e_2)\ \rightarrow\\ [\tau_{\mathrm{s}}(e_1)\sqsubseteq_{\mathrm{q}}\tau_{\mathrm{s}}(e_2)\ \leftrightarrow\ \exists e_2'.\,[e_2'\sqsubseteq_{\mathrm{q}}e_2\ \wedge\ \tau_{\mathrm{s}}(e_1)=\tau_{\mathrm{s}}(e_2')]]\end{array}\right]$$

As mentioned in section 4.3, we cannot define a join operation $\sqcup_{\mathrm{q}}$ on paths as conceived of here, since the least upper bound of two paths need not be unique; this point is illustrated in Figure 4.7.

4.3.3 Continua Continued

We have just seen how equivalence classes can be usefully employed in the path domain to flatten spurious, non-spatial distinctions. In this regard, the preceding discussion is somewhat reminiscent of our discussion of undelimited entities in section 4.2.2, where we noted that these entities cannot be sensibly assigned amounts. This suggests that we view the undelimited entities as representing equivalence classes which flatten information about quantity, distance, or duration, in the same way as an equivalence class approach to paths flattens temporal properties—in other words, whether we are dealing with a substance, a process, or one of their spatio-temporal or plural analogues, an undelimited entity should be seen as a continuum of delimited entities which are essentially the same.

To this point, we have yet to explicitly consider the undelimited paths; let us now examine to what extent the equivalence class view can inform our conceptualization of these entities. In Figure 4.8, a continuum of progressively larger paths is depicted, all of which go in the direction of X. By viewing this collection as an equivalence class p, we may abstract away from any particular distance, while at the same time preserving direction. This motivates letting a path predicate such as toward apply to both delimited and undelimited paths, unlike the measure function ρ. Note, however, that not all path predicates can be sensibly applied to undelimited paths, understood in this way; for example, if we assume that to encodes the end of a path, then we should restrict it to the delimited paths, since endpoints do not remain constant across a continuum of such entities. By and large, the same can be said for all path predicates involving endpoints or transition points, such as from, into, past, and so forth. On the other hand, other path predicates involving direction, such as away-from,

[19] Note that verifying the consistency of this definition requires ensuring that the choice of locative eventualities does not matter; I will omit these details here.

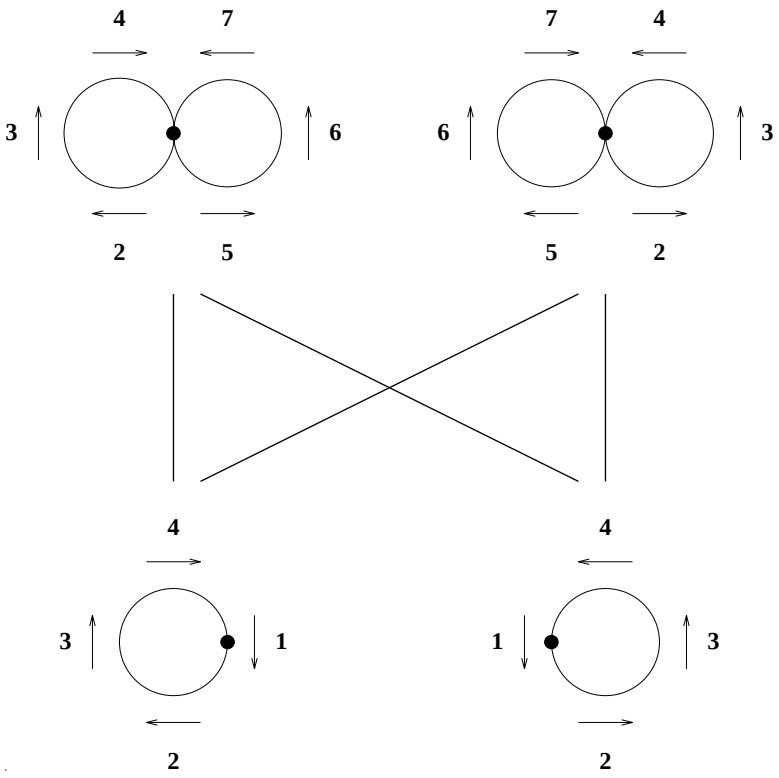


Figure 4.7: The two paths shown in the bottom portion of the figure lack unique least upper bounds, since both of the ones in the upper portion qualify as such.

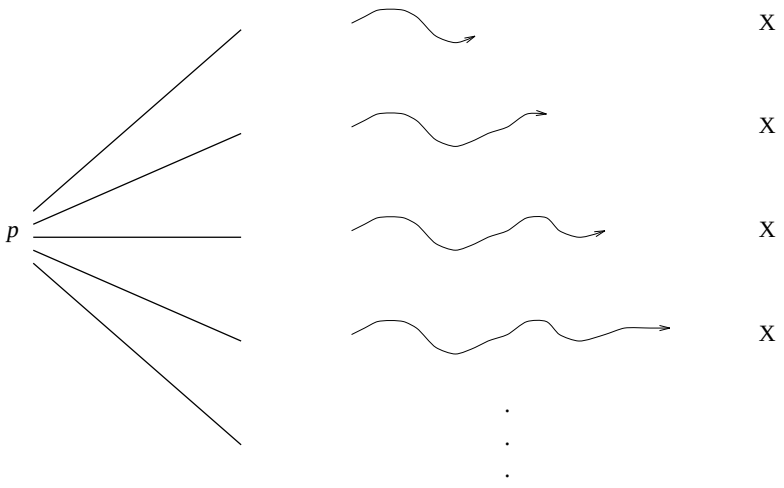


Figure 4.8: An equivalence class p of paths toward X.

northwards, and so on, need not be so restricted. A particularly interesting case is along: while *along the river* does seem to specify distance but not direction, insofar as one cannot run along the river if one is not near it, but one can run along the river either towards or away from the museum, it should be noted that this distance is not start-to-end distance but what Regier (1992) calls **proximal distance**, i.e. the distance from the located object to the closest point on the reference object;[20] of course, proximal distance, unlike start-to-end distance, can remain constant across a continuum of fixed length paths, and thus along can be sensibly predicated of both types of paths.

4.3.4 Extended Traces

To this point, we have implicitly assumed that the spatio-temporal traces preserve the $\mathsf{Delimited}^+$ property:

(4.58) $\forall e.\,[\mathsf{Delimited}^+(e) \rightarrow \mathsf{Delimited}^+(\tau(e))]$

To conclude this section then, it remains only to address the interaction of the spatio-temporal trace functions with processes and canonical states, i.e. the undelimited singular eventualities. Since a continuum of events or periods does not have a unique interval associated with it, I will assume the temporal trace of a process or canonical state likewise forms a continuum:

[20] Cf. (Herskovits, 1986).

(4.59) (a) $\forall e.\,[\mathsf{Process} \sqcup \mathsf{Canonical}(e) \;\rightarrow\; \mathsf{Delimited}^{-}(\tau_{\mathrm{t}}(e))]$

(b) $\forall e.\,\forall t_a.\,[\mathsf{comp}(\tau_{\mathrm{t}}(e), t_a) \;\leftrightarrow\; \exists e_a.\,[\mathsf{comp}(e, e_a) \wedge t_a \sqsubseteq_{\mathrm{q}} \tau_{\mathrm{t}}(e_a)]]$

The spatial case is somewhat more complicated. In the case of locative processes, I will similarly assume that the spatial trace forms a continuum in the path domain:

(4.60) (a) $\forall e.\,[\mathsf{Process} \sqcap \mathsf{Locative}(e) \;\rightarrow\; \mathsf{Delimited}^{-}(\tau_{\mathrm{s}}(e))]$

(b) $\forall e.\,\forall p_a.\,\left[\begin{array}{l}\mathsf{Locative}(e) \;\rightarrow \\ [\mathsf{comp}(\tau_{\mathrm{s}}(e), p_a) \;\leftrightarrow\; \exists e_a.\,[\mathsf{comp}(e, e_a) \wedge p_a \sqsubseteq_{\mathrm{q}} \tau_{\mathrm{s}}(e_a)]]\end{array}\right]$

In the case of canonical states, however, recall that we have assumed that location is unchanging; as such, it does not make sense for the spatial trace function to map such states to undelimited paths.[21] For this reason, I will instead assume that the spatial trace of a canonical state is a single, unchanging location:

(4.61) (a) $\forall e.\,[\mathsf{Canonical}(e) \;\rightarrow\; \mathsf{Delimited}^{+}(\tau_{\mathrm{s}}(e))]$

(b) $\forall e.\,\forall p.\,[\mathsf{Canonical}(e) \;\rightarrow\; [\tau_{\mathrm{s}}(e) = p \;\leftrightarrow\; \forall e_a.\,[\mathsf{comp}(e, e_a) \;\rightarrow\; \tau_{\mathrm{s}}(e_a) = p]]]$

4.4 Incremental Thematic Relations

4.4.1 Space and Time Linked

We have just seen that in the case of locative processes, the spatial trace function maps each eventuality continuum to a continuum in the path domain. Consequently, when the composed-of relation is invoked to 'cut out' an event from a process continuum, it concomitantly cuts out a delimited path from the spatial trace of the process. To help visualize this idea, Figure 4.9 shows how a process e with spatial trace p (from Figure 4.8) is mapped to an event e_a with spatial trace p_a.

Because locative eventualities link space and time in this way, temporal measure adverbials serve to indirectly determine distance; as such, we should also expect distance adverbials to determine duration. This mirror image case does indeed arise with distance measure adverbials such as *for two miles*, which have been largely ignored in the literature. Interestingly, distance measure adverbials and bare distance phrases (e.g. *two miles*) exhibit remarkably different cooccurrence patterns, as shown below:

(4.62) (a) Jack ran along the river for two miles.

(b) Jack ran two miles along the river.

[21] Formally, requiring the spatial trace function to map canonical states to undelimited paths in this way would violate the non-uniqueness condition on the composed-of relation.

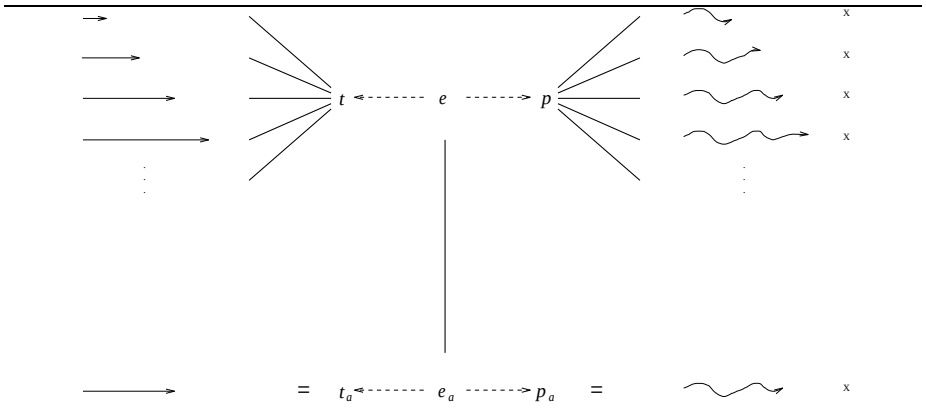


Figure 4.9: The mapping, via the composed-of relation comp, of a process e with spatial trace p and temporal trace t to an event e_a with spatial trace p_a and temporal trace t_a. The composed of relation is indicated by a solid line, and the spatio-temporal trace mappings by dotted lines.

(c) Jack ran two miles to the bridge.

(d) * Jack ran to the bridge for two miles.

Examples (4.62a) and (4.62b) show that bare distance phrases are virtually synonymous with their *for*-adverbial counterparts. This makes all the more striking the stark contrast between (4.62c) and (4.62d). As we shall see below, the present treatment of measure phrases suffices to account for these facts.[22]

To aid in analyzing these sentences, I will introduce the following eventuality subsort:

(4.63) (a) $\mathsf{DirectedMotion} \leq \mathsf{NonState} \sqcap \mathsf{Dimension}^1 \sqcap \mathsf{Plural}^-$

(b) $\mathsf{Hom}(\mathsf{DirectedMotion})$

The sort DirectedMotion is intended to classify the events and processes that may be described by the verbs *amble, bolt, carom, gallop, mosey, run, scoot, traipse, wade, zigzag, zoom* etc.[23] As in the case of the Locative eventualities,

[22]It should be mentioned at the outset that there is a similar contrast which will not be accounted for here, namely that between *Jack ran two miles along the river in ten minutes* and *? Jack ran along the river for two miles in ten minutes*—according to the present analysis, both should be acceptable. Interestingly though, note that the two sentence version of this example is much better: *Jack ran along the river for two miles. He did so in ten minutes.* On the other hand, the two sentence version of (4.62d) does not improve: *Jack ran to the bridge. * He did so for two miles.*

[23]Cf. Levin (1993, p. 106) for a discussion of this class.

I will require the directed motion eventualites to always have defined spatial traces:

(4.64) $\forall e.\,[\mathsf{DirectedMotion}(e) \rightarrow \mathsf{Defined}(\tau_{\mathrm{s}}(e))]$

I will likewise require the spatial trace function to map directed motion processes to path continua:

(4.65) (a) $\forall e.\,[\mathsf{Process} \sqcap \mathsf{DirectedMotion}(e) \rightarrow \mathsf{Delimited}^{-}(\tau_{\mathrm{s}}(e))]$

(b) $\forall e.\,\forall p_a.\,\left[\begin{array}{l}\mathsf{DirectedMotion}(e) \rightarrow \\ [\mathsf{comp}(\tau_{\mathrm{s}}(e), p_a) \leftrightarrow \exists e_a.\,[p_a \sqsubseteq_{\mathrm{q}} \tau_{\mathrm{s}}(e_a) \wedge \mathsf{comp}(e, e_a)]]\end{array}\right]$

Turning now to the syntax of these sentences, I will assume that bare distance NPs are optional complements to motion verbs, whereas locative PPs are adverbial modifiers:[24]

(4.66) (a)

Jack	$\mapsto$	np	: j
run	$\mapsto$	$\mathsf{s} \backslash \mathsf{np}$	: Run_1
run	$\mapsto$	$\mathsf{s} \backslash \mathsf{np} / \mathsf{np}$	: Run_2
along	$\mapsto$	$\mathsf{vp} \backslash \mathsf{vp} / \mathsf{np}$	: $Along$
the	$\mapsto$	np / n	: the
river	$\mapsto$	n	: river
for	$\mapsto$	$\mathsf{vp} \backslash \mathsf{vp} / \mathsf{np}$	: For_d
two	$\mapsto$	nm	: 2
miles	$\mapsto$	$\mathsf{np} \backslash \mathsf{nm}$	: miles

(b)

Run_1	$\equiv$	$\lambda x.\,\lambda e.\,\mathsf{run}(x, e)$
Run_2	$\equiv$	$\lambda d.\,\lambda x.\,\lambda e.\,[\mathsf{run}(x, e) \wedge \rho(\tau_{\mathrm{s}}(e)) = d]$
$Along$	$\equiv$	$\lambda y.\,\lambda P.\,\lambda x.\,\lambda e.\,[P(x)(e) \wedge \mathsf{along}(y, \tau_{\mathrm{s}}(e))]$
For_d	$\equiv$	$\lambda d.\,\lambda P.\,\lambda x.\,\lambda e_a.\,\exists e.\,[P(x)(e) \wedge \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{s}}(e_a)) = d]$

Before turning to the ungrammaticality of (4.62d), let us first examine what is needed to successfully derive (4.62a) - (4.62c). As shown above, the major difference between the lexical specifications of distance measure phrases and optional distance complements is the inclusion of the composed-of mapping comp in the former specification only. Because comp maps undelimited entities to their delimited counterparts, the $\mathcal{L}_\sigma$-terms derived for these phrases will differ in their sortal requirements: the former ones will select for processes, the latter ones for protracted events.

The lexical specifications in (4.66) yield the following sentence radicals for (4.62a) and (4.62b), respectively:

(4.67) (a) $\lambda e_a.\,\exists e.\,\left[\begin{array}{l}\mathsf{run}(\mathsf{j}, e) \wedge \mathsf{along}(\mathsf{the}(\mathsf{river}), \tau_{\mathrm{s}}(e)) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{s}}(e_a)) = \mathsf{miles}(2)\end{array}\right]$

[24] In a more sophisticated treatment, path PPs would be treated as optional complements too (but not exclusively). This would enable one to account for examples such as *Starting from the museum, Jack ran two miles past the bridge*, where two miles specifies the distance from the end of his run to the bridge rather than to the museum.

(b) $\lambda e.\left[\begin{array}{l}\mathsf{run}(\mathsf{j},e)\ \wedge\ \rho(\tau_{\mathrm{s}}(e))=\mathsf{miles}(2)\wedge\\ \mathsf{along}(\mathsf{the}(\mathsf{river}),\tau_{\mathrm{s}}(e))\end{array}\right]$

At first glance, it may appear that these dissimilar translations are incompatible with the virtual synonymy of (4.62a) and (4.62b); nevertheless, we will see shortly how the $\mathcal{L}_\sigma$-sentences formed from (4.67a) and (4.67b) can be made to be mutually entailing in a principled way. For the moment, however, we will concentrate on the differing sortal requirements which accompany these derivations.

Taking a closer look at the sentence radicals (4.67a) and (4.67b), we may observe in the first case that the presence of comp forces e to be a process; contrariwise, the presence of the amount function ρ in the second case forces e to be a protracted event. Because both run and along are non-commital on delimitedness, this is unproblematic:

(4.68) $\begin{array}{l}\forall x.\,\forall e.\,[\mathsf{Defined}(\mathsf{run}(x,e))\ \leftrightarrow\ [\mathsf{Agent}(x)\ \wedge\ \mathsf{DirectedMotion}(e)]]\\ \forall x.\,\forall p.\,[\mathsf{Defined}(\mathsf{along}(x,p))\ \leftrightarrow\ [\mathsf{Object}(x)\ \wedge\ \mathsf{Path}(p)]]\end{array}$

As mentioned in section 4.3.3, however, the path predicate to should be restricted to the delimited paths:

(4.69) $\forall x.\,\forall p.\,[\mathsf{Defined}(\mathsf{to}(x,p))\ \leftrightarrow\ [\mathsf{Object}(x)\ \wedge\ \mathsf{Path}\sqcap\mathsf{Delimited}^{+}(p)]]$

Let us now examine the effect of this sort axiom in detail.

The respective sentence radicals for (4.62c) and (4.62d) are shown below:

(4.70) (c) $\lambda e.\left[\begin{array}{l}\mathsf{run}(\mathsf{j},e)\ \wedge\ \rho(\tau_{\mathrm{s}}(e))=\mathsf{miles}(2)\wedge\\ \mathsf{to}(\mathsf{the}(\mathsf{bridge}),\tau_{\mathrm{s}}(e))\end{array}\right]$

(d) * $\lambda e_a.\,\exists e.\left[\begin{array}{l}\mathsf{run}(\mathsf{j},e)\ \wedge\ \mathsf{to}(\mathsf{the}(\mathsf{bridge}),\tau_{\mathrm{s}}(e))\wedge\\ \mathsf{comp}(e,e_a)\ \wedge\ \rho(\tau_{\mathrm{s}}(e_a))=\mathsf{miles}(2)\end{array}\right]$

Naturally, as was the case for (4.67b), in (4.70c) the amount function ρ forces e to be a protracted event, which means that its spatial trace $\tau_{\mathrm{s}}(e)$ turns out delimited, as required. In contrast, (4.70d) is not well-sorted, for the reasons explained below:

(4.71) **Theorem:** (4.70d) is not well-sorted.

Proof: Since to requires its path to be delimited, we know the spatial trace $\tau_{\mathrm{s}}(e)$ of e must be delimited; since the spatial trace function preserves delimitedness in the case of directed motion eventualities, e must therefore be an event. However, this is not consistent with comp, which requires e to be undelimited. Because there can be no eventuality e satisying the sort restrictions, (4.70d) is not well-sorted.

Having explained the anomaly in (4.62d), we return now to the synonymy of (4.62a) and (4.62b), repeated below:

(4.72) (a) Jack ran along the river for two miles.

(b) Jack ran two miles along the river.

As was suggested above, the $\mathcal{L}_\sigma$-sentences formed from their dissimilar sentence radicals can be made to be mutually entailing. To do so, we first need to apply existential closure:

(4.73) $Close \equiv \lambda P.\, \exists e.\, P(e)$

(4.74) (a) $Close(4.67a) \equiv \exists e_a.\, \exists e.\, \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e) \ \wedge \ \mathsf{along}(\mathsf{the}(\mathsf{river}), \tau_\mathrm{s}(e)) \wedge \\ \mathsf{comp}(e, e_a) \ \wedge \ \rho(\tau_\mathrm{s}(e_a)) = \mathsf{miles}(2) \end{array} \right]$

(b) $Close(4.67b) \equiv \exists e_a.\, \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e_a) \ \wedge \ \rho(\tau_\mathrm{s}(e_a)) = \mathsf{miles}(2) \wedge \\ \mathsf{along}(\mathsf{the}(\mathsf{river}), \tau_\mathrm{s}(e_a)) \end{array} \right]$

The choice of variables should make clear why (4.74a) and (4.74b) are not logically equivalent in any obvious way: in the former case, run appears with the process e, but not with the event e_a, whereas in the latter case, run appears with the event e_a, and there is no process e present at all; moreover, similar considerations hold for along.

To prove this equivalence then, we need to know that run and along form homogeneous predicates:

(4.75) (a) $\forall x.\, \mathsf{Hom}(\lambda e.\, \mathsf{run}(x, e))$

(b) $\forall x.\, \mathsf{Hom}(\lambda p.\, \mathsf{along}(x, p))$

Together these postulates suffice to prove the following theorem:

(4.76) **Theorem:** (4.74a) and (4.74b) are logically equivalent.

(a) (4.74a) $\models$ (4.74b).

Proof: Let e be a process and e_a an event which make (4.74a) true. The conjuncts in (4.74b) are then entailed as follows:

(i) Since $\lambda e.\, \mathsf{run}(\mathsf{j}, e)$ is a homogeneous predicate, $\mathsf{run}(\mathsf{j}, e_a)$ follows immediately from $\mathsf{run}(\mathsf{j}, e)$.

(ii) $\rho(\tau_\mathrm{s}(e_a)) = \mathsf{miles}(2)$ is by assumption.

(iii) From (4.65b), it follows that $\mathsf{comp}(\tau_\mathrm{s}(e), \tau_\mathrm{s}(e_a))$ is a consequence of $\mathsf{comp}(e, e_a)$; since $\lambda p.\, \mathsf{along}(\mathsf{the}(\mathsf{river}), p)$ is a homogeneous predicate, $\mathsf{along}(\mathsf{the}(\mathsf{river}), \tau_\mathrm{s}(e_a))$ therefore follows from $\mathsf{along}(\mathsf{the}(\mathsf{river}), \tau_\mathrm{s}(e))$.

(b) (4.74b) $\models$ (4.74a).

Proof: Let e_a be an event which makes ϕ true, and let e be the process $\mathsf{gr}(e_a)$. The conjuncts in (4.74a) are then entailed as follows:

(i) Since $\lambda e.\,\mathsf{run}(\mathsf{j}, e)$ is a homogeneous predicate, $\mathsf{run}(\mathsf{j}, e)$ follows immediately from $\mathsf{run}(\mathsf{j}, e_a)$.

(ii) From (4.65b), it follows that $\tau_{\mathrm{s}}(e) = \tau_{\mathrm{s}}(\mathsf{gr}(e_a)) = \mathsf{gr}(\tau_{\mathrm{s}}(e_a))$; since $\lambda p.\,\mathsf{along}(\mathsf{the}(\mathsf{river}), p)$ is a homogeneous predicate, $\mathsf{along}(\mathsf{the}(\mathsf{river}), \tau_{\mathrm{s}}(e))$ is therefore a consequence of $\mathsf{along}(\mathsf{the}(\mathsf{river}), \tau_{\mathrm{s}}(e_a))$.

(iii) $\mathsf{comp}(e, e_a)$ follows from the definitions of comp and gr.

(iv) $\rho(\tau_{\mathrm{s}}(e_a)) = \mathsf{miles}(2)$ is true by assumption.

As might be expected, the machinery developed thus far is almost sufficient to account for the following downward entailment:

(4.77) $\models$ Jack ran along the river for ten minutes. / Jack ran along the river for nine minutes.

The closed sentence radicals for the antecedent and consequent appear below:

(4.78) (a) $\exists e_a.\,\exists e.\left[\begin{array}{l}\mathsf{run}(\mathsf{j}, e) \wedge \mathsf{along}(\mathsf{the}(\mathsf{river}), \tau_{\mathrm{s}}(e)) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10)\end{array}\right]$

(b) $\exists e_b.\,\exists e.\left[\begin{array}{l}\mathsf{run}(\mathsf{j}, e) \wedge \mathsf{along}(\mathsf{the}(\mathsf{river}), \tau_{\mathrm{s}}(e)) \wedge \\ \mathsf{comp}(e, e_b) \wedge \rho(\tau_{\mathrm{t}}(e_b)) = \mathsf{minutes}(9)\end{array}\right]$

To make this entailment go through, we need to guarantee that e_a has a subevent e_b of duration nine minutes. Unfortunately, postulating the existence of subevents of a given size runs us directly into the minimal parts problem. Rather than going to the trouble of codifying the lower bounds on the amounts of such subevents, I will simply make the following idealization:

(4.79) **Condition on the Existence of Subevents of Lesser Amounts:** $\forall e_a.\,\forall \alpha.\,[[\mathsf{DirectedMotion}(e_a) \wedge \alpha \preceq_{\mathrm{a}} \rho(\tau(e_a))] \rightarrow \exists e_b.\,[e_b \sqsubseteq_{\mathrm{q}} e_a \wedge \rho(\tau(e_b)) = \alpha]]$

Postulate (4.79) requires all directed motion events e_a to have subevents e_b of arbitrary smaller distances or duration α.[25] The addition of this postulate suffices to prove the above entailment:

(4.80) **Theorem:** (4.78a) $\models$ (4.78b).

Proof: Let e be a process and e_a an event which make (4.78a) true. Then the first and second conjuncts of (4.78b) follow by assumption. By postulate (4.79) there exists a subevent e_b of e_a such that $\rho(\tau_{\mathrm{t}}(e_b)) = \mathsf{minutes}(9)$; finally, the third conjunct $\mathsf{comp}(e, e_b)$ follows by definition.

[25] While this is clearly too strong, note that it still does not require directed motion events to be atomic.

4.4.2 Substances and Processes

Let us now consider how the preceding treatment of paths can be generalized to verbs such as *dribble, drip, leak, ooze, pour, seep, siphon, spurt, stream* etc.[26] With verbs such as these, we find that nominal measure phrases can be used instead of adverbial ones:

(4.81) (a) Jack poured wort into the carboy for ten seconds.
(b) Jack poured twenty liters of wort into the carboy in ten seconds.

We also find that these sentences are downward entailing on amounts:

(4.82) $\models$ $\dfrac{\text{Jack poured twenty liters of wort into the carboy.}}{\text{Jack poured ten liters of wort into the carboy.}}$

To aid in analyzing these sentences, I will introduce the following eventuality subsort, which I will again assume has defined spatial traces:

(4.83) (a) $\mathsf{SubstanceEmission} \leq \mathsf{NonState} \sqcap \mathsf{Dimension}^1 \sqcap \mathsf{Plural}^-$
(b) $\forall e.\, [\mathsf{SubstanceEmission}(e) \rightarrow \mathsf{Defined}(\tau_{\mathrm{s}}(e))]$

An example of a lexical specification involving this sort appears below:[27]

(4.84) (a) $pour \;\mapsto\; \mathsf{s} \setminus \mathsf{np} / \mathsf{np} \;:\; \lambda y.\, \lambda x.\, \lambda e.\, \mathsf{pour}(x, y, e)$
(b) $\forall x.\, \forall y.\, \forall e.\, [\mathsf{Defined}(\mathsf{pour}(x, y, e)) \leftrightarrow \left[\begin{array}{c} \mathsf{Agent}(x) \wedge \mathsf{Material}(y) \wedge \\ \mathsf{SubstanceEmission}(e) \end{array}\right]]$

Interestingly, example (4.81a) shows that a process expression (*Jack pour wort*) can sometimes combine with a delimited path expression (*into the carboy*) and still yield a process expression (*Jack pour wort into the carboy*); this is in direct contrast to the pair *Jack run* and *to the bridge* discussed above, where we found this was not possible. Intuitively, what distinguishes these two cases is as follows: with *run*, what changes when we look at progressively larger or smaller subevents is how much of the path is traversed; with *pour*, however, what changes is how much of the substance is transferred—the path remains more or less the same. Accordingly, I will require the spatial traces of such processes to be constant, as in the case of canonical states:

(4.85) (a) $\forall e.\, [\mathsf{SubstanceEmission}(e) \rightarrow \mathsf{Delimited}^+(\tau_{\mathrm{s}}(e))]$
(b) $\forall e.\, \forall p.\, \left[\begin{array}{l} \mathsf{SubstanceEmission}(e) \rightarrow \\ [\tau_{\mathrm{s}}(e) = p \leftrightarrow \forall e_a.\, [\mathsf{comp}(e, e_a) \rightarrow \tau_{\mathrm{s}}(e_a) = p]] \end{array}\right]$

[26] Cf. Levin (1993, p. 237) for a discussion of this class.

[27] Note that this sort name is somewhat misleading, since the object NPs need not be singular—cf. *Jack poured the hops into the pot.*

As the preceding discussion suggests, the challenge in the case of a substance emission process is to establish an appropriate linking between it and its associated material continuum, rather than to its spatial trace. To do so, I will introduce the general notion of an **incremental thematic relation**, where 'incremental' is used here in the same sense of Dowty's (1991) term 'incremental theme'.[28] As in Krifka's (1989) approach, I will take thematic relations to hold between entities (usually material ones) and eventualites; however, following Dowty, I will not assume these relations are primitive. For example, in the case of the verb *pour*, its translation can be required to form an incremental thematic relation (ITR) as follows:

(4.86) $\forall x.\,[\mathsf{ITR}(\lambda y.\,\lambda e.\,[\mathsf{pour}(x,y,e)])]$

Ensuring that the entities to which an incremental thematic relation applies are related in the desired way essentially consists of generalizing the interaction of directed motion eventuality predicates with the spatial trace function to arbitrary thematic relations. In formal terms, this means that we are looking for an axiom which entails $\mathcal{L}_\sigma$-sentences such as (4.87):

(4.87) $\mathsf{ITR}(\lambda p.\,\lambda e.\,[\mathsf{run}(\mathsf{j},e)\ \wedge\ \tau_{\mathsf{s}}(e)=p])$

As intimated above, the desired postulate may be formulated in a fashion analogous to the definition of homogeneous predication:

(4.88) **Definition (Incremental Thematic Relation):**

$$\text{(a)}\ \forall R.\,\forall x.\,\forall y.\left[\begin{array}{l}[\mathsf{ITR}(R)\ \wedge\ \mathsf{Defined}(R(x)(y))]\ \rightarrow\\ [\mathsf{Delimited}^{+}(x,y)\ \vee\ \mathsf{Delimited}^{-}(x,y)]\end{array}\right]$$

$$\text{(b)}\ \forall R.\,[\mathsf{ITR}(R)\ \leftrightarrow$$
$$\left[\begin{array}{l}\forall x.\,\forall e.\,\forall x_a.\left[\begin{array}{l}[R(x)(e)\ \wedge\ \mathsf{comp}(x,x_a)]\ \rightarrow\\ \exists e_a.\,[R(x_a)(e_a)\ \wedge\ \mathsf{comp}(e,e_a)]\end{array}\right]\ \wedge\\ \forall x.\,\forall e.\,\forall e_a.\left[\begin{array}{l}[R(x)(e)\ \wedge\ \mathsf{comp}(e,e_a)]\ \rightarrow\\ \exists x_a.\,[R(x_a)(e_a)\ \wedge\ \mathsf{comp}(x,x_a)]\end{array}\right]\ \wedge\\ \forall x_a.\,\forall e_a.\,[R(x_a)(e_a)\ \rightarrow\ R(\mathsf{gr}(x_a))(\mathsf{gr}(e_a))]\end{array}\right]]$$

Axiom (4.88) defines an incremental thematic relation to be one that is preserved by the mappings comp and gr on appropriate pairs. In other words, an incremental thematic relation R is one that

(i) preserves delimitedness,

[28] Dowty's use of the term 'incremental' is meant to capture the bit-by-bit nature of such eventualities; note that this use of the term should not be confused with the notion of incrementality involved in either incremental processing or dynamic semantics.

(ii) establishes a bidirectional mapping between the (usually material) continuum x and the eventuality continuum e, if it holds of x and e, and

(iii) holds of the continua $\mathsf{gr}(x_a)$ and $\mathsf{gr}(e_a)$, if x_a and e_a are related in this way.

Note that the above definition supports the following analogue of the divisive reference lemma:

(4.89) **Lemma:** An incremental thematic relation is preserved by the part-of relations $\sqsubseteq$, i.e.

$$\forall R.\,[\mathsf{ITR}(R) \rightarrow \left[\begin{array}{l} \forall x_a.\,\forall e_a.\,\forall x_b.\left[\begin{array}{l}[R(x_a)(e_a) \wedge x_b \sqsubseteq x_a] \rightarrow \\ \exists e_b.\,[R(x_b)(e_b) \wedge e_b \sqsubseteq e_a]\end{array}\right] \\ \forall x_a.\,\forall e_a.\,\forall e_b.\left[\begin{array}{l}[R(x_a)(e_a) \wedge e_b \sqsubseteq e_a] \rightarrow \\ \exists x_b.\,[R(x_b)(e_b) \wedge x_b \sqsubseteq x_a]\end{array}\right]\end{array}\right]]$$

Proof: Consider the first case. By the definitions of comp and gr, we have $\mathsf{comp}(\mathsf{gr}(x_a), x_b)$ and $\mathsf{comp}(\mathsf{gr}(e_a), e_b)$; since R satisfies ITR, R is preserved by gr and comp; consequently, we have $R(\mathsf{gr}(x_a), \mathsf{gr}(e_a))$, which then guarantees the existence of such an e_b satisfying $R(x_b, e_b)$. The second case is symmetric.

Furthermore, $\mathcal{L}_\sigma$-sentences such as (4.87) do in fact turn out to be valid, as desired:

(4.90) **Lemma:** $\models$ (4.87)

Proof: This follows from $\mathsf{Hom}(\lambda e.\,[\mathsf{run}(\mathsf{j}, e)])$ and the fact that the spatial trace function τ_{s} is preserved by comp and gr in the case of directed motion eventualities.

Let us now examine how the notion of incremental thematic relation can be used to explain the distribution of the temporal adverbials in example (4.81). The closed sentence radicals for these sentences appear below:

(4.91) (a) $\exists e_a.\exists e.\exists x.\left[\begin{array}{l}\mathsf{wort}(x) \wedge \mathsf{pour}(\mathsf{j}, x, e) \wedge \mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_{\mathrm{s}}(e)) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{seconds}(10)\end{array}\right]$

(b) $\exists e_a.\,\exists x_a.\,\exists x.\left[\begin{array}{l}\mathsf{wort}(x) \wedge \mathsf{comp}(x, x_a) \wedge \rho(x_a) = \mathsf{liters}(20) \wedge \\ \mathsf{pour}(\mathsf{j}, x_a, e_a) \wedge \mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_{\mathrm{s}}(e_a)) \wedge \\ \rho(\tau_{\mathrm{t}}(e_a)) \preceq_{\mathrm{a}} \mathsf{seconds}(10)\end{array}\right]$

In (4.91a), the predicate wort allows x to be a substance; since pour forms an incremental thematic relation, this then forces e to be process. Because the spatial trace of this process is a delimited path, the path predicate into is unproblematic, and thus the composed-of relation comp introduced by the measure adverbial (*for ten seconds*) can apply at this point to cut out an event e_a

of duration ten seconds. In (4.91b), on the other hand, the composed-of relation introduced by the nominal measure phrase cuts out a twenty-liter quantity x_a from the substance continuum x, which then forces e_a to be an event. As such, a restrictive modifier (*in ten seconds*) is a forced move in this case.

Turning now to the downward entailments, we will again need to establish the existence of subevents of a given duration; this time though, we will also need to establish the existence of subparts of a given quantity:[29]

(4.92) **Condition on the Existence of Subevents involving Lesser Quantities:**

$$\forall R.\, \forall x_a.\, \forall e_a.\, \left[\begin{array}{l} \left[\begin{array}{l} \mathsf{ITR}(R) \wedge \\ R(x_a)(e_a) \end{array} \right] \rightarrow \\ \forall \alpha.\, \left[\alpha \preceq_{\mathrm{a}} \rho(x_a) \rightarrow \exists x_b.\, \exists e_b.\, \left[\left[\begin{array}{l} R(x_b)(e_b) \wedge \\ x_b \sqsubseteq_{\mathrm{q}} x_a \wedge \\ e_b \sqsubseteq_{\mathrm{q}} e_a \end{array} \right] \wedge \rho(x_b) = \alpha \right] \right] \end{array} \right]$$

Given this idealized condition, the downward entailment in (4.82) follows directly from lemma (4.89).

4.4.3 Plurals and Bare Plurals

To keep matters manageable, I will focus here on the **distributive** readings of plurals, to the exclusion of their collective, cumulative and other possible ones.[30] Distributive readings are the ones in which the predication is applied to each member of a plurality; for example, in (4.93) below, the distributive reading is simply the one where there are twenty swans and each of them is understood to have glided past the dock:

(4.93) Twenty swans glided past the dock in ten minutes.

To formalize examples such as these, I will introduce the following plural operator plur, defined in terms of the atomic individual part-of relation $\sqsubseteq_{\mathrm{ai}}$:[31]

(4.94) **Definition (Plural Operator):**
$\forall P.\, \forall y.\, [\mathsf{plur}(P)(y) \leftrightarrow \forall x.\, [x \sqsubseteq_{\mathrm{ai}} y \rightarrow P(x)]]$

For simplicity, I will assume that plural nouns and cardinals receive the following lexical specifications:

[29] I will omit the durational case, which is completely analogous.

[30] Again, cf. (Landman, 1989a; Landman, 1989b) and (Schwarzschild, 1992) for a lively discussion of collective/group readings, and also (Krifka, 1989; Krifka, 1990) for a discussion of cumulative ones.

[31] Note that this definition requires $\mathsf{plur}(P)$ to apply to atoms. As such, one is entitled to wonder why *swans* should seem strange when applied to a single swan, and why * *a swans* is ungrammatical. Although one might consider pursuing other analyses, I will simply assume that the former case is one of Gricean implicature, and the latter case one of syntactic agreement.

(4.95)
$$\begin{array}{lcccl} \textit{swans} & \mapsto & n & : & \mathsf{plur}(\mathsf{swan}) \\ \textit{twenty} & \mapsto & \mathsf{np}\,/\,\mathsf{n} & : & \lambda Q.\,\lambda P.\,\exists x.\,[Q(x) \wedge\ |\,x\,| = 20\ \wedge\ P(x)] \end{array}$$

These mappings yield the following translation for the NP *twenty swans*:

(4.96) $\lambda P.\,\exists x.\,[\mathsf{plur}(\mathsf{swan})(x) \wedge\ |\,x\,| = 20\ \wedge\ P(x)]$

To account for distributive readings, I will introduce a relation distr as an optional adverbial modifier, along the lines of Krifka (1989):[32]

(4.97) (a)
$$\begin{array}{rcl} \mathsf{vp} : P & \Longrightarrow & \mathsf{vp} : \mathit{SubjDistr}(P) \\ \mathsf{vp}\,/\,\mathsf{np} : Q & \Longrightarrow & \mathsf{vp}\,/\,\mathsf{np} : \mathit{ObjDistr}(Q) \end{array}$$

(b)
$$\begin{array}{l} \mathit{SubjDistr} \equiv \lambda P.\,\lambda x.\,\lambda e.\,[\mathsf{distr}(P, x, e)] \\ \mathit{ObjDistr} \equiv \lambda Q.\,\lambda y.\,\lambda x.\,\lambda e.\,[\mathsf{distr}(\lambda y.\,\lambda e.\,[Qyxe], y, e)] \end{array}$$

(4.98) **Definition (Distributive Relation):**
$$\begin{array}{l} \forall P.\,\forall x.\,\forall e.\,\mathsf{distr}(P, x, e) \leftrightarrow \\ \left[\begin{array}{l} \forall x'.\,[x' \sqsubseteq_{\mathrm{ai}} x \rightarrow \exists e'.\,[e' \sqsubseteq_{\mathrm{ai}} e \wedge P(x')(e')]] \wedge \\ \forall e'.\,[e' \sqsubseteq_{\mathrm{ai}} e \rightarrow \exists x'.\,[x' \sqsubseteq_{\mathrm{ai}} x \wedge P(x')(e')]] \end{array}\right] \end{array}$$

With these additions, our grammar assigns (4.93) the following sentence radical (using first *SubjDistr* then *SubjLift*):

(4.99)
$$\lambda e.\,\exists x.\left[\begin{array}{l} \mathsf{plur}(\mathsf{swan})(x) \wedge\ |\,x\,| = 20 \wedge \\ \mathsf{distr}\left(\lambda x.\,\lambda e.\left[\begin{array}{l} \mathsf{glide}(x, e) \wedge \\ \mathsf{past}(\mathsf{the}(\mathsf{dock}), \tau_{\mathrm{s}}(e)) \end{array}\right], x, e\right) \wedge \\ \rho(\tau_{\mathrm{t}}(e)) \preceq_{\mathrm{a}} \mathsf{minutes}(10) \end{array}\right]$$

Note that the *in*-adverbial here requires the duration of the meta-event to be less than ten minutes, where the duration of the meta-event is equal to the sum of the durations of its atomic subevents (cf. section 3.3.3). Because these durations are totaled, (4.99) will hold of an event whose atomic subevents have an average duration of thirty seconds or less, irrespective of how much time elapses between each of these subevents. While this reading does seem possible, it is not the most natural one in this case, where we are likely to include the inbetween times.

To account for this latter reading, we may employ the **convex hull** operator defined in section 3.3.3:

(4.100) (a)
$$\begin{array}{lcccl} \textit{for} & \mapsto & \mathsf{vp} \setminus \mathsf{vp}\,/\,\mathsf{np} & : & \mathit{For}_c \\ \textit{in} & \mapsto & \mathsf{vp} \setminus \mathsf{vp}\,/\,\mathsf{np} & : & \mathit{In}_c \end{array}$$

(a)
$$\begin{array}{lcl} \mathit{For}_c & \equiv & \lambda d.\,\lambda P.\,\lambda x.\,\lambda e_a.\,\exists e.\,[P(x)(e) \wedge\ \mathsf{comp}(e, e_a) \wedge\ \rho(\mathsf{cvx}(\tau(e_a))) = d] \\ \mathit{In}_c & \equiv & \lambda d.\,\lambda P.\,\lambda x.\,\lambda e.\,[P(x)(e) \wedge\ \rho(\mathsf{cvx}(\tau_{\mathrm{t}}(e))) \preceq_{\mathrm{a}} d] \end{array}$$

The closed sentence radical assigned to (4.93) under the convex reading appears below:

[32] Of course, as in the case of the type-lifting operators, additional operators are necessary for other syntactic positions.

$$(4.101)\quad \exists e.\,\exists x.\left[\begin{array}{l}\mathsf{plur}(\mathsf{swan})(x)\wedge\ |x|=20\wedge\\ \mathsf{distr}\left(\lambda x.\,\lambda e.\left[\begin{array}{l}\mathsf{glide}(x,e)\wedge\\ \mathsf{past}(\mathsf{the}(\mathsf{dock}),\tau_{\mathrm{s}}(e))\end{array}\right],x,e\right)\wedge\\ \rho(\mathsf{cvx}(\tau_{\mathrm{t}}(e)))\preceq_{\mathrm{a}}\mathsf{minutes}(10)\end{array}\right]$$

In the preceding exposition we have implicitly assumed that the plural operator plur and the distributive relation distr only make sense for delimited entities. However, these higher-order terms may be sensibly extended to apply to undelimited entities as well, by simply requiring (i) that the plural operator form homogeneous predicates, and (ii) that the distributive relation form incremental thematic relations:

(4.102) (i) $\forall P.\,\mathsf{Hom}(\mathsf{plur}(P))$

(ii) $\forall P.\,\mathsf{ITR}(\lambda x.\,\lambda e.\,[\mathsf{distr}(P,x,e)])$

Naturally, this generalization permits the analysis of (4.81) to be extended to the following pair:

(4.103) (a) Swans glided past the dock for ten minutes.

(b) Twenty swans glided past the dock in ten minutes.

Much the same as before, only (4.103a) will allow for temporal measure adverbials, and (4.103a) will be downward entailing on cardinalities.

It is important to note that in each of the examples above, the distributive relation is introduced in the derivation prior to the temporal adverbial; consequently, the temporal adverbial specifies the duration of the entire meta-event, rather than the duration of each atomic event in this collection. However, since the distributive relation is introduced as an optional adverbial modifier, nothing prevents the reverse ordering. This predicts that there should also be a reading in which the temporal adverbial is understood distributively; furthermore, it predicts that two temporal adverbials should be able to coherently appear in the same sentence, as long as the first is interpreted distributively and the second collectively. As we saw in section 2.1, such examples do indeed exist; here we will consider the following pair:

(4.104) (a) Swans glided past the dock in 30 seconds for 10 minutes.

(b) Twenty swans glided along the shore for 30 seconds in 10 minutes.

The closed sentence radicals assigned to (4.104) appear below:

(4.105) (a) $\exists e_a.\, \exists e.\, \exists x.$
$$\left[\begin{array}{l} \mathsf{plur(swan)}(x) \wedge \\ \mathsf{distr}\left(\lambda x.\, \lambda e.\, \left[\begin{array}{l} \mathsf{glide}(x,e) \wedge \\ \mathsf{past}(\mathsf{the}(\mathsf{dock}), \tau_{\mathrm{s}}(e)) \wedge \\ \rho(\tau_{\mathrm{t}}(e)) \preceq_{\mathrm{a}} \mathsf{seconds}(30) \end{array}\right], x, e\right) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\mathsf{cvx}(\tau_{\mathrm{t}}(e_a))) = \mathsf{minutes}(10) \end{array}\right]$$

(b) $\exists e_b.\, \exists x_b.$
$$\left[\begin{array}{l} \mathsf{plur(swan)}(x_b) \wedge \ |x_b| = 20 \wedge \\ \mathsf{distr}\left(\lambda x_a.\, \lambda e_a.\, \exists e.\, \left[\begin{array}{l} \mathsf{glide}(x,e) \wedge \\ \mathsf{along}(\mathsf{the}(\mathsf{shore}), \tau_{\mathrm{s}}(e)) \wedge \\ \mathsf{comp}(e, e_a) \wedge \\ \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{seconds}(30) \end{array}\right], x_b, e_b\right) \wedge \\ \rho(\mathsf{cvx}(\tau_{\mathrm{t}}(e_b))) \preceq_{\mathrm{a}} \mathsf{minutes}(10) \end{array}\right]$$

Since the distributive readings of these adverbials are analyzed as falling under our distributive relation, it should be clear that the following sentences are straightforwardly entailed by the definition of distr:[33]

(4.106) (a) A swan glided past the dock in 30 seconds.
(b) A swan glided along the shore for 30 seconds.

4.5 Worlds and Times

In order to address the modal aspect of the imperfective paradox, I will introduce a simple distinction between 'actual' and 'non-actual' events. Since this bare bones treatment of possible worlds in terms of 'actuality' can be developed within $\mathcal{L}_\sigma$, I will not complicate the denotation function by introducing world and time parameters; note, however, that this is a matter of convenience only.

To formalize actuality, I will make use of the structural part-of relation $\trianglelefteq$ (cf. section 3.3.1), which has been largely ignored to this point. To motivate some of the necessary machinery, let us first consider the present treatment of negation, which likewise involves the structural part-of relation.

4.5.1 Negation

Recall that in the domain of objects, the structural part-of relation $\trianglelefteq$ is the one holding between Landman's hand and Landman himself; in the domain of events, it makes sense to think of this relation as the one holding between the 'contingently' related events making up an 'episode', to use the terminology of both Moens and Steedman (1988) and Hwang and Schubert (1992). While there is much one might want to say about the contingency relations subsumed by the structural part-of relation, I will restrict my attention here to the interaction of the structural part-of relation with the quantity part-of relation.

The first condition that I will impose on the structural part-of relation concerns temporal inclusion; if this relation is the one holding between an event and

[33] Assuming that *a swan* is translated as $\lambda P.\, \exists x.\, [\mathsf{swan}(x) \wedge P(x)]$.

its contingently related subevents, clearly these subevents should be temporally included in the larger event:

(4.107) $\forall e.\, \forall e'.\, [e' \trianglelefteq e \;\rightarrow\; \tau_{\mathrm{t}}(e') \sqsubseteq_{\mathrm{q}} \tau_{\mathrm{t}}(e)]$

The second condition concerns persistence; if an event has a certain collection of contingently related subevents, then a quantity super-event of this event should have matching quantity super-events of its contingently related subevents:

(4.108) $\forall e_b.\, \forall e_a.\, \forall e_b'.\, [e_b \sqsubseteq_{\mathrm{q}} e_a \;\wedge\; e_b' \trianglelefteq e_b \;\rightarrow\; \exists e_a'.\, [e_b' \sqsubseteq_{\mathrm{q}} e_a' \;\wedge\; e_a' \trianglelefteq e_a]]$

Going in the other direction, we would like a quantity subevent to have matching quantity subevents of its contingently related subevents; moreover, we would like these matching quantity subevents to be as large as possible, if they overlap at all:

(4.109) $\forall e_b.\, \forall e_a.\, \forall e_b'.\, \forall e_a'.\, \left[\begin{array}{l} [e_b \sqsubseteq_{\mathrm{q}} e_a \;\wedge\; e_b' \sqsubseteq_{\mathrm{q}} e_a' \;\wedge\; e_a' \trianglelefteq e_a] \;\rightarrow \\ [e_b' \trianglelefteq e_b \;\leftrightarrow\; [\mathrm{(a)} \wedge \mathrm{(b)}]] \end{array}\right]$

(a) $\tau_{\mathrm{t}}(e_b') \sqsubseteq_{\mathrm{q}} \tau_{\mathrm{t}}(\trianglelefteq e_b)$

(b) $\forall e_b''.\, [[e_b' \sqsubseteq_{\mathrm{q}} e_b'' \;\wedge\; \tau_{\mathrm{t}}(e_b'') \sqsubseteq_{\mathrm{q}} \tau_{\mathrm{t}}(\trianglelefteq e_b)] \;\rightarrow\; e_b' = e_b'']$

To help visualize the intended interaction of the structural and quantity part-of relations, Figure 4.10 shows how an event with contingently related subevents is divided in half, quantity-wise.

As suggested above, we may use the structural part-of relation to account for negation. Negation, as Krifka (1989) observes, is a complicated matter for event-based semantic approaches, since negated expressions are not persistent; to handle negation correctly then, one has to consider situations which are 'large enough'. However, one cannot just move directly to the propositional level, and assume that negation is a sentential operator—that would falsely predict that negation always takes wide scope over event predicate modifiers such as temporal adverbials.

To resolve this impasse, Krifka suggests we define negation in terms of **maximal events** at a certain time. As an example, he suggests that *John didn't laugh* as an event predicate holds of maximal events which do not contain an event of John's laughing; note that as we look at maximal events with increasingly larger intervals, the truth value of this predicate can switch, as desired.[34]

Krifka's account of negation can be reconstructed here in terms of the structural part-of relation $\trianglelefteq$. First, we need to define the notion of a maximal eventuality at a certain time according to $\trianglelefteq$:

[34]Clearly though, the intended maximal event need not be globally maximal, only maximal with respect to some implicit contextual restriction; I will not address this matter here.

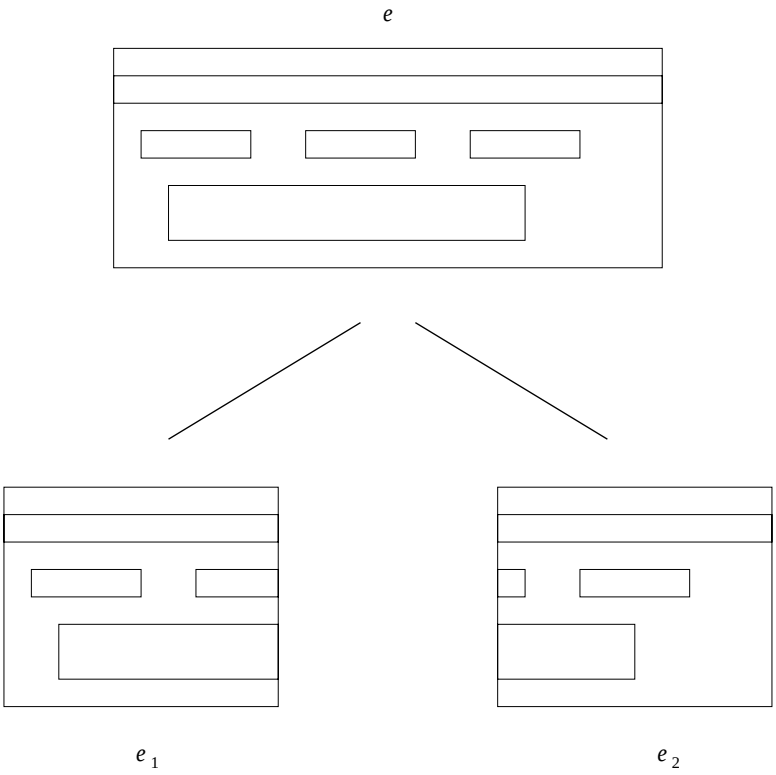


Figure 4.10: The interaction of the structural part-of relation $\trianglelefteq$ and the quantity part-of relation $\sqsubseteq_{\mathrm{q}}$. The structural part-of relation is indicated by box inclusion, where the horizontal axis represents time; the quantity part-of relation between the e_i and e is represented by the diagonal lines (here $e = e_1 \sqcup_{\mathrm{q}} e_2$). The diagram illustrates how the quantity subevents e_1 and e_2 have structural subevents matching the ones in e.

(4.110) (a) **Definition (Maximal Eventuality at a Certain Time):**
$\forall e.\, \forall t.\, [\mathsf{Max}_{\trianglelefteq}(e,t) \leftrightarrow$
$\left[\begin{array}{l} \tau_{\mathrm{t}}(e) = t \wedge \\ \forall e'.\, [[e \trianglelefteq e' \wedge \tau_{\mathrm{t}}(e) = \tau_{\mathrm{t}}(e')] \rightarrow e = e'] \end{array}\right]]$

(b) **Definition (Maximal Eventuality at Some Time):**
$\forall e.\, [\mathsf{Max}_{\trianglelefteq}(e) \leftrightarrow \exists t.\, [\mathsf{Max}_{\trianglelefteq}(e,t)]]$

(c) $\mathsf{Hom}(\mathsf{Max}_{\trianglelefteq})$

Given this definition, VP-negation can be translated as follows:

(4.111) (a) *do not* $\mapsto$ $\mathsf{vp}\,/\,\mathsf{vp}$: $\lambda P.\, \lambda x.\, \mathsf{neg}(P)$

(b) $\forall P.\, \forall e.\, \mathsf{neg}(P)(e) \leftrightarrow [\mathsf{Max}_{\trianglelefteq}(e) \wedge \neg \exists e'.\, [P(e') \wedge e' \trianglelefteq_{\mathrm{q}} e]]$

(c) $\forall x.\, \forall y.\, [x \trianglelefteq_{\mathrm{q}} y \leftrightarrow \exists z.\, [x \sqsubseteq_{\mathrm{q}} z \wedge z \trianglelefteq y]]$

This translation yields the following sentence radical for *Jack didn't fill the carboy*:

(4.112) $\mathsf{neg}(\lambda e'.\, \mathsf{fill}(\mathsf{j}, \mathsf{the}(\mathsf{carboy}), e'))$

As defined above, the negation operator neg makes (4.112) true of maximal eventualities e not containing as a part an event e' of Jack filling the carboy.

Because the structural part-of relation is persistent, eventuality predicates formed using the negation operator turn out downward entailing:

(4.113) **Lemma:** For all event predicates P, $\mathsf{neg}(P)$ is downward entailing with respect to the quantity part-of relation $\sqsubseteq_{\mathrm{q}}$.

Proof: Suppose $e_b \sqsubseteq_{\mathrm{q}} e_a$, and $\mathsf{neg}(P)(e_a)$ but not $\mathsf{neg}(P)(e_b)$. Since $\mathsf{neg}(P)(e_b)$ is false, and since $\mathsf{Max}_{\trianglelefteq}$ is homogeneous, there must be an ${e_b}'$ and an e' such that $P(e')$ and $e' \sqsubseteq_{\mathrm{q}} {e_b}' \trianglelefteq e_b$. But, since $\trianglelefteq$ is persistent, we have ${e_b}' \sqsubseteq_{\mathrm{q}} {e_a}' \trianglelefteq e_a$, and thus $e' \trianglelefteq_{\mathrm{q}} e_a$, by the transitivity of $\sqsubseteq_{\mathrm{q}}$. This suffices to make $\mathsf{neg}(P)(e_a)$ false, a contradiction.

This lemma enables us to require the negation operator to form homogeneous predicates:

(4.114) $\forall P.\, \mathsf{Hom}(\mathsf{neg}(P))$

As such, we may now derive the following closed sentence radical for *Jack didn't fill the carboy for ten minutes*, where the temporal adverbial takes wide scope over the negation:

(4.115) $\exists e_a.\, \exists e.\, \left[\begin{array}{l} \mathsf{neg}(\lambda e'.\, \mathsf{fill}(\mathsf{j}, \mathsf{the}(\mathsf{carboy}), e'))(e) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10) \end{array}\right]$

To clarify this reading, note that by the definitions of homogeneous predication and the negation relation, (4.115) is equivalent to the following $\mathcal{L}_\sigma$-sentence:

$$(4.116)\quad \exists e_a.\ \left[\begin{array}{l} \mathsf{Max}_{\trianglelefteq}(e_a)\ \wedge\ \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10) \wedge \\ \neg\exists e'.\,[\mathsf{fill}(\mathsf{j}, \mathsf{the}(\mathsf{carboy}), e')\ \wedge\ e' \trianglelefteq_{\mathrm{q}} e_a] \end{array}\right]$$

4.5.2 Tense and Mood

Returning now to our discussion of actuality, we may begin by noting that the structural part-of relation $\trianglelefteq$ differs from the individual part-of relation $\sqsubseteq_{\mathrm{i}}$ in that only the latter is required to form a complete join semi-lattice. As such, there need not be a unique upper bound according to $\trianglelefteq$ of eventualities at a certain time. This enables us to think of the various maximal eventualities as representing the various possible states of affairs, only one of which is the actual one.

To formalize this notion, I will introduce a distinguished predicate Actual. Our first condition on this predicate, naturally, is that it uniquely select a maximal eventuality at a certain time:

$$(4.117)\quad \forall e.\forall e'.\forall t.[[\mathsf{Max}_{\trianglelefteq}(e,t) \wedge \mathsf{Max}_{\trianglelefteq}(e',t) \wedge \mathsf{Actual}(e) \wedge \mathsf{Actual}(e')]\ \rightarrow\ e = e']$$

Our second condition on Actual is that it preserve the part-of relations:

$$(4.118)\quad \begin{array}{ll} \text{(a)} & \forall e.\,\forall e'.\,[[e' \trianglelefteq e\ \wedge\ \mathsf{Actual}(e)]\ \rightarrow\ \mathsf{Actual}(e')] \\ \text{(b)} & \forall e.\,\forall e'.\,[[e' \sqsubseteq e\ \wedge\ \mathsf{Actual}(e)]\ \rightarrow\ \mathsf{Actual}(e')] \end{array}$$

These axioms suffice for present purposes; since we are not dealing with verbs of creation, I will leave open the issue of how to extend this notion of actuality to the material entities.

To introduce the distinguished predicate Actual in a derivation, I will simply link it to the tense morpheme, as shown below:[35]

$$(4.119)\quad \begin{array}{lll} \text{(a)} & \begin{array}{lcl} \textit{*past*} & \mapsto & (\mathsf{u} \setminus \mathsf{np}) / (\mathsf{s} \setminus \mathsf{np}) \\ \textit{*pres*} & \mapsto & (\mathsf{u} \setminus \mathsf{np}) / (\mathsf{s} \setminus \mathsf{np}) \end{array} & \begin{array}{l} :\ PAST \\ :\ PRES \end{array} \\ \text{(b)} & \begin{array}{lcl} PAST & \equiv & \lambda P.\,\lambda x.\,\exists e.\,P(x)(e)\ \wedge\ \tau_{\mathrm{t}}(e) \prec_{\mathrm{i}} \mathsf{now}\ \wedge\ \mathsf{Actual}(e) \\ PRES & \equiv & \lambda P.\,\lambda x.\,\exists e.\,P(x)(e)\ \wedge\ \tau_{\mathrm{t}}(e) = \mathsf{now}\ \wedge\ \mathsf{Actual}(e) \end{array} & \\ \text{(c)} & \mathsf{Instant}(\mathsf{now}) & \end{array}$$

Again, rather than parametrizing the denotation function, I will simply employ a distinguished instant, now, to capture the temporal import of these two tenses.[36] Note that since the present tense requires the temporal trace to be equal to now, it is unlike the past tense in being restricted to the zero-dimensional entities; we will return to this difference in section 4.6.

[35] Note that I will not address here the oft-discussed definiteness effect of the past tense (cf. chapter 6).

[36] Cf. (Hinrichs, 1988; Oehrle, 1990; Hwang and Schubert, 1992) for a more sophisticated treatement.

4.5.3 The Imperfective Paradox

In the spirit of Moens and Steedman's analysis (1988), I will formalize the non-futurate meaning of the progressive as mapping a process to a state in which that process is 'in progress'. To capture this notion formally, I will require in-progress states to have events composed of the process as structural parts; of course, as we look at progressively larger or smaller such states, the particular events will vary. In axiomatizing this idea, I will introduce the relation **inprog**, as follows:

$$(4.120)\quad \forall e'.\,\forall e.\,[\mathsf{inprog}(e',e) \;\leftrightarrow\; \exists e_a'.\,\exists e_a.\,\left[\begin{array}{l}\mathsf{comp}(e',e_a') \wedge e \sqsubseteq_{\mathrm{q}} e_a \wedge \\ e_a' \trianglelefteq e_a \wedge \tau_{\mathrm{t}}(e_a') = \tau_{\mathrm{t}}(e_a)\end{array}\right]]$$

According to (4.120), e is a state in which the process e' is in progress iff e is a quantity part of a state e_a that has as a structural part an event e_a' composed of e' taking place at the same time. Unpacking this definition, there are three points to note here. First, allowing e to be a quantity part of e_a enables e to be atomic even if e_a' is not. Second, the structural part-of relation can be used to smooth over gaps (cf. Figure 4.10); consequently, this definition applies equally well to the plural analogues of processes.[37] Finally, since inprog forms downward-entailing predicates, we may consistently require inprog to form homogeneous predicates, as shown below:

$$(4.121)\quad \forall e'.\,\mathsf{Hom}(\lambda e.\,\mathsf{inprog}(e',e))$$

With the relation inprog thus defined, we may lexicalize the progressive as follows:

$$(4.122)\quad \begin{array}{l} \text{-}\quad \begin{array}{lcll} \textit{*prog*} & \mapsto & \mathsf{vp}\,/\,\mathsf{vp} & : \quad PROG_1 \\ \textit{*prog*} & \mapsto & \mathsf{vp}\,/\,\mathsf{vp} & : \quad PROG_2 \end{array} \\ \\ \text{-}\quad \begin{array}{lcl} PROG_1 & \equiv & \lambda P.\,\lambda x.\,\lambda e.\,\exists e'.\,P(x)(e') \wedge \mathsf{inprog}(e',e) \\ PROG_2 & \equiv & \lambda P.\,\lambda x.\,\lambda e.\,\exists e'.\,P(x)(e') \wedge \mathsf{inprog}(\mathsf{gr}(e'),e) \end{array} \end{array}$$

Since the relation inprog requires its first argument to be a process, these two versions of the progressive apply in complementary circumstances: the first version applies when e' is a process, and the second one when e' is an event.

As a case in point, consider the translations of the following pair of sentences:[38]

(4.123) (a) Jack was running along the river for ten minutes.
(b) Jack was running to the bridge for ten minutes.

$$(4.124)\quad \text{(a)}\ \exists e_a.\,\exists e.\,\exists e'.\,\left[\begin{array}{l}\mathsf{run}(\mathsf{j},e') \wedge \mathsf{along}(\mathsf{the}(\mathsf{river}),e') \wedge \mathsf{inprog}(e',e) \wedge \\ \mathsf{comp}(e,e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10) \wedge \\ \tau_{\mathrm{t}}(e_a) \prec_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a)\end{array}\right]$$

[37] Recall that the sort Process is restricted to singular eventualities.

[38] Note that I am focusing here on the reading where the temporal adverbial takes scope over the progressive, rather than the reverse.

$$\text{(b)}\ \ \exists e_a.\,\exists e.\,\exists e'.\ \left[\begin{array}{l}\mathsf{run}(\mathsf{j},e')\ \wedge\ \mathsf{to}(\mathsf{the}(\mathsf{bridge}),e')\ \wedge\ \mathsf{inprog}(\mathsf{gr}(e'),e)\ \wedge\\ \mathsf{comp}(e,e_a)\ \wedge\ \rho(\tau_{\mathrm{t}}(e_a))=\mathsf{minutes}(10)\ \wedge\\ \tau_{\mathrm{t}}(e_a)\prec_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_a)\end{array}\right]$$

As desired, (4.123a) entails the following sentence, with ensuing translation:

(4.125) Jack ran along the river for ten minutes.

$$\text{(4.126)}\quad \exists e_a{}'.\,\exists e'.\ \left[\begin{array}{l}\mathsf{run}(\mathsf{j},e')\ \wedge\ \mathsf{along}(\mathsf{the}(\mathsf{river}),e')\ \wedge\\ \mathsf{comp}(e',e_a{}')\ \wedge\ \rho(\tau_{\mathrm{t}}(e_a{}'))=\mathsf{minutes}(10)\ \wedge\\ \tau_{\mathrm{t}}(e_a{}')\prec_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_a{}')\end{array}\right]$$

The proof of this entailment follows:

(4.127) **Theorem:** (4.124a) $\models$ (4.126).

Proof: Let e_a, e and e' satisfy (4.124a). Since inprog is homogeneous, we know that $\mathsf{inprog}(e',e_a)$ must hold. Given our idealized condition on the existence of subevents, this guarantees that there is an $e_a{}' \trianglelefteq e_a$ such that $\mathsf{comp}(e',e_a{}')$ and $\tau_{\mathrm{t}}(e_a)=\tau_{\mathrm{t}}(e_a{}')$. Finally, because Actual is preserved by $\trianglelefteq$, $\mathsf{Actual}(e_a{}')$ follows from $\mathsf{Actual}(e_a)$, and thus e' and $e_a{}'$ satisfy all the conjuncts in (4.126).

Note, however, that the non-progresssive counterpart to (4.123b) is anomalous:

(4.128) * Jack ran to the bridge for ten minutes.

Furthermore, its acceptable counterpart (4.129) is not entailed by (4.123b):

(4.129) Jack ran to the bridge.

$$\text{(4.130)}\quad \exists e'.\ \left[\begin{array}{l}\mathsf{run}(\mathsf{j},e')\ \wedge\ \mathsf{to}(\mathsf{the}(\mathsf{bridge}),e')\ \wedge\\ \tau_{\mathrm{t}}(e')\prec_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e')\end{array}\right]$$

(4.131) **Theorem:** (4.124b) $\not\models$ (4.130).

Proof: Let e_a, e and e' satisfy (4.124b). Furthermore, let the duration $\rho(\tau_{\mathrm{t}}(e'))$ of the event e' be greater than ten minutes. As before, there must be an $e_a{}' \trianglelefteq e_a$ such that $\mathsf{comp}(\mathsf{gr}(e'),e_a{}')$ and $\tau_{\mathrm{t}}(e_a)=\tau_{\mathrm{t}}(e_a{}')$. But, since e' has duration greater than that of $e_a{}'$, $e_a{}'$ must be a proper subevent of e'. For this reason, nothing guarantees that e' is actual, i.e. nothing guarantees the last conjunct $\mathsf{Actual}(e')$ of (4.130).

Finally, since $\lambda e.\ \mathsf{run}(\mathsf{j},e)$ is a homogeneous predicate, (4.123b) does straightforwardly entail both *Jack was running for ten minutes* and *Jack ran for ten minutes*.

To take another example, consider the following pair:

(4.132) (a) Jack was running along the river at noon.
(b) Jack was running to the bridge at noon.

To this point we have yet to formalize *at*-adverbials; for present purposes, these may be straightforwardly lexicalized as follows:

(4.133)
$$\text{-}\quad \begin{array}{lcll} at & \mapsto & \mathsf{vp} \setminus \mathsf{vp} / \mathsf{np} & : \; At \\ noon & \mapsto & \mathsf{np} & : \; \mathsf{noon} \end{array}$$

(a) $At \;\equiv\; \lambda t.\, \lambda P.\, \lambda x.\, \lambda e.\, [P(x)(e) \wedge \mathsf{at}(e,t)]$

(b) $\forall e.\, \forall t.\, [\mathsf{Defined}(\mathsf{at}(e,t)) \rightarrow \mathsf{Instant}(t)]$

(c) $\forall e.\, \forall t.\, [\mathsf{at}(e,t) \leftrightarrow \tau_{\mathrm{t}}(e) = t]$

Translated in this way, *at*-adverbials serve to specify the instant t of an eventuality e; for simplicity, I will assume expressions such as *noon* are translated as particular instants. Given these additions, our grammar yields the following translations for (4.132):

(4.134) (a) $\exists e.\, \exists e'.\, \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e') \wedge \mathsf{along}(\mathsf{the}(\mathsf{river}), e') \wedge \mathsf{inprog}(e', e) \wedge \\ \mathsf{at}(e, \mathsf{noon}) \wedge \tau_{\mathrm{t}}(e) \prec_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e) \end{array}\right]$

(b) $\exists e.\, \exists e'.\, \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e') \wedge \mathsf{to}(\mathsf{the}(\mathsf{bridge}), e') \wedge \mathsf{inprog}(\mathsf{gr}(e'), e) \wedge \\ \mathsf{at}(e, \mathsf{noon}) \wedge \tau_{\mathrm{t}}(e) \prec_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e) \end{array}\right]$

As desired, (4.132a) can be made to entail (4.135a), without making (4.132b) entail (4.135b):

(4.135) (a) Jack ran along the river.
(b) Jack ran to the bridge.

The proofs are much the same as before; there is just one additional step at the beginning and one at the end, owing to the fact that e' is a moment. In the first case, we get $\mathsf{inprog}(e', e_a)$ for some e_a such that $e \sqsubseteq_{\mathrm{aq}} e_a$; in the second case, we likewise get $\mathsf{inprog}(\mathsf{gr}(e'), e_a)$. After this point, the proofs continue unchanged until we try to prove $\mathsf{Actual}(e_a{}')$ in the first case, and $\mathsf{Actual}(e')$ in the second. To get this conjunct to go through or block as appropriate, we need to know $\mathsf{Actual}(e_a)$; this condition must be added as an additional axiom:

(4.136) $\forall e'.\, \forall e.\, [[\mathsf{inprog}(e', e) \wedge \mathsf{Actual}(e)] \rightarrow \exists e_a.\, [e \sqsubseteq_{\mathrm{q}} e_a \wedge \mathsf{Period}(e_a) \wedge \mathsf{Actual}(e_a)]]$

This postulate states, simply enough, that every actual in-progress state e must be a part of some actual period e_a. Note that this condition is not one that makes sense for states in general, since it would preclude the existence of states holding for no longer than an instant; however, we have already seen that such states can be usefully employed in analyzing locative events, which were assumed to involve continuous motion.

4.5.4 Aspectual Verbs

We continue now with the aspectual verbs *start*, *stop*, and *finish*. Here I will focus only on the gerunds (e.g. *start running*), to the exclusion of the infinitival and plain NP cases (e.g. *start to run* and *start a book*). I will assume these verbs are lexicalize according to the following pattern:

(4.137) (a)
$$\begin{array}{lcll} \textit{start -ing} & \mapsto & \mathsf{vp}\,/\,\mathsf{vp} & : \; \mathit{START}_1 \\ \textit{start -ing} & \mapsto & \mathsf{vp}\,/\,\mathsf{vp} & : \; \mathit{START}_2 \end{array}$$

(b)
$$\begin{array}{lcl} \mathit{START}_1 & \equiv & \lambda P.\,\lambda x.\,\lambda e_0.\,\exists e'.\,P(x)(e') \;\wedge\; \mathsf{start}(e', e_0) \\ \mathit{START}_2 & \equiv & \lambda P.\,\lambda x.\,\lambda e_0.\,\exists e'.\,P(x)(e') \;\wedge\; \mathsf{start}(\mathsf{gr}(e'), e_0) \end{array}$$

As in the case of inprog, the first argument of start will be required to be a process (and likewise for stop and finish); once again, note that this forces START_1 and START_2 to apply in complementary circumstances.

We may characterize the temporal consequences of the predicate start as follows:[39]

(4.138)
$$\forall e'.\,\forall e_0.\,[\mathsf{start}(e', e_0) \;\rightarrow\; \left[\begin{array}{c} \exists e_a'.\,[\mathsf{comp}(e', e_a') \;\wedge\; \mathsf{min}(\mathsf{cvx}(\tau_{\mathrm{t}}(e_a'))) = \tau_{\mathrm{t}}(e_0)] \\ \bigwedge \\ \neg\exists e_a'.\,[\mathsf{comp}(e', e_a') \;\wedge\; \mathsf{min}(\mathsf{cvx}(\tau_{\mathrm{t}}(e_a'))) \prec_{\mathrm{t}} \tau_{\mathrm{t}}(e_0)] \end{array}\right]]$$

This postulate requires a starting event e_0 to mark the onset of its associated process e'. Naturally enough, we will require e_0 to begin a state e in which e' is in progress:[40]

(4.139) (a)
$$\forall e'.\,\forall e_0.\,[\mathsf{start}(e', e_0) \;\rightarrow\; \exists e.\,[\mathsf{inprog}(e', e) \;\wedge\; \mathsf{begin}(e, e_0)]]$$

(b)
$$\forall e.\,\forall e_0.\,[\mathsf{begin}(e, e_0) \;\rightarrow\; \left[\begin{array}{c} \exists e_a.\,[\mathsf{comp}(e, e_a) \;\wedge\; \mathsf{min}(\tau_{\mathrm{t}}(e_a)) = \tau_{\mathrm{t}}(e_0)] \\ \bigwedge \\ \neg\exists e_a.\,[\mathsf{comp}(e, e_a) \;\wedge\; \mathsf{min}(\tau_{\mathrm{t}}(e_a)) \prec_{\mathrm{t}} \tau_{\mathrm{t}}(e_0)] \end{array}\right]]$$

Note that if a state actually begins, then we will require there to be an actual period with the same initial time:

(4.140)
$$\forall e.\,\forall e_0.\,\left[\begin{array}{c} [\mathsf{begin}(e, e_0) \;\wedge\; \mathsf{Actual}(e_0)] \\ \rightarrow \\ \exists e_a.\,[\mathsf{comp}(e, e_a) \;\wedge\; \mathsf{min}(\tau_{\mathrm{t}}(e_a)) = \tau_{\mathrm{t}}(e_0) \;\wedge\; \mathsf{Actual}(e_a)] \end{array}\right]$$

[39] Note that I am not attempting to define the predicate start here, only to sufficiently characterize its meaning for present purposes. In particular, I will leave open the problem of ensuring that if *Jack started running at t* is true, then *Jack was running just before t* is false. To see why this is a problem, consider an event e_0 which starts a process e' satisfying *Jack run*: while the axiom on start guarantees that the onset of e' is marked by e_0, nothing guarantees that there is no other process e'' satisfying *Jack run* which is ongoing just before e_0. As such, the present analysis is missing a maximality condition on e', whose proper formulation I will leave for another occasion.

[40] As before, the axiom on begin is only intended to partially characterize its meaning.

These axioms suffice to make (4.141a) entail *Jack ran along the river*, while not making (4.141b) entail *Jack ran to the bridge*:

(4.141) (a) Jack started running along the river.

(b) Jack started running to the bridge.

The translations of these sentences appear below:

(4.142) (a) $\exists e_0 . \exists e' . \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e') \wedge \mathsf{along}(\mathsf{the}(\mathsf{river}), e') \wedge \mathsf{start}(e', e_0) \wedge \\ \tau_{\mathrm{t}}(e_0) \prec_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_0) \end{array} \right]$

(b) $\exists e_0 . \exists e' . \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e') \wedge \mathsf{to}(\mathsf{the}(\mathsf{bridge}), e') \wedge \mathsf{start}(\mathsf{gr}(e'), e_0) \wedge \\ \tau_{\mathrm{t}}(e_0) \prec_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_0) \end{array} \right]$

As before, the presence of the ground-from function gr in (4.142b) makes the entailment fail in this case only.

I will not treat stop completely symmetrically to start, in order to allow for mid-event interruptions; instead, I will characterize stop solely in terms of inprog:

(4.143) (a) $\forall e' . \forall e_1 . [\mathsf{stop}(e', e_1) \rightarrow \exists e . [\mathsf{inprog}(e', e) \wedge \mathsf{end}(e, e_1)]]$

(b) $\forall e . \forall e_1 . [\mathsf{end}(e, e_1) \rightarrow \left[\begin{array}{c} \exists e_a . [\mathsf{comp}(e, e_a) \wedge \mathsf{max}(\tau_{\mathrm{t}}(e_a)) = \tau_{\mathrm{t}}(e_1)] \\ \bigwedge \\ \neg \exists e_a . [\mathsf{comp}(e, e_a) \wedge \mathsf{max}(\tau_{\mathrm{t}}(e_a)) \succ_{\mathrm{t}} \tau_{\mathrm{t}}(e_1)] \end{array} \right]]$

(4.144) $\forall e . \forall e_1 . \left[\begin{array}{c} [\mathsf{end}(e, e_1) \wedge \mathsf{Actual}(e_1)] \\ \rightarrow \\ \exists e_a . [\mathsf{comp}(e, e_a) \wedge \mathsf{max}(\tau_{\mathrm{t}}(e_a)) = \tau_{\mathrm{t}}(e_1) \wedge \mathsf{Actual}(e_a)] \end{array} \right]$

These postulates require an actual stopping event e_1 to mark the end of an in-progress state of the process e' in the actual situation. By itself, this axiom suffices to make (4.145a) entail *Jack ran along the river*, while not making (4.145b) entail *Jack ran to the bridge*:

(4.145) (a) Jack stopped running along the river.

(b) Jack stopped running to the bridge.

As in (4.141), this difference is due to the introduction of gr only in the translation of (4.145b).

To finish up, I will treat finish as follows:

(4.146) $\forall e' . \forall e_1 . \mathsf{finish}(e', e_1) \rightarrow \left[\begin{array}{c} \exists e_a{}' . [\mathsf{comp}(e', e_a{}') \wedge \mathsf{max}(\tau_{\mathrm{t}}(e_a{}')) = \tau_{\mathrm{t}}(e_1)] \\ \bigwedge \\ \neg \exists e_a{}' . [\mathsf{comp}(e', e_a{}') \wedge \mathsf{max}(\tau_{\mathrm{t}}(e_a{}')) \succ_{\mathrm{t}} \tau_{\mathrm{t}}(e_1)] \end{array} \right]$

$$(4.147)\quad \forall e'.\,\forall e_1.\,\left[\begin{array}{l}[\mathsf{finish}(e',e_1)\ \wedge\ \mathsf{Actual}(e_1)]\ \rightarrow\\ \forall e_a.\,[\mathsf{comp}(e',e_a')\ \rightarrow\ \mathsf{Actual}(e_a')]\end{array}\right]$$

The first postulates require a finishing event e_1 to mark the completion of its associated process e'; the second one ensures that if e_1 is actual, then all the events making up the process e' will be actual as well, including the maximal one. For this reason, example (4.148a) below does entail (4.148b), as desired:

(4.148) (a) Jack finished running to the bridge.

(b) Jack ran to the bridge.

4.6 Aspectual Type Coercion

In the preceding analysis of the progressive and the aspectual verbs *start*, *stop*, and *finish*, we assigned two category-meaning pairs to each of these terms, one with the ground-from function gr and one without; as it turned out, however, only one of these was ever appropriate in a given circumstance.

At first glance, it certainly seems inelegant to have so much lexical duplication when we might just let gr be an aspectual type coercion operator, applying freely in the derivation. However, as we saw in section 2.3, there are occasions when these operators do not appear to be invoked. For example, consider the pair below:

(4.149) (a) Jack was pouring twenty liters of wort into the carboy.

(b) ? Jack poured twenty liters of wort into the carboy for ten seconds.

Whereas the progressive in (4.149a) unproblematically describes a state in which a partially realized event is in progress, (4.149b) does not seem to have a reading involving a subevent of duration ten seconds (which the function gr would make accessible).

While example (4.149) shows that at least some aspectual type coercion operators cannot be allowed to apply freely, the issue still remains as to whether this is the exception or the rule. For example, in section 2.6 we saw that the iterative coercion applies quite generally; as such, we might consider treating it as an optional VP-modifier, rather than as part of the lexical meaning of the progressive, aspectual verbs, temporal measure phrases, and so on. However, even this coercion appears to have its limits:

(4.150) (a) Jack ran to the bridge and back for an hour.

(b) ? Jack ran to the bridge and back for ten miles.

Curiously, (4.150b) resists the reading readily available in (4.150a) where Jack runs back and forth repeatedly.

Example (4.150) suggests that we should take the conservative route of explicitly identifying the cases in which aspectual coercion operators may apply. As we saw in section 2.3, Pustejovsky (1991a) proposes to do so by assigning a set of coercion operators to each expression. While this would solve the problem, he does not indicate how this set of operators should be compositionally specified. For this reason, I will continue to employ the perhaps less elegant solution of placing all of our aspectual type coercion operators in the lexicon, leaving open the issue of how best to capture the generalizations across the various category-meaning pairs assigned to each lexical item.

Let us now consider to the coercions we will need to round out the account, beginning with iteration. I will analyze iteration along the lines of *once, twice,* ..., which may be lexicalized as follows:

(4.151) (a) $\mathit{twice} \;\mapsto\; \mathsf{vp} \setminus \mathsf{vp} \;:\; \mathit{TWICE}$

(b) $\mathit{TWICE} \;\equiv\; \lambda P.\, \lambda x.\, \lambda e.\, \mathsf{plur}(P(x))(e) \wedge |e| = 2$

This specification maps a predicate true of events to one true of pluralities of such events having the appropriate cardinality.

Since the plural operator plur forms homogeneous predicates, we may also employ it to map predicates true of events to ones true of undelimited pluralities; this is shown below, using temporal *for*-adverbials as a case in point:

(4.152) (a) $\mathit{for} \;\mapsto\; \mathsf{vp} \setminus \mathsf{vp} / \mathsf{np} \;:\; \mathit{For}_{\mathrm{i}}$

(b) $\mathit{For}_{\mathrm{i}} \;\equiv\; \lambda d.\, \lambda P.\, \lambda x.\, \lambda e_a.\, \exists e.\, \left[\begin{array}{l} \mathsf{plur}(P(x))(e) \wedge \mathsf{comp}(e, e_a) \wedge \\ \rho(\tau_{\mathrm{t}}(e_a)) = d \end{array} \right]$

This iterative reading of *for*-adverbials may be used to derive the following translation of *Jack filled the carboy for ten minutes*:

(4.153) $\exists e_a.\, \exists e.\, \left[\begin{array}{l} \mathsf{plur}(\lambda e'.\, \mathsf{fill}(\mathsf{j}, \mathsf{the}(\mathsf{carboy}), e'))(e) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10) \wedge \\ \tau_{\mathrm{t}}(e_a) \prec_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array} \right]$

Because plur is homogeneous, (4.153) entails the following $\mathcal{L}_\sigma$-sentence:

(4.154) $\exists e_a.\, \left[\begin{array}{l} \mathsf{plur}(\lambda e'.\, \mathsf{fill}(\mathsf{j}, \mathsf{the}(\mathsf{carboy}), e'))(e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10) \wedge \\ \tau_{\mathrm{t}}(e_a) \prec_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array} \right]$

Formula (4.154) is true in models where there is an actual plurality e_a of events in which Jack fills the carboy such that e_a has duration ten minutes and takes place in the past. Note that since Actual is downward entailing, and since the intervals of the individual events must lie within the interval of the plurality, this reading of *Jack filled the carboy for ten minutes* does entail *Jack filled the carboy*, as desired.

Our next coercion is necessary to account for **preparatory-process** readings, which show up with *in*-adverbials. According to Moens and Steedman (1988),

'preparatory processes' are processes which lead up to an occurrence of a particular event. To capture this reading, I will introduce a relation prep-proc holding between a process and an event which finishes it, as well as a relation leads-to which identifies the maximal event of such a preparatory process:

(4.155) (a) $\forall e.\, \forall e_1.\, [\mathsf{prep\text{-}proc}(e, e_1) \rightarrow \mathsf{finish}(e, e_1)]$

(b) $\forall e_a.\, \forall e_1.\, [\mathsf{leads\text{-}to}(e_a, e_1) \rightarrow \exists e.\, [\mathsf{prep\text{-}proc}(e, e_1) \wedge e_a = \mathsf{max\text{-}comp}(e)]]$

These relations enable us to lexicalize the preparatory-process coercion as follows:

(4.156) (a) $in \;\mapsto\; \mathsf{vp} \setminus \mathsf{vp} / \mathsf{np} \;:\; In_{pp}$

(b) $In_{pp} \;\equiv\; \lambda d.\, \lambda P.\, \lambda x.\, \lambda e_a.\, \exists e_1.\, \left[\begin{array}{l} P(x)(e_1) \wedge \mathsf{leads\text{-}to}(e_a, e_1) \wedge \\ \rho(\tau_{\mathrm{t}}(e_a)) \preceq_{\mathrm{a}} d \end{array}\right]$

Using this entry for *in*, our grammar now yields the following translation for *The wort reached the top in ten seconds*:

(4.157) $\exists e_a.\, \exists e_1.\, \left[\begin{array}{l} \mathsf{reach}(\mathsf{the}(\mathsf{wort}), \mathsf{the}(\mathsf{top}), e_1) \wedge \\ \mathsf{leads\text{-}to}(e_a, e_1) \wedge \rho(\tau_{\mathrm{t}}(e_a)) \preceq_{\mathrm{a}} \mathsf{seconds}(10) \wedge \\ \tau_{\mathrm{t}}(e_a) \prec_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]$

This translation specifies the duration of the event e_a leading up to the reaching event e_1 to be twenty seconds, where e_a is the maximal element of the preparatory process e. Since identifying this process e is a highly context-dependent matter, I will leave open the question of how to further specify the semantics of leads-to.

Next we turn to inchoative (or onset) readings, which show up with *at*-adverbials and the present tense. In the case of processes and protracted events, the inchoative readings correspond to adding the meaning of *start* as a coercion:

(4.158) (a)
$$\begin{array}{lcl} at & \mapsto & \mathsf{vp} \setminus \mathsf{vp} / \mathsf{np} \;:\; At_1 \\ at & \mapsto & \mathsf{vp} \setminus \mathsf{vp} / \mathsf{np} \;:\; At_2 \\ that\ moment & \mapsto & \mathsf{np} \;:\; \mathsf{t} \end{array}$$

(b)
$$\begin{array}{lcl} At_1 & \equiv & \lambda t.\, \lambda P.\, \lambda x.\, \lambda e_0.\, \exists e'.\, P(x)(e') \wedge \mathsf{start}(e', e_0) \wedge \mathsf{at}(e_0, t) \\ At_2 & \equiv & \lambda t.\, \lambda P.\, \lambda x.\, \lambda e_0.\, \exists e'.\, P(x)(e') \wedge \mathsf{start}(\mathsf{gr}(e'), e_0) \wedge \mathsf{at}(e_0, t) \end{array}$$

With these additional lexical assignments, our grammar yields the following translations for *Jack ran at that moment* and *Jack ran to the museum at that moment*, respectively:[41]

(4.159) (a) $\exists e_0.\, \exists e'.\, \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e') \wedge \mathsf{start}(e', e_0) \wedge \\ \mathsf{at}(e_0, \mathsf{t}) \wedge \tau_{\mathrm{t}}(e_0) \prec_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_0) \end{array}\right]$

[41] As discussed in section 2.6, onset reading with accomplishments are somewhat unnatural; to account for this, restrictions could be placed on AT_2, or it could be left out of the lexicon entirely.

(b) $\exists e.\, \exists e'.\, \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e') \wedge \mathsf{to}(\mathsf{the}(\mathsf{museum}), e') \wedge \mathsf{start}(\mathsf{gr}(e'), e_0) \wedge \\ \mathsf{at}(e_0, \mathsf{t}) \wedge \tau_{\mathrm{t}}(e_0) \prec_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_0) \end{array} \right]$

Note that because of the sortal requirements imposed by at and start, these are the only possibilities which are well-sorted.

Turning now to states, here inchoative readings correspond to the insertion of the locution *come to*—witness the synonymy of *Jack knew the answer at that moment* (under this reading) and *Jack came to know the answer at that moment.* To lexicalize this reading, I will add an event beginning the state as a coercion:

(4.160) (a) $at \;\mapsto\; \mathsf{vp} \setminus \mathsf{vp} / \mathsf{np} \;:\; At_3$

(b) $At_3 \;\equiv\; \lambda t.\, \lambda P.\, \lambda x.\, \lambda e_0.\, \exists e.\, P(x)(e) \wedge \mathsf{begin}(e, e_0) \wedge \mathsf{at}(e_0, t)$

This assignment yields the following translation for *Jack understood the answer at that moment*:

(4.161) $\exists e_0.\, \exists e.\, \left[\begin{array}{l} \mathsf{understand}(\mathsf{j}, \mathsf{the}(\mathsf{answer}), e) \wedge \mathsf{begin}(e, e_0) \wedge \\ \mathsf{at}(e_0, \mathsf{t}) \wedge \tau_{\mathrm{t}}(e_0) \prec_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_0) \end{array} \right]$

Note that in this case the inchoative reading is not forced, since the normal stative reading is also a possibility.[42]

Finally, we return to the progressive to discuss its various futurate readings. As we noted in section 2.6, the progressive does combine with achievement expressions, but not under its normal in-progress reading:

(4.162) (a) Jack was winning the race.

(b) The wort was reaching the top.

(c) Jack was leaving the following week.

Interestingly, Moens and Steedman (1988) suggest that the progressive's in-progress and futurate readings may be given a uniform analysis in terms of preparatory processes, once one adds aspectual type coercions to the grammar.[43] To take (4.162a) as a case in point, they suggest that since the progressive requires a process as input, it must first coerce the event of Jack winning the race to the process leading up to this event; only then may it map this process to the state in which it is in progress, as usual.

Upon further inspection, however, it does not seem that Moens and Steedman's suggestion can account for all the vagaries of the progressive.[44] The first

[42] It is not entirely clear whether this coercion is fully general, especially in the case of stative expressions. An alternative one might consider pursuing is to let the stative-inchoative ambiguity reside in the lexical entries of the verbs which readily exhibit this meaning.

[43] Cf. (Kent, 1993) for a further formalization of this idea.

[44] Cf. (Binnick, 1991, pp. 281–290) for an excellent review of the controversial literature the progressive has generated; cf. also (Goldsmith and Woisetschlaeger, 1982) for an interesting viewpoint on the progressive which has been largely ignored in the recent literature.

problem we will consider concerns the time at which the progressive is allowed to apply. In the case of (4.162a), let us suppose that Jack is first off the line and thus succeeds in taking the initial lead; then it makes sense to say that *Jack was winning the race* speaking of that moment, even if Jack goes on to crash in the final lap. This situation may be contrasted with that of (4.162b): here let us suppose that Jack has just started filling the carboy with wort; in this case, even if Jack goes on to finish filling it as planned, then we still may not sensibly say that *The wort was reaching the top* referring to that moment; instead it appears that this expression is restricted to the final moments in which this process is in progress.

The second problem we will consider concerns the 'scheduled event' reading of the progressive evident in (4.162c). Here the most natural reading is one where Jack is scheduled to leave the week after the time referred to; under this reading, the sentence remains true even if Jack changes his mind (or otherwise subverts the schedule) and leaves the next day. This reading is clearly not the same as the other two: in the case of (4.162a), we may note that *? Jack was winning the race three minutes later* is odd unless the outcome was fixed beforehand; in the case of (4.162c), we may likewise observe that *? The wort was reaching the top thirty seconds later* is quite odd, since such an event is not normally one to be so precisely scheduled.

Since these three futurate readings do not seem to have a uniform analysis, I will simply lexicalize them using distinct constants, as shown below:

(4.163) (a)
$$\begin{array}{lcll} \textit{*prog*} & \mapsto & \mathsf{vp}\,/\,\mathsf{vp} & : \; PROG_a \\ \textit{*prog*} & \mapsto & \mathsf{vp}\,/\,\mathsf{vp} & : \; PROG_b \\ \textit{*prog*} & \mapsto & \mathsf{vp}\,/\,\mathsf{vp} & : \; PROG_c \end{array}$$

(b)
$$\begin{array}{lcl} PROG_a & \equiv & \lambda P.\,\lambda x.\,\lambda e.\,\mathsf{fut}_a(P(x),e) \\ PROG_b & \equiv & \lambda P.\,\lambda x.\,\lambda e.\,\mathsf{fut}_b(P(x),e) \\ PROG_c & \equiv & \lambda P.\,\lambda x.\,\lambda e.\,\mathsf{fut}_c(P(x),e) \end{array}$$

We will return to these readings when we discuss the scope ambiguities that arise when the progressive appears with temporal adverbials.

4.7 The Syntactic Desiderata Revisited

4.7.1 *For*-Adverbials

For-adverbials, as measure phrases, introduce the composed-of mapping comp into the derivation. This restricts the eventuality predicates they can modify to ones sorted for undelimited eventualities, i.e. canonical states, processes, or their plural counterparts.

Our first set of acceptable examples involve activity expressions, i.e. expressions which denote predicates sorted for processes or their plural counterparts:

(4.164) (a) Jack poured wort into the carboy for ten seconds.

(b) Jack filled carboys with wort for ten minutes.

(c) Jack ran along the river for ten minutes.

Their respective translations appear below:

(4.165) (a) $\exists e_a.\, \exists e.\, \exists x.\, \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{pour}(\mathsf{j}, x, e) \wedge \mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_{\mathrm{s}}(e)) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{seconds}(10) \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]$

(b) $\exists e_a.\, \exists e.\, \exists x.\, \left[\begin{array}{l} \mathsf{plur}(\mathsf{carboy})(x) \wedge \\ \mathsf{distr}\,([\lambda x.\, \lambda e.\, \mathsf{fill\text{-}with}(\mathsf{j}, x, \mu(\mathsf{wort}), e)], x, e) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10) \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]$

(c) $\exists e_a.\, \exists e.\, \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e) \wedge \mathsf{along}(\mathsf{the}(\mathsf{river}), e) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10) \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]$

In (4.165a), the eventuality e turns out to be a process because of the lexical semantics of *pour* together with the present treatment of mass terms. In (4.165b), e turns out to be the plural counterpart of a process, since the plural operator plur forms homogenous predicates over pluralities and the distributive relation distr forms an incremental thematic relation over pluralities. Finally, in (4.165c), e turns out be a process because of the lexical semantics of *run* and *along*.

Our next set of acceptable examples consists of the following stative expressions, together with their respective translations:

(4.166) (a) The carboy was full for ten minutes.

(b) Jack didn't fill the carboy for ten minutes.

(c) Jack was filling the carboy for ten seconds.

(4.167) (a) $\exists e_a.\, \exists e.\, \left[\begin{array}{l} \mathsf{full}(\mathsf{the}(\mathsf{carboy}), e) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10) \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]$

(b) $\exists e_a.\, \exists e.\, \left[\begin{array}{l} \mathsf{neg}(\lambda e'.\, \mathsf{fill}(\mathsf{j}, \mathsf{the}(\mathsf{carboy}), e'))(e) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10) \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]$

(c) $\exists e_a.\, \exists e.\, \exists e'.\, \left[\begin{array}{l} \mathsf{fill}(\mathsf{j}, \mathsf{the}(\mathsf{carboy}), e') \wedge \mathsf{inprog}(\mathsf{gr}(e'), e) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{seconds}(10) \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]$

In (4.167a), the sort axioms for the lexical predicate full allow e to be a canonical state. Likewise, in (4.167b) and (4.167c), the sort axioms for our negation operator neg and the in-progress relation inprog again allow e to be a canonical state.

When *for*-adverbials are combined with accomplishment expressions, i.e. expressions whose translations are defined only for protracted events or pluralities of such events, the resulting translations are not compatible with the

sortal requirements imposed by the composed-of relation comp (unless iteration is assumed). For this reason, these sentences turn out anomalous.

Our first set of anomalous examples is shown below, followed by their respective translations:

(4.168) (a) * Jack poured twenty liters of wort into the carboy for ten seconds.

(b) * Jack filled twenty carboys with wort for ten minutes.

(b′) * Jack filled the carboy with wort for ten seconds.

(c) * Jack ran to the museum for ten minutes.

(c′) * Jack ran two miles for ten minutes.

(4.169) (a) $* \exists e_a.\, \exists e.\, \exists x_a.\, \exists x.$ $\left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{comp}(x, x_a) \wedge \rho(x_a) = \mathsf{liters}(20) \wedge \\ \mathsf{pour}(\mathsf{j}, x_a, e) \wedge \mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_{\mathrm{s}}(e)) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{seconds}(10) \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]$

(b) $* \exists e_a.\, \exists e.\, \exists x_a.\, \exists x.$ $\left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{plur}(\mathsf{carboy})(x_a) \wedge |x_a| = 20 \wedge \\ \mathsf{distr}\left([\lambda y.\, \lambda e.\, \mathsf{fill\text{-}with}(\mathsf{j}, y, x, e)], x_a, e\right) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10) \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]$

(b′) $* \exists e_a.\, \exists e.\, \exists x.$ $\left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{fill\text{-}with}(\mathsf{j}, \mathsf{the}(\mathsf{carboy}), x, e) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{seconds}(10) \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]$

(c) $* \exists e_a.\, \exists e.$ $\left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e) \wedge \mathsf{to}(\mathsf{the}(\mathsf{museum}), e) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10) \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]$

(c′) $* \exists e_a.\, \exists e.$ $\left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e) \wedge \rho(\tau_{\mathrm{s}}(e)) = \mathsf{miles}(2) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10) \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]$

In (4.169a) and (4.169b), the requirement that the second and third arguments of pour and distr agree on delimitedness forces e to be an event. In (4.169b′), fill-with is assumed to be sorted for protracted events only. Finally, in (4.169c) and (4.169c′), though run is compatible with e being a process, the sort axioms on the lexical predicate to and the amount function ρ (together with the temporal trace function τ_{t}) force e to be an event.

Our next set of anomalous examples consists of the following non-individuating accomplishment expressions, together with their respective translations:

(4.170) (a) * Jack poured some amount of wort into the carboy for ten seconds.

(b) * Jack filled some number of carboys with wort for ten minutes.

(c) * Jack ran somewhere for ten minutes.

(c′) * Jack ran some distance for ten minutes.

(4.171) (a) $* \exists e_a.\, \exists e.\, \exists x_a.\, \exists x.\, \exists \alpha.$ $\left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{comp}(x, x_a) \wedge \\ \rho(x_a) = \alpha \wedge \mathsf{Amount}(\alpha) \wedge \\ \mathsf{pour}(\mathsf{j}, x_a, e) \wedge \mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_{\mathrm{s}}(e)) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{seconds}(10) \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]$

(b) $*\ \exists e_a.\,\exists e.\,\exists x_a.\,\exists x.\,\exists n.\ \left[\begin{array}{l} \mathsf{wort}(x)\ \wedge\ \mathsf{plur}(\mathsf{carboy})(x_a)\ \wedge \\ |\,x_a\,| = n\ \wedge\ \mathsf{Number}(n)\ \wedge \\ \mathsf{distr}\,([\lambda y.\,\lambda e.\,\mathsf{fill\text{-}with}(\mathsf{j}, y, x, e)]\,, x_a, e)\ \wedge \\ \mathsf{comp}(e, e_a)\ \wedge\ \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10)\ \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now}\ \wedge\ \mathsf{Actual}(e_a) \end{array}\right]$

(c) $*\ \exists e_a.\,\exists e.\,\exists p.\ \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e)\ \wedge\ \mathsf{to}(p, e)\ \wedge\ \mathsf{Place}(p)\ \wedge \\ \mathsf{comp}(e, e_a)\ \wedge\ \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10)\ \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now}\ \wedge\ \mathsf{Actual}(e_a) \end{array}\right]$

(c′) $*\ \exists e_a.\,\exists e.\,\exists d.\ \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e)\ \wedge\ \rho(\tau_{\mathrm{s}}(e)) = d\ \wedge\ \mathsf{Distance}(d)\ \wedge \\ \mathsf{comp}(e, e_a)\ \wedge\ \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10)\ \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now}\ \wedge\ \mathsf{Actual}(e_a) \end{array}\right]$

As before, in each of these examples the eventuality e is forced to be an event. Because the present account of their ungrammaticality does not rely on tests for homogeneous reference, the fact that these expressions fail to individuate is unproblematic.

As noted above, *for*-adverbials do cooccur with accomplishment expressions under iterated interpretations; the same is also true for achievement expressions, i.e. expressions defined for momentaneous events. Using the iterated translation of the *for*-adverbials, our grammar does yield acceptable translations for the following sentences:

(4.172)
(a) Jack poured twenty liters of wort into the carboy for forty minutes.

(b) Jack filled the carboy with wort for forty minutes.

(c) Jack ran to the museum for forty minutes.

(d) Jack winked for ten seconds.

These translations appear below:

(4.173)
(a) $\exists e_a.\,\exists e.\ \left[\begin{array}{l} \mathsf{plur}(P)(e)\ \wedge\ \mathsf{comp}(e, e_a)\ \wedge\ \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(40)\ \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now}\ \wedge\ \mathsf{Actual}(e_a) \end{array}\right]$,
where $P \equiv \lambda e'.\,\exists x_a.\,\exists x.\ \left[\begin{array}{l} \mathsf{wort}(x)\ \wedge\ \mathsf{comp}(x, x_a)\ \wedge\ \rho(x_a) = \mathsf{liters}(20)\ \wedge \\ \mathsf{pour}(\mathsf{j}, x_a, e')\ \wedge\ \mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_{\mathrm{s}}(e')) \end{array}\right]$

(b) $\exists e_a.\,\exists e.\ \left[\begin{array}{l} \mathsf{plur}(P)(e)\ \wedge\ \mathsf{comp}(e, e_a)\ \wedge\ \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(40)\ \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now}\ \wedge\ \mathsf{Actual}(e_a) \end{array}\right]$,
where $P \equiv \lambda e'.\,\exists x.\,\mathsf{fill\text{-}with}(\mathsf{j}, \mathsf{the}(\mathsf{carboy}), x, e')$

(c) $\exists e_a.\,\exists e.\ \left[\begin{array}{l} \mathsf{plur}(P)(e)\ \wedge\ \mathsf{comp}(e, e_a)\ \wedge\ \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(40)\ \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now}\ \wedge\ \mathsf{Actual}(e_a) \end{array}\right]$,
where $P \equiv \lambda e'.\,\mathsf{run}(\mathsf{j}, e')\ \wedge\ \mathsf{to}(\mathsf{the}(\mathsf{museum}), e')$

(d) $\exists e_a.\,\exists e.\ \left[\begin{array}{l} \mathsf{plur}(P)(e)\ \wedge\ \mathsf{comp}(e, e_a)\ \wedge\ \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{seconds}(10)\ \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now}\ \wedge\ \mathsf{Actual}(e_a) \end{array}\right]$,
where $P \equiv \lambda e'.\,\mathsf{wink}(\mathsf{j}, e')$

4.7.2 *In*-Adverbials

In-adverbials, in contrast to *for*-adverbials, do not introduce the composed-of mapping into the derivation. For this reason, *in*-adverbials are compatible with accomplishment expressions, even the non-individuating ones; this is shown below for the *in*-adverbial counterparts to the preceding ungrammatical examples:

(4.174) (a) Jack poured twenty liters of wort into the carboy in ten seconds.
(b) Jack filled twenty carboys with wort in ten minutes.
(b′) Jack filled the carboy with wort in ten seconds.
(c) Jack ran to the museum in ten minutes.
(c′) Jack ran two miles in ten minutes.

(4.175) (a) $\exists e_a.\,\exists x_a.\,\exists x.\left[\begin{array}{l}\mathsf{wort}(x)\ \wedge\ \mathsf{comp}(x,x_a)\ \wedge\ \rho(x_a)=\mathsf{liters}(20)\ \wedge\\ \mathsf{pour}(\mathsf{j},x_a,e_a)\ \wedge\ \mathsf{into}(\mathsf{the}(\mathsf{carboy}),\tau_{\mathrm{s}}(e_a))\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_a))\preceq_{\mathrm{a}}\mathsf{seconds}(10)\ \wedge\\ \tau_{\mathrm{t}}(e_a)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_a)\end{array}\right]$

(b) $\exists e_a.\,\exists x_a.\,\exists x.\left[\begin{array}{l}\mathsf{wort}(x)\ \wedge\ \mathsf{plur}(\mathsf{carboy})(x_a)\ \wedge\ |x_a|=20\ \wedge\\ \mathsf{distr}\left([\lambda y.\ \lambda e.\ \mathsf{fill\text{-}with}(\mathsf{j},y,x,e)],x_a,e_a\right)\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_a))\preceq_{\mathrm{a}}\mathsf{minutes}(10)\ \wedge\\ \tau_{\mathrm{t}}(e_a)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_a)\end{array}\right]$

(b′) $\exists e_a.\,\exists x.\left[\begin{array}{l}\mathsf{wort}(x)\ \wedge\ \mathsf{fill\text{-}with}(\mathsf{j},\mathsf{the}(\mathsf{carboy}),x,e_a)\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_a))\preceq_{\mathrm{a}}\mathsf{seconds}(10)\ \wedge\\ \tau_{\mathrm{t}}(e_a)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_a)\end{array}\right]$

(c) $\exists e_a.\left[\begin{array}{l}\mathsf{run}(\mathsf{j},e_a)\ \wedge\ \mathsf{to}(\mathsf{the}(\mathsf{museum}),e_a)\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_a))\preceq_{\mathrm{a}}\mathsf{minutes}(10)\ \wedge\\ \tau_{\mathrm{t}}(e_a)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_a)\end{array}\right]$

(c′) $\exists e_a.\left[\begin{array}{l}\mathsf{run}(\mathsf{j},e_a)\ \wedge\ \rho(\tau_{\mathrm{s}}(e_a))=\mathsf{miles}(2)\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_a))\preceq_{\mathrm{a}}\mathsf{minutes}(10)\ \wedge\\ \tau_{\mathrm{t}}(e_a)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_a)\end{array}\right]$

(4.176) (a) Jack poured some amount of wort into the carboy in ten seconds.
(b) Jack filled some number of carboys with wort in ten minutes.
(c) Jack ran somewhere in ten minutes.
(c′) Jack ran some distance in ten minutes.

(4.177) (a) $\exists e_a.\,\exists x_a.\,\exists x.\,\exists\alpha.\left[\begin{array}{l}\mathsf{wort}(x)\ \wedge\ \mathsf{comp}(x,x_a)\ \wedge\ \rho(x_a)=\alpha\ \wedge\ \mathsf{Amount}(\alpha)\ \wedge\\ \mathsf{pour}(\mathsf{j},x_a,e_a)\ \wedge\ \mathsf{into}(\mathsf{the}(\mathsf{carboy}),\tau_{\mathrm{s}}(e_a))\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_a))\preceq_{\mathrm{a}}\mathsf{seconds}(10)\ \wedge\\ \tau_{\mathrm{t}}(e_a)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_a)\end{array}\right]$

(b) $\exists e_a.\,\exists x_a.\,\exists x.\,\exists n.\left[\begin{array}{l}\mathsf{wort}(x)\ \wedge\ \mathsf{plur}(\mathsf{carboy})(x_a)\ \wedge\ |x_a|=n\ \wedge\ \mathsf{Number}(n)\ \wedge\\ \mathsf{distr}\left([\lambda y.\ \lambda e.\ \mathsf{fill\text{-}with}(\mathsf{j},y,x,e)],x_a,e_a\right)\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_a))\preceq_{\mathrm{a}}\mathsf{minutes}(10)\ \wedge\\ \tau_{\mathrm{t}}(e_a)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_a)\end{array}\right]$

(c) $\exists e_a.\,\exists p.\left[\begin{array}{l}\mathsf{run}(\mathsf{j},e_a)\ \wedge\ \mathsf{to}(p,e_a)\ \wedge\ \mathsf{Place}(p)\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_a))\preceq_{\mathrm{a}}\mathsf{minutes}(10)\ \wedge\\ \tau_{\mathrm{t}}(e_a)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_a)\end{array}\right]$

(c′) $\exists e_a.\,\exists d.\left[\begin{array}{l}\mathsf{run}(\mathsf{j},e_a)\ \wedge\ \rho(\tau_{\mathrm{s}}(e))=d\ \wedge\ \mathsf{Distance}(d)\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_a))\preceq_{\mathrm{a}}\mathsf{minutes}(10)\ \wedge\\ \tau_{\mathrm{t}}(e_a)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_a)\end{array}\right]$

In each of these examples, e_a is forced to be a protracted event (or its plural counterpart) as before, and thus compatible with the durational restriction provided by the *in*-adverbial.

Again in contrast to *for*-adverbials, *in*-adverbials are not ordinarily compatible with activity expressions, unless there is some contextually understood amount, quantity or distance:

(4.178) (a) * Jack poured wort into the carboy in ten seconds.

(b) * Jack filled carboys with wort in ten minutes.

(c) * Jack ran along the river in ten minutes.

(4.179) (a) (This time) Jack poured wort into the carboy in ten seconds.

(b) (This time) Jack filled carboys with wort in ten minutes.

(c) (This time) Jack ran along the river in ten minutes.

Although the present theory does yield appropriate translations for the latter sentences, it cannot explain this intuitive contrast. This should not be surprising, however, since a proper account of this difference would seem to rely on notions of discourse context not present in ordinary truth-conditional semantics.

Finally, *in*-adverbials are unlike *for*-adverbials in combining with achievement expressions under preparatory-process readings. Using the preparatory-process translation of the *in*-adverbials, our grammar yields the translations shown below for the following sentences:

(4.180) (a) Jack won the race in thirty minutes.

(b) The wort reached the top in ten seconds.

(4.181) (a) $\exists e_a . \exists e_1 . \left[\begin{array}{l} \mathsf{win}(\mathsf{j}, \mathsf{the}(\mathsf{race}), e_1) \wedge \\ \mathsf{leads\text{-}to}(e_a, e_1) \wedge \rho(\tau_{\mathrm{t}}(e_a)) \preceq_{\mathrm{a}} \mathsf{minutes}(30) \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array} \right]$

(b) $\exists e_a . \exists e_1 . \left[\begin{array}{l} \mathsf{reach}(\mathsf{the}(\mathsf{wort}), \mathsf{the}(\mathsf{top}), e_1) \wedge \\ \mathsf{leads\text{-}to}(e_a, e_1) \wedge \rho(\tau_{\mathrm{t}}(e_a)) \preceq_{\mathrm{a}} \mathsf{seconds}(10) \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array} \right]$

These are the only readings derived for these sentences, since durations are not defined for momentaneous events.

4.7.3 *At*-Adverbials

At-adverbials equate the temporal trace of an eventuality with a particular instant. As such, they are restricted to the zero-dimensional eventualities, i.e. the moments and the momentaneous events. In Vendlerian terms, *at*-adverbials normally combine with stative expressions and achievement expressions:

(4.182) (a) The carboy was full at that moment.

(b) Jack winked at that moment.

(4.183) (a) $\exists e.\ \left[\begin{array}{l} \mathsf{full}(\mathsf{the}(\mathsf{carboy}), e) \wedge \mathsf{at}(e, \mathsf{t}) \wedge \\ \tau_{\mathsf{t}}(e) \preceq_{\mathsf{i}} \mathsf{now} \wedge \mathsf{Actual}(e) \end{array}\right]$

(b) $\exists e.\ \left[\begin{array}{l} \mathsf{wink}(\mathsf{j}, e) \wedge \mathsf{at}(e, \mathsf{t}) \wedge \\ \tau_{\mathsf{t}}(e) \preceq_{\mathsf{i}} \mathsf{now} \wedge \mathsf{Actual}(e) \end{array}\right]$

At-adverbials may also combine with activity and accomplishment expressions under inchoative (onset) readings; these are derived using the lexical entry for *at* that includes the predicate start as a coercion:[45]

(4.184) (a) Jack ran at that moment.

(b) ? Jack ran to the museum at that moment.

(4.185) (a) $\exists e_0.\, \exists e'.\ \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e') \wedge \mathsf{start}(e', e_0) \wedge \\ \mathsf{at}(e_0, \mathsf{t}) \wedge \tau_{\mathsf{t}}(e_0) \preceq_{\mathsf{i}} \mathsf{now} \wedge \mathsf{Actual}(e_0) \end{array}\right]$

(b) $\exists e.\, \exists e'.\ \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e') \wedge \mathsf{to}(\mathsf{the}(\mathsf{museum}), e') \wedge \mathsf{start}(\mathsf{gr}(e'), e_0) \wedge \\ \mathsf{at}(e_0, \mathsf{t}) \wedge \tau_{\mathsf{t}}(e_0) \preceq_{\mathsf{i}} \mathsf{now} \wedge \mathsf{Actual}(e_0) \end{array}\right]$

Because of the sortal requirements imposed by at and start, these are the only possibilities which are well-sorted.

Finally, to account for the inchoative readings of stative expressions, we use the lexical entry for *at* which adds the predicate begin as a coercion:

(4.186) (a) Jack understood the answer at that moment.

(b) Jack knew the results at that moment.

(4.187) (a) $\exists e_0.\, \exists e.\ \left[\begin{array}{l} \mathsf{understand}(\mathsf{j}, \mathsf{the}(\mathsf{answer}), e) \wedge \mathsf{begin}(e, e_0) \wedge \\ \mathsf{at}(e_0, \mathsf{t}) \wedge \tau_{\mathsf{t}}(e_0) \preceq_{\mathsf{i}} \mathsf{now} \wedge \mathsf{Actual}(e_0) \end{array}\right]$

(b) $\exists e_0.\, \exists e.\ \left[\begin{array}{l} \mathsf{know}(\mathsf{j}, \mathsf{the}(\mathsf{plur}(\mathsf{result})), e) \wedge \mathsf{begin}(e, e_0) \wedge \\ \mathsf{at}(e_0, \mathsf{t}) \wedge \tau_{\mathsf{t}}(e_0) \preceq_{\mathsf{i}} \mathsf{now} \wedge \mathsf{Actual}(e_0) \end{array}\right]$

In contrast to the previous case, here the inchoative reading is not forced, since the normal stative reading is also a possibility. Note that this might explain why these readings are predominant with verbs such as *know* and *understand* which Vendler deemed to have secondary achievement uses.[46]

4.7.4 Present Tense

The present tense tense is treated analogously to *at*-adverbials, the only difference being that it specifies the time of an eventuality to be the present moment. For this reason, the present tense may combine with both stative and achievement expressions under its normal interpretation:

(4.188) (a) The carboy is full.

(b) (Now) Jack winks.

(4.189) (a) $\exists e.\ \left[\ \mathsf{full}(\mathsf{the}(\mathsf{carboy}), e) \wedge \tau_{\mathsf{t}}(e) = \mathsf{now} \wedge \mathsf{Actual}(e)\ \right]$

[45] Note that the coercion for the somewhat degraded accomplishment case could easily be left out.

[46] As mentioned in the last section, an alternative one might consider pursuing is to let the stative-inchoative ambiguity reside in the lexical entries of the verbs in question.

(b) $\exists e.\ \left[\ \mathsf{wink}(\mathsf{j}, e)\ \wedge\ \tau_{\mathrm{t}}(e) = \mathsf{now}\ \wedge\ \mathsf{Actual}(e)\ \right]$

As noted in section 2.6, achievement expressions usually cooccur with the present tense under habitual readings, which are not treated here; the reading shown above is perhaps the more unusual one, restricted to reports of ongoing events. Similarly, activity and accomplishment expressions usually cooccur with the present tense under habitual readings; however, in reporting contexts, these expressions also exhibit the inchoative readings shown below:

(4.190) (a) (Now) Jack runs.

(b) (Now) Jack runs to the museum.

(4.191) (a) $\exists e_0.\, \exists e'.\ \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e')\ \wedge\ \mathsf{start}(e', e_0) \wedge \\ \tau_{\mathrm{t}}(e_0) = \mathsf{now}\ \wedge\ \mathsf{Actual}(e_0) \end{array}\right]$

(b) $\exists e.\, \exists e'.\ \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e')\ \wedge\ \mathsf{to}(\mathsf{the}(\mathsf{museum}), e')\ \wedge\ \mathsf{start}(\mathsf{gr}(e'), e_0) \wedge \\ \tau_{\mathrm{t}}(e_0) = \mathsf{now}\ \wedge\ \mathsf{Actual}(e_0) \end{array}\right]$

Just as in the case of *at*-adverbials, the predicate start is added as a coercion to the meaning of the present tense morpheme.

4.7.5 The Progressive

The progressive cooccurs with activity and accomplishment expressions under its normal in-progress reading. As noted in section 2.6, it does not generally cooccur with stative expressions, though there are exceptional readings we will not discuss. With achievement expressions, the progressive displays a variety of futurate readings:

(4.192) (a) Jack was winning the race.

(b) The wort was reaching the top.

(4.193) (a) $\exists e.\ \left[\begin{array}{l} \mathsf{fut}_a([\lambda e_1.\, \mathsf{win}(\mathsf{j}, \mathsf{the}(\mathsf{race}), e_1)], e) \wedge \\ \tau_{\mathrm{t}}(e) \preceq_{\mathrm{i}} \mathsf{now}\ \wedge\ \mathsf{Actual}(e) \end{array}\right]$

(b) $\exists e.\ \left[\begin{array}{l} \mathsf{fut}_b([\lambda e_1.\, \mathsf{reach}(\mathsf{the}(\mathsf{wort}), \mathsf{the}(\mathsf{top}), e_1)], e) \wedge \\ \tau_{\mathrm{t}}(e) \preceq_{\mathrm{i}} \mathsf{now}\ \wedge\ \mathsf{Actual}(e) \end{array}\right]$

In section 4.6, we observed that the various futurate readings of the progressive do not appear to have a uniform analysis, contra Moens and Steedman (1988). For this reason, the present account simply translates them using distinct nonlogical constants, as shown above.[47]

The progressive also cooccurs with achievement expressions under iterative readings:

(4.194) (a) Jack was winning the race (each time).

(b) The wort was reaching the top (each time).

(4.195) (a) $\exists e.\, \exists e'.\ \left[\begin{array}{l} \mathsf{plur}([\lambda e_1.\, \mathsf{win}(\mathsf{j}, \mathsf{the}(\mathsf{race}), e_1)])(e')\ \wedge\ \mathsf{inprog}(e', e) \wedge \\ \tau_{\mathrm{t}}(e) \preceq_{\mathrm{i}} \mathsf{now}\ \wedge\ \mathsf{Actual}(e) \end{array}\right]$

[47] Note that this treatment is too simplistic to tell us anything about when the various futurate readings are available or appropriate.

(b) $\exists e.\, \exists e'.\, \left[\begin{array}{l} \mathsf{plur}([\lambda e_1.\, \mathsf{reach}(\mathsf{the}(\mathsf{wort}), \mathsf{the}(\mathsf{top}), e_1)])(e') \wedge \mathsf{inprog}(e', e) \wedge \\ \tau_{\mathrm{t}}(e) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e) \end{array} \right]$

These readings may be accounted for by simply adding the plural operator plur as a coercion in the in-progress reading.

Finally, the in-progress and futurate readings of the progressive lead to interesting ambiguities with temporal adverbials, as shown below:

(4.196) (a) Jack was running at noon (and had been for some time).

(b) Jack was running at noon (so he started loosening up).

(4.197) (a) $\exists e.\, \exists e'.\, \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e') \wedge \mathsf{inprog}(e', e) \wedge \mathsf{at}(e, \mathsf{noon}) \wedge \\ \tau_{\mathrm{t}}(e) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e) \end{array} \right]$

(b) $\exists e.\, \left[\begin{array}{l} \mathsf{fut}_c\left([\lambda e_0.\, \exists e'.\, \mathsf{run}(\mathsf{j}, e') \wedge \mathsf{start}(e', e_0) \wedge \mathsf{at}(e_0, \mathsf{noon})], e\right) \wedge \\ \tau_{\mathrm{t}}(e) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e) \end{array} \right]$

According to the present account, this ambiguity arises because the progressive and *at noon* may apply in either order: in (4.197a), the progressive applies to the VP *run*, then *at noon* applies to the VP **prog* run*; in (4.197b), on the other hand, *at noon* applies first to the VP *run*, then the progressive applies to the VP *run at noon*. Interestingly, note that since *at noon* applies at different points in the derivation in the two cases, the inchoative reading of the *at noon* is forced in the latter case only.[48]

4.7.6 Aspectual Verbs

Aspectual verbs are like the progressive in that they cooccur with activity and accomplishment expressions, possibly under iterative readings. Curiously, however, aspectual verbs do not seem to exhibit futurate readings:

(4.198) (a) ? Jack stopped winning the race.

(b) ? The wort stopped reaching the top.

Since aspectual type coercions are required to be lexicalized, the absence of futurate readings is unproblematic—we simply do not include preparatory-process or futurate coercions in the lexical entries for the aspectual verbs. Of course, in order to account for the iterative readings, we do include lexical entries adding the plural operator as a coercion:

(4.199) (a) Jack stopped winning the race (each time).

(b) The wort stopped reaching the top (each time).

(4.200) (a) $\exists e_1.\, \exists e'.\, \left[\begin{array}{l} \mathsf{plur}([\lambda e_1.\, \mathsf{win}(\mathsf{j}, \mathsf{the}(\mathsf{race}), e_1)])(e') \wedge \mathsf{stop}(e', e_1) \wedge \\ \tau_{\mathrm{t}}(e_1) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_1) \end{array} \right]$

(b) $\exists e_1.\, \exists e'.\, \left[\begin{array}{l} \mathsf{plur}([\lambda e_1.\, \mathsf{reach}(\mathsf{the}(\mathsf{wort}), \mathsf{the}(\mathsf{top}), e_1)])(e') \wedge \mathsf{stop}(e', e_1) \wedge \\ \tau_{\mathrm{t}}(e_1) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_1) \end{array} \right]$

[48] Note also that the progressive cannot apply to the VP *run at noon* under its in-progress reading, since inprog is not well-sorted with momentaneous events.

4.8 The Semantic Desiderata Revisited

4.8.1 Downward Entailments

We begin with the downward entailments of activity expressions:

(4.201) (a) $\models$ $\dfrac{\text{Jack poured wort into the carboy for ten seconds.}}{\text{Jack poured wort into the carboy for nine seconds.}}$

(b) $\models$ $\dfrac{\text{Jack filled carboys with wort for ten minutes.}}{\text{Jack filled carboys with wort for nine minutes.}}$

(c) $\models$ $\dfrac{\text{Jack ran along the river for ten minutes.}}{\text{Jack ran along the river for nine minutes.}}$

(4.202) (a) $\models$
$$\frac{\exists e_a.\, \exists e.\, \exists x.\, \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{pour}(\mathsf{j}, x, e) \wedge \mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_{\mathrm{s}}(e)) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{seconds}(10) \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]}{\exists e_b.\, \exists e.\, \exists x.\, \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{pour}(\mathsf{j}, x, e) \wedge \mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_{\mathrm{s}}(e)) \wedge \\ \mathsf{comp}(e, e_b) \wedge \rho(\tau_{\mathrm{t}}(e_b)) = \mathsf{seconds}(9) \wedge \\ \tau_{\mathrm{t}}(e_b) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_b) \end{array}\right]}$$

(b) $\models$
$$\frac{\exists e_a.\, \exists e.\, \exists y.\, \exists x.\, \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{plur}(\mathsf{carboy})(y) \wedge \\ \mathsf{distr}\left([\lambda y.\, \lambda e.\, \mathsf{fill\text{-}with}(\mathsf{j}, y, x, e)], y, e\right) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10) \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]}{\exists e_a.\, \exists e.\, \exists y.\, \exists x.\, \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{plur}(\mathsf{carboy})(y) \wedge \\ \mathsf{distr}\left([\lambda y.\, \lambda e.\, \mathsf{fill\text{-}with}(\mathsf{j}, y, x, e)], y, e\right) \wedge \\ \mathsf{comp}(e, e_b) \wedge \rho(\tau_{\mathrm{t}}(e_b)) = \mathsf{minutes}(9) \wedge \\ \tau_{\mathrm{t}}(e_b) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_b) \end{array}\right]}$$

(c) $\models$
$$\frac{\exists e_a.\, \exists e.\, \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e) \wedge \mathsf{along}(\mathsf{the}(\mathsf{river}), e) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10) \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]}{\exists e_b.\, \exists e.\, \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e) \wedge \mathsf{along}(\mathsf{the}(\mathsf{river}), e) \wedge \\ \mathsf{comp}(e, e_b) \wedge \rho(\tau_{\mathrm{t}}(e_b)) = \mathsf{minutes}(10) \wedge \\ \tau_{\mathrm{t}}(e_b) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_b) \end{array}\right]}$$

In each of these examples, the antecedent contains an event e_a composed of the process e. Because the composed-of relation comp encodes equivalence classes, we know that all the subevents of e_a must also be composed of e. In particular, we may assume that e_a has a proper subevent e_b of duration nine minutes (seconds). Since e_b is a subevent, we know that its temporal trace must be contained within that of e_a, and thus must also precede the instant now; moreover, since the special predicate Actual is preserved by subevents, e_b must also be an actual event.

Note that these entailments do not go through without a caveat concerning the minimal parts problem. In the case of (4.202a) and (4.202c), which involve a single event, our idealized condition on the existence of subevents of lesser durations applies. While this does guarantee that e_a has some subevent of duration nine minutes (seconds), it is of course only an idealization. In the case of (4.202b), which involves a plurality of events, we must simply assume that e_a has such a subevent.

The minimal parts problem does not arise when we turn to stative expressions, which are also downward entailing:

(4.203) (a) $\models$ $\dfrac{\text{The carboy was full for ten minutes.}}{\text{The carboy was full for nine minutes.}}$

(b) $\models$ $\dfrac{\text{Jack didn't fill the carboy for ten minutes.}}{\text{Jack didn't fill the carboy for nine minutes.}}$

(c) $\models$ $\dfrac{\text{Jack was filling the carboy for ten seconds.}}{\text{Jack was filling the carboy for nine seconds.}}$

(4.204) (a) $\models$
$$\frac{\exists e_a.\, \exists e.\ \left[\begin{array}{l}\mathsf{full}(\mathsf{the}(\mathsf{carboy}), e) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10) \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a)\end{array}\right]}{\exists e_b.\, \exists e.\ \left[\begin{array}{l}\mathsf{full}(\mathsf{the}(\mathsf{carboy}), e) \wedge \\ \mathsf{comp}(e, e_b) \wedge \rho(\tau_{\mathrm{t}}(e_b)) = \mathsf{minutes}(9) \wedge \\ \tau_{\mathrm{t}}(e_b) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_b)\end{array}\right]}$$

(b) $\models$
$$\frac{\exists e_a.\, \exists e.\ \left[\begin{array}{l}\mathsf{neg}(\lambda e'.\, \mathsf{fill}(\mathsf{j}, \mathsf{the}(\mathsf{carboy}), e'))(e) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{minutes}(10) \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a)\end{array}\right]}{\exists e_b.\, \exists e.\ \left[\begin{array}{l}\mathsf{neg}(\lambda e'.\, \mathsf{fill}(\mathsf{j}, \mathsf{the}(\mathsf{carboy}), e'))(e) \wedge \\ \mathsf{comp}(e, e_b) \wedge \rho(\tau_{\mathrm{t}}(e_b)) = \mathsf{minutes}(9) \\ \tau_{\mathrm{t}}(e_b) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_b)\end{array}\right]}$$

(c) $\models$
$$\frac{\exists e_a.\, \exists e.\, \exists e'.\ \left[\begin{array}{l}\mathsf{fill}(\mathsf{j}, \mathsf{the}(\mathsf{carboy}), e') \wedge \mathsf{inprog}(\mathsf{gr}(e'), e) \wedge \\ \mathsf{comp}(e, e_a) \wedge \rho(\tau_{\mathrm{t}}(e_a)) = \mathsf{seconds}(10) \wedge \\ \tau_{\mathrm{t}}(e_a) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a)\end{array}\right]}{\exists e_b.\, \exists e.\, \exists e'.\ \left[\begin{array}{l}\mathsf{fill}(\mathsf{j}, \mathsf{the}(\mathsf{carboy}), e') \wedge \mathsf{inprog}(\mathsf{gr}(e'), e) \wedge \\ \mathsf{comp}(e, e_b) \wedge \rho(\tau_{\mathrm{t}}(e_b)) = \mathsf{seconds}(9) \wedge \\ \tau_{\mathrm{t}}(e_b) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_b)\end{array}\right]}$$

These entailments go through roughly as before. In each of these examples, the antecedent contains a period e_a composed of the (canonical) state e. Because e_a is a period, it must have subparts of arbitrarily smaller size; in particular, it must have a subpart e_b of duration nine minutes (seconds). Since the composed-of relation comp is closed under subparts, we know that e_b is also composed of e; finally, we know that e_b must precede the present moment now and satisfy the predicate Actual for the same reasons as given previously.

Unlike activity expressions, accomplishment expressions (at least, the individuating ones) are not downward entailing:

(4.205) (a) $\not\models$ $\dfrac{\text{Jack poured twenty liters of wort into the carboy in ten seconds.}}{\text{Jack poured twenty liters of wort into the carboy in nine seconds.}}$

(b) $\not\models$ $\dfrac{\text{Jack filled twenty carboys with wort in ten minutes.}}{\text{Jack filled twenty carboys with wort in nine minutes.}}$

(b′) $\not\models$ $\dfrac{\text{Jack filled the carboy with wort in ten seconds.}}{\text{Jack filled the carboy with wort in nine seconds.}}$

(c) $\not\models$ $\dfrac{\text{Jack ran to the museum river in ten minutes.}}{\text{Jack ran to the museum in nine minutes.}}$

(c′) $\not\models$ $\dfrac{\text{Jack ran two miles in ten minutes.}}{\text{Jack ran two miles in nine minutes.}}$

(4.206)

(a) $\not\models$ $\dfrac{\exists e_a.\,\exists x_a.\,\exists x.\ \left[\begin{array}{l}\mathsf{wort}(x)\ \wedge\ \mathsf{comp}(x,x_a)\ \wedge\ \rho(x_a)=\mathsf{liters}(20)\ \wedge\\ \mathsf{pour}(\mathsf{j},x_a,e_a)\ \wedge\ \mathsf{into}(\mathsf{the}(\mathsf{carboy}),\tau_{\mathrm{s}}(e_a))\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_a))\preceq_{\mathrm{a}}\mathsf{seconds}(10)\ \wedge\\ \tau_{\mathrm{t}}(e_a)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_a)\end{array}\right]}{\exists e_b.\,\exists x_b.\,\exists x.\ \left[\begin{array}{l}\mathsf{wort}(x)\ \wedge\ \mathsf{comp}(x,x_b)\ \wedge\ \rho(x_b)=\mathsf{liters}(20)\ \wedge\\ \mathsf{pour}(\mathsf{j},x_b,e_b)\ \wedge\ \mathsf{into}(\mathsf{the}(\mathsf{carboy}),\tau_{\mathrm{s}}(e_b))\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_b))\preceq_{\mathrm{a}}\mathsf{seconds}(9)\ \wedge\\ \tau_{\mathrm{t}}(e_b)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_b)\end{array}\right]}$

(b) $\not\models$ $\dfrac{\exists e_a.\,\exists x_a.\,\exists x.\ \left[\begin{array}{l}\mathsf{wort}(x)\ \wedge\ \mathsf{plur}(\mathsf{carboy})(x_a)\ \wedge\ |x_a|=20\ \wedge\\ \mathsf{distr}\left([\lambda y.\,\lambda e.\,\mathsf{fill\text{-}with}(\mathsf{j},y,x,e)],x_a,e_a\right)\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_a))\preceq_{\mathrm{a}}\mathsf{minutes}(10)\ \wedge\\ \tau_{\mathrm{t}}(e_a)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_a)\end{array}\right]}{\exists e_b.\,\exists x_b.\,\exists x.\ \left[\begin{array}{l}\mathsf{wort}(x)\ \wedge\ \mathsf{plur}(\mathsf{carboy})(x_b)\ \wedge\ |x_b|=20\ \wedge\\ \mathsf{distr}\left([\lambda y.\,\lambda e.\,\mathsf{fill\text{-}with}(\mathsf{j},y,x,e)],x_b,e_b\right)\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_b))\preceq_{\mathrm{a}}\mathsf{minutes}(9)\ \wedge\\ \tau_{\mathrm{t}}(e_b)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_b)\end{array}\right]}$

(b′) $\not\models$ $\dfrac{\exists e_a.\,\exists x.\ \left[\begin{array}{l}\mathsf{wort}(x)\ \wedge\ \mathsf{fill\text{-}with}(\mathsf{j},\mathsf{the}(\mathsf{carboy}),x,e_a)\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_a))\preceq_{\mathrm{a}}\mathsf{seconds}(10)\ \wedge\\ \tau_{\mathrm{t}}(e_a)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_a)\end{array}\right]}{\exists e_b.\,\exists x.\ \left[\begin{array}{l}\mathsf{wort}(x)\ \wedge\ \mathsf{fill\text{-}with}(\mathsf{j},\mathsf{the}(\mathsf{carboy}),x,e_b)\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_b))\preceq_{\mathrm{a}}\mathsf{seconds}(9)\ \wedge\\ \tau_{\mathrm{t}}(e_b)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_b)\end{array}\right]}$

(c) $\not\models$ $\dfrac{\exists e_a.\ \left[\begin{array}{l}\mathsf{run}(\mathsf{j},e_a)\ \wedge\ \mathsf{to}(\mathsf{the}(\mathsf{museum}),e_a)\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_a))\preceq_{\mathrm{a}}\mathsf{minutes}(10)\ \wedge\\ \tau_{\mathrm{t}}(e_a)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_a)\end{array}\right]}{\exists e_b.\ \left[\begin{array}{l}\mathsf{run}(\mathsf{j},e_b)\ \wedge\ \mathsf{to}(\mathsf{the}(\mathsf{museum}),e_b)\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_b))\preceq_{\mathrm{a}}\mathsf{minutes}(9)\ \wedge\\ \tau_{\mathrm{t}}(e_b)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_b)\end{array}\right]}$

(c′) $\not\models$ $\dfrac{\exists e_a.\ \left[\begin{array}{l}\mathsf{run}(\mathsf{j},e_a)\ \wedge\ \rho(\tau_{\mathrm{s}}(e_a))=\mathsf{miles}(2)\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_a))\preceq_{\mathrm{a}}\mathsf{minutes}(10)\ \wedge\\ \tau_{\mathrm{t}}(e_a)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_a)\end{array}\right]}{\exists e_b.\ \left[\begin{array}{l}\mathsf{run}(\mathsf{j},e_b)\ \wedge\ \rho(\tau_{\mathrm{s}}(e_b))=\mathsf{miles}(2)\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_b))\preceq_{\mathrm{a}}\mathsf{minutes}(9)\ \wedge\\ \tau_{\mathrm{t}}(e_b)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_b)\end{array}\right]}$

In each of these examples, if we assume that the event e_a in the antecedent takes exactly ten minutes (seconds), then clearly e_a does not satisfy the consequent; moreover, if we look at proper subevents e_b of e_a taking nine minutes (seconds), nothing guarantees that these e_b satisfy the additional conjuncts of the consequent. For this reason, these entailments are not valid.

While most accomplishment expressions are not downward entailing, some of the non-individuating ones do turn out downward entailing:

(4.207)

(a) $\models$ $\dfrac{\text{Jack poured some amount of wort into the carboy in ten seconds.}}{\text{Jack poured some amount of wort into the carboy in nine seconds.}}$

(b) $\models$ $\dfrac{\text{Jack filled some carboys with wort in ten minutes.}}{\text{Jack filled some carboys with wort in nine minutes.}}$

(c) $\models$ Jack ran somewhere in ten minutes. / Jack ran somewhere in nine minutes.

(c′) $\models$ Jack ran some distance in ten minutes. / Jack ran some distance in nine minutes.

(4.208)

(a) $\models$

$$\frac{\exists e_a.\,\exists x_a.\,\exists x.\,\exists \alpha.\left[\begin{array}{l}\mathsf{wort}(x)\ \wedge\ \mathsf{comp}(x,x_a)\ \wedge\\ \rho(x_a)=\alpha\ \wedge\ \mathsf{Amount}(\alpha)\ \wedge\\ \mathsf{pour}(\mathsf{j},x_a,e_a)\ \wedge\ \mathsf{into}(\mathsf{the}(\mathsf{carboy}),\tau_{\mathrm{s}}(e_a))\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_a))\preceq_{\mathrm{a}}\mathsf{seconds}(10)\ \wedge\\ \tau_{\mathrm{t}}(e_a)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_a)\end{array}\right]}{\exists e_b.\,\exists x_b.\,\exists x.\,\exists \beta.\left[\begin{array}{l}\mathsf{wort}(x)\ \wedge\ \mathsf{comp}(x,x_b)\ \wedge\\ \rho(x_b)=\beta\ \wedge\ \mathsf{Amount}(\beta)\ \wedge\\ \mathsf{pour}(\mathsf{j},x_b,e_b)\ \wedge\ \mathsf{into}(\mathsf{the}(\mathsf{carboy}),\tau_{\mathrm{s}}(e_b))\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_b))\preceq_{\mathrm{a}}\mathsf{seconds}(9)\ \wedge\\ \tau_{\mathrm{t}}(e_b)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_b)\end{array}\right]}$$

(b) $\models$

$$\frac{\exists e_a.\,\exists x_a.\,\exists x.\,\exists n.\left[\begin{array}{l}\mathsf{wort}(x)\ \wedge\ \mathsf{plur}(\mathsf{carboy})(x_a)\ \wedge\\ |x_a|=n\ \wedge\ \mathsf{Number}(n)\ \wedge\\ \mathsf{distr}\left([\lambda y.\,\lambda e.\,\mathsf{fill\text{-}with}(\mathsf{j},y,x,e)],x_a,e_a\right)\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_a))\preceq_{\mathrm{a}}\mathsf{minutes}(10)\ \wedge\\ \tau_{\mathrm{t}}(e_a)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_a)\end{array}\right]}{\exists e_b.\,\exists x_b.\,\exists x.\,\exists m.\left[\begin{array}{l}\mathsf{wort}(x)\ \wedge\ \mathsf{plur}(\mathsf{carboy})(x_b)\ \wedge\\ |x_b|=m\ \wedge\ \mathsf{Number}(m)\ \wedge\\ \mathsf{distr}\left([\lambda y.\,\lambda e.\,\mathsf{fill\text{-}with}(\mathsf{j},y,x,e)],x_b,e_b\right)\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_b))\preceq_{\mathrm{a}}\mathsf{minutes}(9)\ \wedge\\ \tau_{\mathrm{t}}(e_b)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_b)\end{array}\right]}$$

(c) $\models$

$$\frac{\exists e_a.\,\exists p_a.\left[\begin{array}{l}\mathsf{run}(\mathsf{j},e_a)\ \wedge\ \mathsf{to}(p_a,e_a)\ \wedge\ \mathsf{Place}(p_a)\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_a))\preceq_{\mathrm{a}}\mathsf{minutes}(10)\ \wedge\\ \tau_{\mathrm{t}}(e_a)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_a)\end{array}\right]}{\exists e_b.\,\exists p_b.\left[\begin{array}{l}\mathsf{run}(\mathsf{j},e_b)\ \wedge\ \mathsf{to}(p_b,e_b)\ \wedge\ \mathsf{Place}(p_b)\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_b))\preceq_{\mathrm{a}}\mathsf{minutes}(9)\ \wedge\\ \tau_{\mathrm{t}}(e_b)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_b)\end{array}\right]}$$

(c′) $\models$

$$\frac{\exists e_a.\,\exists d_a.\left[\begin{array}{l}\mathsf{run}(\mathsf{j},e_a)\ \wedge\ \rho(\tau_{\mathrm{s}}(e))=d_a\ \wedge\ \mathsf{Distance}(d_a)\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_a))\preceq_{\mathrm{a}}\mathsf{minutes}(10)\ \wedge\\ \tau_{\mathrm{t}}(e_a)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_a)\end{array}\right]}{\exists e_b.\,\exists d_b.\left[\begin{array}{l}\mathsf{run}(\mathsf{j},e_b)\ \wedge\ \rho(\tau_{\mathrm{s}}(e))=d_b\ \wedge\ \mathsf{Distance}(d_b)\ \wedge\\ \rho(\tau_{\mathrm{t}}(e_b))\preceq_{\mathrm{a}}\mathsf{minutes}(9)\ \wedge\\ \tau_{\mathrm{t}}(e_b)\preceq_{\mathrm{i}}\mathsf{now}\ \wedge\ \mathsf{Actual}(e_b)\end{array}\right]}$$

In each of these examples, we may assume as before that the event e_a has a subevent e_b of duration nine minutes (seconds) which satisfies the last three conjuncts of the consequent. Since the quantity, number, end location or distance is allowed to vary from the antecedent to the consequent, the subevent e_b ends up satisfying the rest of the conjuncts in the consequent too. To see why, let us examine each case in turn:

(a) Since $\lambda x.\,\lambda e.\,\mathsf{pour}(\mathsf{j},x,e)$ is an incremental thematic relation, we know that there is a subpart x_b of x_a which makes $\mathsf{pour}(\mathsf{j},x_b,e_b)$ true. Since x_b is a subpart of x_a, we also know that $\mathsf{comp}(x,x_b)$. Because x_b is delimited, it has some quantity β associated with it, which may be less than α. Finally, with pouring events we assume the spatial trace is unchanging, and thus $\mathsf{into}(\mathsf{the}(\mathsf{carboy}),\tau_{\mathrm{s}}(e_b))$ trivially holds.

(b) Let $R \equiv [\lambda y.\, \lambda e.\, \mathsf{fill\text{-}with}(\mathsf{j}, y, x, e)]$. Since $\lambda x.\, \lambda e.\, \mathsf{distr}(R, x, e)$ is an incremental thematic relation, we know there is individual subpart x_b of x_a which makes $\mathsf{distr}(R, x_b, e_b)$ true. Because $\mathsf{plur}(\mathsf{carboy})$ is homogeneous, $\mathsf{plur}(\mathsf{carboy})(x_b)$ follows. Finally, since x_b is delimited, it must have some cardinality n.

(c) Because $\lambda e.\, \mathsf{run}(\mathsf{j}, e)$ is homogeneous, we know that $\mathsf{run}(\mathsf{j}, e_b)$ must be true. Since e_b is delimited, its spatial trace must have some end location p_b.

(c′) As in the previous case, we know $\mathsf{run}(\mathsf{j}, e_b)$ must be true; likewise, since e_b is delimited, we know its spatial trace must have some distance d_b.

Finally, we may observe that while accomplishment expressions are not (normally) downward entailing temporally, they are downward entailing on the relevant quantity or distance:

(4.209) (a) $\models$ $\dfrac{\text{Jack poured twenty liters of wort into the carboy.}}{\text{Jack poured ten liters of wort into the carboy.}}$

(b) $\models$ $\dfrac{\text{Jack filled twenty carboys with wort.}}{\text{Jack filled ten carboys with wort.}}$

(c′) $\models$ $\dfrac{\text{Jack ran two miles.}}{\text{Jack ran one mile.}}$

These entailments are very much like the previous ones, so we will omit the translations. In the first two cases, we have the same incremental thematic relations as before, which means that the x_b in the consequents determine appropriate subevents e_b; in the third case, we need only appeal to our idealized condition on the existence of subevents of lesser distances.

4.8.2 Existential Entailments

The relation between activity expressions and the accomplishment expressions derived from them by adding amount or destination phrases is further illuminated by the following logical equivalences:

(4.210) (a) $\models$ $\dfrac{\text{Jack poured wort into the carboy.}}{\text{Jack poured some amount of wort into the carboy.}}$

$\models$ $\dfrac{\text{Jack poured some amount of wort into the carboy.}}{\text{Jack poured wort into the carboy.}}$

(b) $\models$ $\dfrac{\text{Jack filled carboys with wort.}}{\text{Jack filled some carboys with wort.}}$

$\models$ $\dfrac{\text{Jack filled some carboys with wort.}}{\text{Jack filled carboys with wort.}}$

(c) $\models$ $\dfrac{\text{Jack ran.}}{\text{Jack ran somewhere.}}$

$\models$ $\dfrac{\text{Jack ran somewhere.}}{\text{Jack ran.}}$

(c′) $\models$ $\dfrac{\text{Jack ran.}}{\text{Jack ran some distance.}}$

$\models$ $\dfrac{\text{Jack ran some distance.}}{\text{Jack ran.}}$

The translations for these entailments are shown below (omitting the symmetric reverse entailments):

(4.211)

(a) $\models$ $\dfrac{\exists e_a.\, \exists e.\, \exists x.\, \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{pour}(\mathsf{j}, x, e) \wedge \mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_{\mathsf{s}}(e)) \wedge \\ \mathsf{comp}(e, e_a) \wedge \tau_{\mathsf{t}}(e_a) \preceq_{\mathsf{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]}{\exists e_a.\, \exists x_a.\, \exists x.\, \exists \alpha.\, \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{comp}(x, x_a) \wedge \\ \rho(x_a) = \alpha \wedge \mathsf{Amount}(\alpha) \wedge \\ \mathsf{pour}(\mathsf{j}, x_a, e_a) \wedge \mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_{\mathsf{s}}(e_a)) \wedge \\ \tau_{\mathsf{t}}(e_a) \preceq_{\mathsf{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]}$

(b) $\models$ $\dfrac{\exists e_a.\, \exists e.\, \exists y.\, \exists x.\, \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{plur}(\mathsf{carboy})(y) \wedge \\ \mathsf{distr}\left([\lambda y.\, \lambda e.\, \mathsf{fill\text{-}with}(\mathsf{j}, y, x, e)], y, e\right) \wedge \\ \mathsf{comp}(e, e_a) \wedge \tau_{\mathsf{t}}(e_a) \preceq_{\mathsf{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]}{\exists e_a.\, \exists y_a.\, \exists x.\, \exists n.\, \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{plur}(\mathsf{carboy})(y_a) \wedge \\ |y_a| = n \wedge \mathsf{Number}(n) \wedge \\ \mathsf{distr}\left([\lambda y.\, \lambda e.\, \mathsf{fill\text{-}with}(\mathsf{j}, y, x, e)], y_a, e_a\right) \wedge \\ \tau_{\mathsf{t}}(e_a) \preceq_{\mathsf{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]}$

(c) $\models$ $\dfrac{\exists e_a.\, \exists e.\, \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e) \wedge \mathsf{comp}(e, e_a) \wedge \\ \tau_{\mathsf{t}}(e_a) \preceq_{\mathsf{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]}{\exists e_a.\, \exists p_a.\, \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e_a) \wedge \mathsf{to}(p_a, e_a) \wedge \mathsf{Place}(p_a) \wedge \\ \tau_{\mathsf{t}}(e_a) \preceq_{\mathsf{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]}$

(c′) $\models$ $\dfrac{\exists e_a.\, \exists e.\, \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e) \wedge \mathsf{comp}(e, e_a) \wedge \\ \tau_{\mathsf{t}}(e_a) \preceq_{\mathsf{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]}{\exists e_a.\, \exists d_a.\, \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e_a) \wedge \rho(\tau_{\mathsf{s}}(e)) = d_a \wedge \mathsf{Distance}(d_a) \wedge \\ \tau_{\mathsf{t}}(e_a) \preceq_{\mathsf{i}} \mathsf{now} \wedge \mathsf{Actual}(e_a) \end{array}\right]}$

Let us examine each case in turn:

(a) First let us consider the top-to-bottom entailment. Since $\lambda x.\lambda e.\mathsf{pour}(\mathsf{j}, x, e)$ is an incremental thematic relation, we know that there is some x_a satisfying the conjunct $\mathsf{comp}(x, x_a)$ which also makes $\mathsf{pour}(\mathsf{j}, x_a, e_a)$ true. Because x_a is delimited, it has some quantity α associated with it. Furthermore, since the spatial trace is assumed to be unchanging, the predicate $\mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_{\mathsf{s}}(e_a))$ must hold as well. Finally, since the e_a are the same, the last conjuncts hold trivially. Now let us consider the reverse entailment. Since pour forms incremental thematic relations, we know that $\mathsf{pour}(\mathsf{j}, \mathsf{gr}(x_a), \mathsf{gr}(e_a))$ must be true. Since the spatial trace is unchanging, $\tau_{\mathsf{s}}(\mathsf{gr}(e_a))$ is equal to $\tau_{\mathsf{s}}(e_a)$, and thus $\mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_{\mathsf{s}}(\mathsf{gr}(e_a)))$ holds as well. By definition, we have $\mathsf{comp}(\mathsf{gr}(e_a), e_a)$. Finally, since wort is homogeneous, $\mathsf{wort}(\mathsf{gr}(x_a))$ must also be true, completing the proof.

(b) Here we will only consider the top-to-bottom entailment; the reverse case is analogous (mutatis mutandis) to the previous case. Let $R \equiv [\lambda y.\, \lambda e.\, \mathsf{fill\text{-}with}(\mathsf{j}, y, x, e)]$. Since $\lambda y.\, \lambda e.\, \mathsf{distr}(R, y, e)$ is an incremental thematic relation, we know there is some y_a satisfying $\mathsf{comp}(y, y_a)$ which

makes $\mathsf{distr}(R, y_a, e_a)$ true. Because $\mathsf{plur(carboy)}$ is homogeneous, the conjunct $\mathsf{plur(carboy)}(y_a)$ follows. Since y_a is delimited, it must have some cardinality n. Finally, since the e_a are the same, the last conjuncts hold trivially.

(c) First let us consider the top-to-bottom entailment. Because $\lambda e.\,\mathsf{run}(\mathsf{j}, e)$ is homogeneous, we know that $\mathsf{run}(\mathsf{j}, e_a)$ must be true. Since e_a is delimited, its spatial trace must have some end location p_a. Finally, since the e_a are the same, the last conjuncts hold trivially. Now let us consider the reverse entailment. Because $\lambda e.\,\mathsf{run}(\mathsf{j}, e)$ is homogeneous, we know that $\mathsf{run}(\mathsf{j}, \mathsf{gr}(e_a))$ must be true. By the definitions of comp and gr, we also know that $\mathsf{comp}(\mathsf{gr}(e_a), e_a)$ holds generally. Finally, since the e_a are the same, the last conjuncts hold trivially.

(c′) These entailments follow for almost exactly the same reasons as in the previous case; we need only substitute the condition that the spatial trace of e_a must have some distance d_a, since it is delimited.

4.8.3 The Imperfective Paradox

The imperfective paradox distinguishes activity expressions from accomplishment ones, as long as we fix the interpretation of the progressive to the non-iterated in-progress reading (and restrict our attention to the individuating accomplishment expressions):

(4.212) (a) $\models$ $\dfrac{\text{Jack was pouring wort into the carboy.}}{\text{Jack poured wort into the carboy.}}$

(b) $\models$ $\dfrac{\text{Jack was filling carboys with wort.}}{\text{Jack filled carboys with wort.}}$

(c) $\models$ $\dfrac{\text{Jack was running along the river.}}{\text{Jack ran along the river.}}$

(4.213) (a) $\models$ $\dfrac{\exists e.\,\exists e'.\,\exists x.\,\left[\begin{array}{l}\mathsf{wort}(x)\ \wedge\ \mathsf{pour}(\mathsf{j}, x, e')\ \wedge\ \mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_{\mathrm{s}}(e'))\ \wedge \\ \mathsf{inprog}(e', e)\ \wedge\ \tau_{\mathrm{t}}(e) \preceq_{\mathrm{i}} \mathsf{now}\ \wedge\ \mathsf{Actual}(e)\end{array}\right]}{\exists e_a'.\,\exists e'.\,\exists x.\,\left[\begin{array}{l}\mathsf{wort}(x)\ \wedge\ \mathsf{pour}(\mathsf{j}, x, e')\ \wedge\ \mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_{\mathrm{s}}(e'))\ \wedge \\ \mathsf{comp}(e', e_a')\ \wedge\ \tau_{\mathrm{t}}(e_a') \preceq_{\mathrm{i}} \mathsf{now}\ \wedge\ \mathsf{Actual}(e_a')\end{array}\right]}$

(b) $\models$

$$\frac{\exists e.\,\exists e'.\,\exists x_a.\,\exists x.\,\left[\begin{array}{l}\mathsf{wort}(x)\ \wedge\ \mathsf{plur(carboy)}(x_a)\ \wedge \\ \mathsf{distr}\left([\lambda y.\,\lambda e.\,\mathsf{fill\text{-}with}(\mathsf{j}, y, x, e)], x_a, e'\right)\ \wedge \\ \mathsf{inprog}(e', e)\ \wedge\ \tau_{\mathrm{t}}(e) \preceq_{\mathrm{i}} \mathsf{now}\ \wedge\ \mathsf{Actual}(e)\end{array}\right]}{\exists e_a'.\,\exists e'.\,\exists x_a.\,\exists x.\,\left[\begin{array}{l}\mathsf{wort}(x)\ \wedge\ \mathsf{plur(carboy)}(x_a)\ \wedge \\ \mathsf{distr}\left([\lambda y.\,\lambda e.\,\mathsf{fill\text{-}with}(\mathsf{j}, y, x, e)], x_a, e'\right) \\ \wedge\ \mathsf{comp}(e', e_a')\ \wedge\ \tau_{\mathrm{t}}(e_a') \preceq_{\mathrm{i}} \mathsf{now}\ \wedge\ \mathsf{Actual}(e_a')\end{array}\right]}$$

(c) $\models$ $\dfrac{\exists e.\,\exists e'.\,\left[\begin{array}{l}\mathsf{run}(\mathsf{j}, e')\ \wedge\ \mathsf{along}(\mathsf{the}(\mathsf{river}), e')\ \wedge \\ \mathsf{inprog}(e', e)\ \wedge\ \tau_{\mathrm{t}}(e) \preceq_{\mathrm{i}} \mathsf{now}\ \wedge\ \mathsf{Actual}(e)\end{array}\right]}{\exists e_a'.\,\exists e'.\,\left[\begin{array}{l}\mathsf{run}(\mathsf{j}, e')\ \wedge\ \mathsf{along}(\mathsf{the}(\mathsf{river}), e')\ \wedge \\ \mathsf{comp}(e', e_a')\ \wedge\ \tau_{\mathrm{t}}(e_a') \preceq_{\mathrm{i}} \mathsf{now}\ \wedge\ \mathsf{Actual}(e_a')\end{array}\right]}$

In cases (a) and (c), let e be a moment and e' a process satisfying the antecedent. By our condition on in-progress moments, we are guaranteed the existence of a period e_a of which e is an element satisfying $\mathsf{inprog}(e', e_a)$. This then guarantees that there is an ${e_a}' \trianglelefteq e_a$ such that $\mathsf{comp}(e', {e_a}')$ and $\tau_{\mathrm{t}}(e_a) = \tau_{\mathrm{t}}({e_a}')$, assuming our idealized condition on the existence of subevents holds. Finally, because Actual is preserved by the structural part-of relation $\trianglelefteq$, we have $\mathsf{Actual}({e_a}')$, and thus e' and ${e_a}'$ satisfy all the conjuncts in the consequent. Now, in case (b), the proof proceeds as before, up to the point where the idealized condition on the existence of subevents applies; because pluralities of events have readily identifiable minimal parts, such a condition does not make sense here. For this reason, the proof will not go through unless we make a caveat that the time of evaluation is not in the middle of the first filling event (at least).

(4.214) (a) $\not\models$ Jack was pouring twenty liters of wort into the carboy. / Jack poured twenty liters of wort into the carboy.

(b) $\not\models$ Jack was filling twenty carboys with wort. / Jack filled twenty carboys with wort.

(b′) $\not\models$ Jack was filling the carboy with wort. / Jack filled the carboy with wort.

(c) $\not\models$ Jack was running to the museum. / Jack ran to the museum.

(c′) $\not\models$ Jack was running two miles. / Jack ran two miles.

(4.215) (a) $\not\models$
$$\frac{\exists e.\, \exists e'.\, \exists x'.\, \exists x.\, \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{comp}(x, x') \wedge \rho(x') = \mathsf{liters}(20) \wedge \\ \mathsf{pour}(\mathsf{j}, x', e') \wedge \mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_{\mathrm{s}}(e')) \wedge \\ \mathsf{inprog}(\mathsf{gr}(e'), e) \wedge \tau_{\mathrm{t}}(e) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e) \end{array}\right]}{\exists e'.\, \exists x'.\, \exists x.\, \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{comp}(x, x') \wedge \rho(x') = \mathsf{liters}(20) \wedge \\ \mathsf{pour}(\mathsf{j}, x', e') \wedge \mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_{\mathrm{s}}(e')) \wedge \\ \tau_{\mathrm{t}}(e') \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e') \end{array}\right]}$$

(b) $\not\models$
$$\frac{\exists e.\, \exists e'.\, \exists x'.\, \exists x.\, \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{plur}(\mathsf{carboy})(x') \wedge |x'| = 20 \wedge \\ \mathsf{distr}\left([\lambda y.\, \lambda e.\, \mathsf{fill\text{-}with}(\mathsf{j}, y, x, e)], x', e'\right) \wedge \\ \mathsf{inprog}(\mathsf{gr}(e'), e) \wedge \tau_{\mathrm{t}}(e) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e) \end{array}\right]}{\exists e'.\, \exists x'.\, \exists x.\, \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{plur}(\mathsf{carboy})(x') \wedge |x'| = 20 \wedge \\ \mathsf{distr}\left([\lambda y.\, \lambda e.\, \mathsf{fill\text{-}with}(\mathsf{j}, y, x, e)], x', e'\right) \wedge \\ \tau_{\mathrm{t}}(e') \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e') \end{array}\right]}$$

(b′) $\not\models$
$$\frac{\exists e.\, \exists e'.\, \exists x.\, \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{fill\text{-}with}(\mathsf{j}, \mathsf{the}(\mathsf{carboy}), x, e') \wedge \\ \mathsf{inprog}(\mathsf{gr}(e'), e) \wedge \tau_{\mathrm{t}}(e) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e) \end{array}\right]}{\exists e'.\, \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{fill\text{-}with}(\mathsf{j}, \mathsf{the}(\mathsf{carboy}), x, e') \wedge \\ \tau_{\mathrm{t}}(e') \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e') \end{array}\right]}$$

(c) $\not\models$
$$\frac{\exists e.\, \exists e'.\, \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e') \wedge \mathsf{to}(\mathsf{the}(\mathsf{museum}), e') \wedge \\ \mathsf{inprog}(\mathsf{gr}(e'), e) \wedge \tau_{\mathrm{t}}(e) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e) \end{array}\right]}{\exists e'.\, \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e') \wedge \mathsf{to}(\mathsf{the}(\mathsf{museum}), e') \wedge \\ \tau_{\mathrm{t}}(e') \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e') \end{array}\right]}$$

(c′) $\not\models$
$$\frac{\exists e.\, \exists e'.\, \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e') \wedge \rho(\tau_{\mathrm{s}}(e')) = \mathsf{miles}(2) \wedge \\ \mathsf{inprog}(\mathsf{gr}(e'), e) \wedge \tau_{\mathrm{t}}(e) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e) \end{array}\right]}{\exists e'.\, \left[\begin{array}{l} \mathsf{run}(\mathsf{j}, e') \wedge \rho(\tau_{\mathrm{s}}(e')) = \mathsf{miles}(2) \wedge \\ \tau_{\mathrm{t}}(e') \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e') \end{array}\right]}$$

In each of these cases, the process $\mathsf{gr}(e')$ ground from the event e' can be in progress without e' necessarily being an actual event.

4.8.4 Aspectual Verbs

The preceding examples of the imperfective paradox work equally well if the progressive is replaced by either of the aspectual verbs *start* or *stop*; we will consider just two here:

(4.216) (a) $\not\models$ Jack stopped pouring twenty liters of wort into the carboy. / Jack poured twenty liters of wort into the carboy.

(b′) $\not\models$ Jack stopped filling the carboy with wort. / Jack filled the carboy with wort.

(4.217) (a) $\not\models$
$$\frac{\exists e_1 . \exists e' . \exists x' . \exists x . \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{comp}(x, x') \wedge \rho(x') = \mathsf{liters}(20) \wedge \\ \mathsf{pour}(\mathsf{j}, x', e') \wedge \mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_{\mathrm{s}}(e')) \wedge \\ \mathsf{stop}(\mathsf{gr}(e'), e_1) \wedge \tau_{\mathrm{t}}(e_1) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_1) \end{array} \right]}{\exists e' . \exists x' . \exists x . \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{comp}(x, x') \wedge \rho(x') = \mathsf{liters}(20) \wedge \\ \mathsf{pour}(\mathsf{j}, x', e') \wedge \mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_{\mathrm{s}}(e')) \wedge \\ \tau_{\mathrm{t}}(e') \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e') \end{array} \right]}$$

(b′) $\not\models$
$$\frac{\exists e_1 . \exists e' . \exists x . \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{fill\text{-}with}(\mathsf{j}, \mathsf{the}(\mathsf{carboy}), x, e') \wedge \\ \mathsf{stop}(\mathsf{gr}(e'), e_1) \wedge \tau_{\mathrm{t}}(e_1) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_1) \end{array} \right]}{\exists e' . \exists x . \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{fill\text{-}with}(\mathsf{j}, \mathsf{the}(\mathsf{carboy}), x, e') \wedge \\ \tau_{\mathrm{t}}(e') \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e') \end{array} \right]}$$

Since the stopping event e_1 marks the end of a state in which the process $\mathsf{gr}(e')$ is in progress, the conjunct in the consequent requiring the event e' to be actual is not satisfied.

Of course, if we replace *stop* by *finish*, these judgements reverse:

(4.218) (a) $\models$ Jack finished pouring twenty liters of wort into the carboy. / Jack poured twenty liters of wort into the carboy.

(b′) $\models$ Jack finished filling the carboy with wort. / Jack filled the carboy with wort.

(4.219) (a) $\models$
$$\frac{\exists e_1 . \exists e' . \exists x' . \exists x . \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{comp}(x, x') \wedge \rho(x') = \mathsf{liters}(20) \wedge \\ \mathsf{pour}(\mathsf{j}, x', e') \wedge \mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_{\mathrm{s}}(e')) \wedge \\ \mathsf{finish}(\mathsf{gr}(e'), e_1) \wedge \tau_{\mathrm{t}}(e_1) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_1) \end{array} \right]}{\exists e' . \exists x' . \exists x . \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{comp}(x, x') \wedge \rho(x') = \mathsf{liters}(20) \wedge \\ \mathsf{pour}(\mathsf{j}, x', e') \wedge \mathsf{into}(\mathsf{the}(\mathsf{carboy}), \tau_{\mathrm{s}}(e')) \wedge \\ \tau_{\mathrm{t}}(e') \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e') \end{array} \right]}$$

(b′) $\models$
$$\frac{\exists e_1 . \exists e' . \exists x . \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{fill\text{-}with}(\mathsf{j}, \mathsf{the}(\mathsf{carboy}), x, e') \wedge \\ \mathsf{finish}(\mathsf{gr}(e'), e_1) \wedge \tau_{\mathrm{t}}(e_1) \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e_1) \end{array} \right]}{\exists e' . \exists x . \left[\begin{array}{l} \mathsf{wort}(x) \wedge \mathsf{fill\text{-}with}(\mathsf{j}, \mathsf{the}(\mathsf{carboy}), x, e') \wedge \\ \tau_{\mathrm{t}}(e') \preceq_{\mathrm{i}} \mathsf{now} \wedge \mathsf{Actual}(e') \end{array} \right]}$$

Since the finishing event e_1 marks the end of a maximal state in which the process $\mathsf{gr}(e')$ is in progress, e' must be actual in this case.

4.8.5 Distributive Temporal Adverbials

The distributive readings of the temporal adverbials shown below give rise to the following entailments:

(4.220) (a) $\models$ $\dfrac{\text{Swans glided past the dock in 30 seconds for 10 minutes.}}{\text{A swan glided past the dock in 30 seconds.}}$

(b) $\models$ $\dfrac{\text{Twenty swans glided along the shore for 30 seconds in 10 minutes.}}{\text{A swan glided along the shore for 30 seconds.}}$

(4.221) (a) $\models$

$$\frac{\exists e_a.\,\exists e.\,\exists x.\left[\begin{array}{l}\mathsf{plur}(\mathsf{swan})(x)\wedge\\ \mathsf{distr}\left(\lambda x.\,\lambda e.\left[\begin{array}{l}\mathsf{glide}(x,e)\wedge\\ \mathsf{past}(\mathsf{the}(\mathsf{dock}),\tau_{\mathrm{s}}(e))\wedge\\ \rho(\tau_{\mathrm{t}}(e))\preceq_{\mathrm{a}}\mathsf{seconds}(30)\end{array}\right],x,e\right)\wedge\\ \mathsf{comp}(e,e_a)\wedge\rho(\mathsf{cvx}(\tau_{\mathrm{t}}(e_a)))=\mathsf{minutes}(10)\wedge\\ \tau_{\mathrm{t}}(e_a)\preceq_{\mathrm{i}}\mathsf{now}\wedge\mathsf{Actual}(e_a)\end{array}\right]}{\exists e'.\,\exists x'.\left[\begin{array}{l}\mathsf{swan}(x')\wedge\mathsf{glide}(x',e')\wedge\mathsf{past}(\mathsf{the}(\mathsf{dock}),\tau_{\mathrm{s}}(e'))\wedge\\ \rho(\tau_{\mathrm{t}}(e'))\preceq_{\mathrm{a}}\mathsf{seconds}(30)\wedge\tau_{\mathrm{t}}(e')\preceq_{\mathrm{i}}\mathsf{now}\wedge\mathsf{Actual}(e')\end{array}\right]}$$

(b) $\models$

$$\frac{\exists e_a.\,\exists x_a.\left[\begin{array}{l}\mathsf{plur}(\mathsf{swan})(x_a)\wedge\ |x_a|=20\wedge\\ \mathsf{distr}\left(\lambda x.\,\lambda e_a.\,\exists e.\left[\begin{array}{l}\mathsf{glide}(x,e)\wedge\\ \mathsf{along}(\mathsf{the}(\mathsf{shore}),\tau_{\mathrm{s}}(e))\wedge\\ \mathsf{comp}(e,e_a)\wedge\\ \rho(\tau_{\mathrm{t}}(e_a))=\mathsf{seconds}(30)\end{array}\right],x_a,e_a\right)\wedge\\ \rho(\mathsf{cvx}(\tau_{\mathrm{t}}(e_a)))\preceq_{\mathrm{a}}\mathsf{minutes}(10)\wedge\tau_{\mathrm{t}}(e_a)\preceq_{\mathrm{i}}\mathsf{now}\wedge\mathsf{Actual}(e_a)\end{array}\right]}{\exists e_a'.\,\exists e'.\,\exists x'.\left[\begin{array}{l}\mathsf{swan}(x')\wedge\mathsf{glide}(x',e')\wedge\mathsf{along}(\mathsf{the}(\mathsf{shore}),\tau_{\mathrm{s}}(e'))\wedge\\ \mathsf{comp}(e',e_a')\wedge\rho(\tau_{\mathrm{t}}(e_a'))=\mathsf{seconds}(30)\wedge\\ \tau_{\mathrm{t}}(e_a')\preceq_{\mathrm{i}}\mathsf{now}\wedge\mathsf{Actual}(e_a')\end{array}\right]}$$

These entailments follow straightforwardly from the axiomatization of the distributive relation distr. In (4.221a), we know from before that the plurality of events e_a is entailed to be one in which some number of swans glide past the dock in thirty seconds; to get the entailment to go through then, we need only let x' and e' be one of these swan-event pairs. In (4.221b), the antecedent directly contains a plurality of events e_a in which some number (twenty) of swans glide along the shore for thirty seconds; here again we need only let x' and ${e_a}'$ be one of these swan-event pairs to make the entailment go through, as desired.

4.9 Summary of Part I

In the first part of this thesis, I have pursued a sortal approach to aspectual composition based upon the intuitive idea that **events** and **processes** are ontologically distinct entities on a par with **objects** and **substances**. In so doing, I have developed an account which synthesis proposals by Jackendoff (1991), Hinrichs (1985) and Krifka (1989; 1992) to make correct predictions in many cases not considered by these authors.

Of the many components of the account, there are two worth singling out as central. The first is the use of an **order-sorted logic** as the translation

language, which enables one to distinguish formulas that are merely false from those that are not well-sorted, and thus semantically anomalous. Without this component, the account could not explain the striking contrast between the pair *Jack ran two miles along the river* and *Jack ran along the river for two miles*, where the two sentences are logically equivalent, and the pair *Jack ran two miles to the bridge* and * *Jack ran to the bridge for two miles*, where the latter sentence is semantically anomalous.

The second key component of the account is its treatment of delimitedness, i.e. that feature which distinguishes substances and processes (undelimited) from objects and events (delimited). In the present account, the undelimited entities—viz., the substances, processes, and their plural and path analogues—are given a natural and uniform treatment in terms of **equivalence classes**, a notion that also plays a central role in the present adaptation of Habel's (1990) approach to formalizing paths. In a nutshell, the idea is simply to treat an undelimited entity as a reified continuum of its delimited counterparts.

This abstract treatment of the generalized count/mass distinction enables the definition of the broadly applicable notion of an **incremental thematic relation**, i.e. a relation which serves to link a continuum in the eventuality domain with one in either the material domain or path domain. Without this component of the theory, the account could not explain why *Jack poured twenty liters of wort into the carboy* entails *Jack poured ten liters of wort into the carboy*, nor why *Jack poured wort into the carboy for ten seconds* entails *Jack poured some amount of wort into the carboy in ten seconds*.

The successes of the account include, on the syntactic side, explanations of many of the acceptable readings involving complex interactions amongst temporal adverbials, the present tense, the progressive, aspectual verbs, negation, mass terms, and bare plurals. On the semantic side, the successes include explanations of downward entailments, existential entailments, distributive entailments, and, partially, the imperfective paradox. A hallmark of the present account is its ability to explain the syntactic and semantic behavior of **non-individuating accomplishment expressions**, which have heretofore remained mysterious.

Inevitably, for every contribution there is of course a question that must be left open. First and foremost is the generality of the account. Interestingly, the present treatment of undelimited entities turns out to be quite similar to Hinrichs' treatment of Carlsonian **kinds**; as such, it would do much to bolster the present approach if it could be shown to be compatible with recent work on **genericity**, where abstract kinds are usually employed.

Another interesting question that arises in this regard is how the present account could be made to fit with recent work on dynamically-changing discourse contexts. Not surprisingly, there are several places in the account where a static semantics turns out to be inadequate to explain the phenomena at hand, the most notable one being the interaction of *in*-adverbials with activity expressions.

The last question relating to the generality of the account concerns its treatment of modality, which is admittedly simplistic. While the bare bones distinc-

tion between actual and non-actual eventualities employed here does suffice to account for the cases of the imperfective paradox under study, it remains to be seen whether the present account can be generalized in accord with more sophisticated treatments of possible worlds or situations.

Perhaps the next most important question concerns the treatment of momentaneous events. While I have partially characterized these in terms of the processes or states they mark the ends of, much remains to be said about these events, including exactly how they differ from moments, i.e. states at particular instants. The reason these have been left relatively unexplored here is that a proper treatment of these events would seem to require a sophisticated theory of **granularity**, which is beyond the scope of this study.

The final two questions are more technical in nature. First, while the account could not work without **aspectual type coercions**, it rather simplistically requires all such coercions to be lexicalized in order to avoid overgeneration; this creates the obvious technical problem of how to recapture the generalizations across the various category-meaning pairs assigned to each lexical item. The second question concerns the sort checking mechanism: at present, well-sortedness is defined semantically in terms of consistency; clearly it would be preferable to devise a more restricted notion that could be incorporated into the type-checking mechanism.

Rather than dwell on these exciting possibilities for future work, however, let us now turn to Part II, where we will explore the potential computational applications of the model-theoretic analysis.

Part II

Computational Applications

Chapter 5

A Calculus of Eventualities

In this chapter I present an implemented calculus of eventualities which covers a subset of the linguistic account developed in Part I. In section 5.1, I discuss the motivations behind the calculus and compare it with previous research. In section 5.2, I present the calculus itself, at a conceptual level; details of the implementation are left for section 5.3. Finally, in section 5.4, I summarize the calculus, identifying its merits and shortcomings.

5.1 Motivation

In his thesis (Moens, 1987, p. 112), Moens argues that the identification of the correct ontology for a principled treatment of aspect is also a vital preliminary to the construction and management of temporal databases:[1]

> A bad fit between datastructures and human concepts will almost always result in a loss of interaction flexibility. In the case of natural language systems the penalties for a bad fit will be that some of the information expressed in the language will be lost when it is being forced into the incompatible datastructures. Conversely, these datastructures cannot be queried very easily using natural language since the conceptual structure implicit in the query does not fit the conceptual structure of the database.

To make his argument more concrete, Moens shows how his ontology facilitates the construction of a simple statement verifier that checks input against a

[1] We might question, of course, whether it is realistic to expect an application to structure its data with the needs of a natural language interface in mind. This point is made by Fedder (1991), who investigates ways of mapping information stored in a database to more linguistically relevant structures. Nevertheless, this does not negate the main point of Moens' argument; instead, it merely divides up the problem into application-specific and application-independent components of a natural language interface.

database. His program works by translating English yes-no questions into Prolog queries, which are then answered using an augmented version of Kowalski and Sergot's (1986) **event calculus**. While his program adequately handles a wide range of interesting cases, it does not address the problem of aspectual composition. Moreover, as we shall see, there are some difficulties inherent in his treatment of the imperfective paradox.

This chapter aims to improve upon Moens' treatment by developing a calculus of eventualities—based upon a subset of the model-theoretic account developed in part I—which addresses these shortcomings. As in Moens' approach, the goal of this chapter will be to show how the calculus facilitates the construction of a simple statement verifier. Note, however, that the relative importance of these issues to this and other natural language processing tasks will not be directly addressed; doing so would require a corpus study for an appropriate task domain, which is beyond the scope of this work.

Before examining Moens' account, let us briefly review his reasons for taking Kowalski and Sergot's event calculus as his starting point. By taking events (rather than situations) to be basic, Kowalski and Sergot are able to represent both simultaneous and partially-ordered events. This sets their calculus apart from the well-known **situation calculus** of McCarthy and Hayes (1969), where neither possibility exists. Furthermore, by letting the situations which precede or follow events to be partial (rather than complete), Kowalski and Sergot's calculus largely circumvents the wearisome **frame problem**. Moens (1987, p. 123) sums up their event calculus as follows:[2]

> On the whole, the event calculus is a very flexible general purpose language that can be used for planning, database updating and querying, as well as for natural language processing. Although it does not contain all the machinery that will be needed to deal with an extended fragment of language, its underlying ontology is such that an extension of the calculus is easy to envisage.

Moens implements a number of extensions to Kowalski and Sergot's calculus. Of particular interest here is his treatment of extended events, which Kowalski and Sergot do not definitively formalize. To handle these, Moens follows Steedman (1982) in introducing two punctual endpoint events for each extended event. These endpoint events may either start, stop or culminate the extended event with which they are associated. For example, Moens represents an event of Mary walking to the restaurant as follows:

(5.1)
```
event(walk(mary, restaurant), c3).
occur(start(c3), 1935).
occur(cul(c3), 2015).
```

[2]For a more complete review, as well as a comparison with Allen's temporal logic (1984), see Moens' thesis (Moens, 1987).

These clauses state that `c3` is an event of mary walking to the restaurant which starts at time 1935 and culminates at time 2015. To represent processes, or events which do not culminate, `stop` is used instead of `cul`:

(5.2)
```
event(eat(mary, dinner), c5).
occur(start(c5), 2030).
occur(stop(c5), 2100).
```

Here we have an event `c5` of Mary eating dinner which is interrupted (i.e. stops) at time 2100, and thus does not culminate.

As was noted in section 2.2, Moens' treatment of incomplete events is quite reminiscent of Parsons' (Parsons, 1990) analysis. To help clarify the connection, let us examine the following problem with Parsons' approach, due to Zucchi (1993):

> Parsons' theory is based on the assumption that in the denotation of an event predicate of type P some events may be incomplete, i.e. may be events which are not culminated. ... This assumption is the key to Parsons' solution to the imperfective paradox: a progressive sentence, unlike a simple past one, does not require the event it describes to culminate, and this is why *John was building a bookcase* fails to entail *John built a bookcase*. There is a difficulty with this view, however, which is brought out by the following case. Suppose Gianni was going by train from Milan to Florence and that, due to a strike of the railroad workers, he went only as far as Piacenza. According to Parsons' analysis of the progressive, the trip that takes Gianni to Piacenza is an event of Gianni's going from Milan to Florence, but not a culminated one. But now, let's ask the question: did Gianni go to Piacenza? The answer is yes. In Parsons' analysis, this means that the event of Gianni's going to Piacenza is a culminated one. Since the event of Gianni's going to Piacenza is also a non-culminated event of Gianni's going to Florence, it follows, paradoxically, that the same event is both culminated and non-culminated.

Zucchi's paradox may be straightforwardly translated into Moens-style representations as follows:

(5.3)
```
event(go(gianni, milan, florence), e1).
occurs(start(e1), 2000).
occurs(stop(e1), 2200).

event(go(gianni, milan, piacenza), e1).
occurs(start(e1), 2000).
occurs(cul(e1), 2200).
```

For Moens, the obvious way out of Zucchi's paradox is to deny that these events should be identified in the first place, as they are in Parson's treatment. While this move is certainly the correct one to take, it has the unfortunate consequence of leaving the relationship between Gianni's trip to Florence and his passage to Piacenza unspecified. The linguistic account developed in part I addresses this problem by simply letting Gianni's excursion to Piacenza be a proper subevent of his journey to Florence; this captures the relationship between these two events in a natural way, without requiring Gianni to actually reach Florence.

Interestingly, Oberlander and Dale (1991) observe what is roughly the converse problem in their treatment of nominalizations. Remarking on the pair *I had a discussion with Fred* and *I had two hours of discussion with Fred*, they note that their knowledge-base representations preclude the two object NPs from referring to the same event. These representations appear in (5.4)a and (5.5)a below:

(5.4) (a)
$$\left[\begin{array}{l} index = e_2 \\ spec = \left[\begin{array}{l} structure = individual \\ substance = discussing \\ particips = [in = [agent = \langle s, f \rangle]] \end{array}\right] \end{array}\right]$$

(b) I had a discussion with Fred.

(5.5) (a)
$$\left[\begin{array}{l} index = e_5 \\ spec = \left[\begin{array}{l} structure = mass \\ substance = discussing \\ quantity = \left[\begin{array}{l} unit = hour \\ number = 2 \end{array}\right] \\ particips = [in = [agent = \langle s, f \rangle]] \end{array}\right] \end{array}\right]$$

(b) I had two hours of discussion with Fred.

The problem here is that the two eventualities e_2 and e_5 cannot possibly be the same, since they have conflicting values for the attribute labeled *structure*. Ultimately, the authors suggest, we would like to have some means of tying these representations together as different perspectives on the same entity; they do not, however, give any further insight into how one might do this.

Oberlander and Dale's observation is reminiscent of our treatment of the pair *Jack ran to the bridge in thirty minutes* and *Jack ran along the river for thirty minutes*: while these two sentences need not refer to the same event, nothing prevents them from doing so, despite the different perspectives they convey. In the next section, we will see how the calculus handles similar cases in a different domain.

5.2 The Calculus

This section presents both general and domain specific rules for constructing and reasoning about eventualities. The calculus is primarily formulated in the Horn-clause subset of first-order logic, though some second-order terms are also employed; as such, the calculus may be (and has been) straightforwardly implemented as a logic program. While the resulting implementation is not claimed to be an efficient one, it does serve as a suitable basis for a simple statement verifier, and thus provides a means for automatically verifying the inferences at the core of the present study. Details of the implementation appear in the next section, together with some suggestions for improved efficiency.

The domain specific rules apply to an environment consisting of two agents (Jack and Jill), two substances (water and wort), and two containers (a smaller and a larger carboy). The agents have two known methods of transferring a substance into or out of a container, pouring and siphoning; in so doing, they may cause the container to become full or empty. For convenience, a naive physics is assumed whereby all transfers take place without interruption, and at constant rate.

5.2.1 Ontology

The calculus simplifies the sortal lattice developed in section 4.1 in two ways:

1. The sort axioms and relativizations employed in the calculus ensure that only well-sorted formulas may be true, but they do not distinguish formulas that are false from those that are not well-sorted.

2. Plural and spatial entities are not considered.

Otherwise I will follow the ontology of section 4.1 quite closely.

The calculus employs the named eventuality subsorts which appear in Figure 5.1. As the diagram shows, the feature stative divides the stative eventualities from the rest. The eventualities are again partitioned by the attribute dimension, which can take on values of 0 and 1. Finally, the feature delimited subdivides the one-dimensional eventualities.[3]

As noted in section 4.1, the sublattice of eventualities just described facilitates an elegant treatment of the cooccurrence patterns of temporal adverbials. For example, *at*-adverbials may be specified to select for zero-dimensional eventualities, while leaving stativity unspecified. This enables them to modify both moments (e.g, *The carboy was full at time 5*) and momentaneous events (e.g,

[3]Note that the symbols for the stative eventualities are lighter than their non-stative counterparts, to graphically suggest their relative inactivity. As for the delimitedness distinction, note that the undelimited eventualities appear as dotted or dashed lines without endpoints, to suggest their status as equivalence classes of their delimited counterparts. This distinction should become clearer in the ensuing subsections.

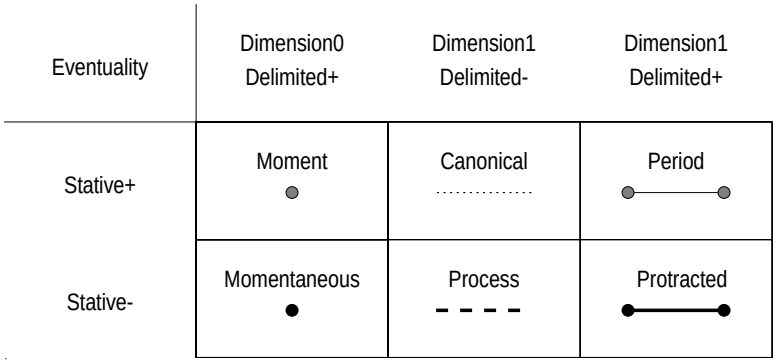


Figure 5.1: The named eventuality subsorts.

Jack started emptying the carboy at time 10), as desired. In a similar vein, *for*-adverbials may be specified to select for undelimited eventualities, irrespective of stativity—that is, for canonical states (*The carboy was full for 10 seconds*) and processes (*Jack poured wort into the carboy for 10 seconds*). Finally, *in*-adverbials may be specified to select for protracted events only (*Jack filled the carboy in 10 seconds*).

In the sortal lattice, the material entities are likewise partitioned by the delimited feature, yielding the named subsorts substance and object. As we shall see, this enables the definition of a schema equating the delimitedness features of the material and eventuality arguments to an **incremental thematic relation** (cf. section 4.4).

5.2.2 An Example

To simplify the exposition, we will consider a single extended example throughout the section. Let us suppose we have the database listed in Figures 5.3 and 5.4 and pictured in Figures 5.5 and 5.6, which might be derived from the hypothetical text shown in Figure 5.2. While automatically extracting such a database from naturally occurring English texts lies beyond the scope of this project, it is nevertheless useful to consider some of what would be involved in doing so.

Each clause in the hypothetical text introduces one or more eventualities into the database; these appear grouped together in Figures 5.3 and 5.4. The explicitly mentioned eventualities are pictured in Figure 5.5, where those pertaining to large carboy appear in upper part of the diagram, and those pertaining to the small carboy in the lower part; since the former ones always involve the substance wort and the agent Jack, and the latter ones the substance water and

1. At time 0 (seconds), the large carboy was full of wort, and the small one full of water.

2. At time 10, Jack started siphoning wort out of the large carboy. At the same time, Jill started emptying the small one.

3. Jill finished 10 seconds later. After waiting a full 20 seconds, she refilled the small carboy, in the same amount of time as before.

4. Jack stopped siphoning when she finished. He paused 10 seconds, then poured the 8 liters of wort back into the large carboy.

5. At time 80, 12 seconds later, both carboys were still full.

Figure 5.2: Hypothetical text for example database.

the agent Jill, these additional participants are included in the headings of these two groups.[4]

In addition to the explicitly mentioned eventualities, the database contains the implicit ones pictured in Figure 5.6. These eventualities serve to connect the explicitly mentioned ones so that there is a continuous episodic chain for each carboy. For example, consider the state e2_s_c2 which follows the event e1_e_c2 of Jill emptying the small carboy. Following the terminology of Steedman (1982), this state is termed the **consequent state** of this event.[5] This state is the same one that precedes the next event involving the small carboy, e3_c2, where Jill fills it up again; symmetrically, this state is termed the **antecedent state** of e3_c2. That the state e2_s_c2 is one in which the small carboy is empty is inferred via a simple rule pertaining to the consequent states of emptying events.

It is important to note that for further inferencing to go through—e.g., to infer that the small carboy is full at time 30, say—equalities such as the one between the consequent state of the filling event e1_e_c2 and the antecedent state of the emptying event e3_c2 must be present in the database. In this respect, the present calculus is much less ambitious than the one of Kowalski and Sergot, which focuses on precisely this issue of how to infer such equalities through default reasoning about the persistence of states. Since this issue seems to be largely independent of the problems at the core of the present study, it has been left aside here.

[4]Note that the emptying event e1_e_c2 is not pictured, just the emptying process e1_p_c2.

[5]It also taken to be the consequent state of the event e2_c2 of Jill finishing emptying the small carboy; this is what appears in the database.

```
%% background
wort(wort). water(water).
carboy(carboy1). vol(carboy1, 28.0).
carboy(carboy2). vol(carboy2, 20.0).


%% 1.
full_of(carboy1, wort, e0_c1).
time(e0_c1, 0.0). actual(e0_c1).
moment(e0_s_c1, e0_c1).

full_of(carboy2, water, e0_c2).
time(e0_c2, 0.0). actual(e0_c2).
moment(e0_s_c2, e0_c2).


%% 2.
start(e1_p_c1, e1_c1).
  siphon_out_of(jack, wort, carboy1, e1_p_c1).
time(e1_c1, 10.0). actual(e1_c1),
antec_st(e1_c1, e0_s_c1).

start(e1_p_c2, e1_c2). gr(e1_e_c2, e1_p_c2).
  empty_of(jill, carboy2, water, e1_e_c2).
time(e1_c2, 10.0). actual(e1_c2).
antec_st(e1_c2, e0_s_c2).
```

Figure 5.3: Example database (i).

```
%% 3.
cul(e1_p_c2, e2_c2).
time(e2_c2, 20.0). actual(e2_c2),
conseq_st(e2_c2, e2_s_c2).

fill_with(jill, carboy2, water, e3_c2).
time(e3_c2, intv(40.0, 50.0)). actual(e3_c2).
antec_st(e3_c2, e2_s_c2). conseq_st(e3_c2, e3_s_c2).


%% 4.
stop(e1_p_c1, e2_c1).
time(e2_c1, 50.0). actual(e2_c1).
conseq_st(e2_c1, e2_s_c1).
pour_into(jack, wort1, carboy1, e3_c1).
  comp(wort, wort1). quant(wort1, 8.0).
time(e3_c1, intv(60.0, 68.0)). actual(e3_c1).
antec_st(e3_c1, e2_s_c1). conseq_st(e3_c1, e3_s_c1).


%% 5.
full_of(carboy1, wort, e4_c1).
time(e4_c1, 80.0). actual(e4_c1),
moment(e3_s_c1, e4_c1).

full_of(carboy2, water, e4_c2).
time(e4_c2, 80.0). actual(e4_c2).
moment(e3_s_c2, e4_c2).
```

Figure 5.4: Example database (ii).

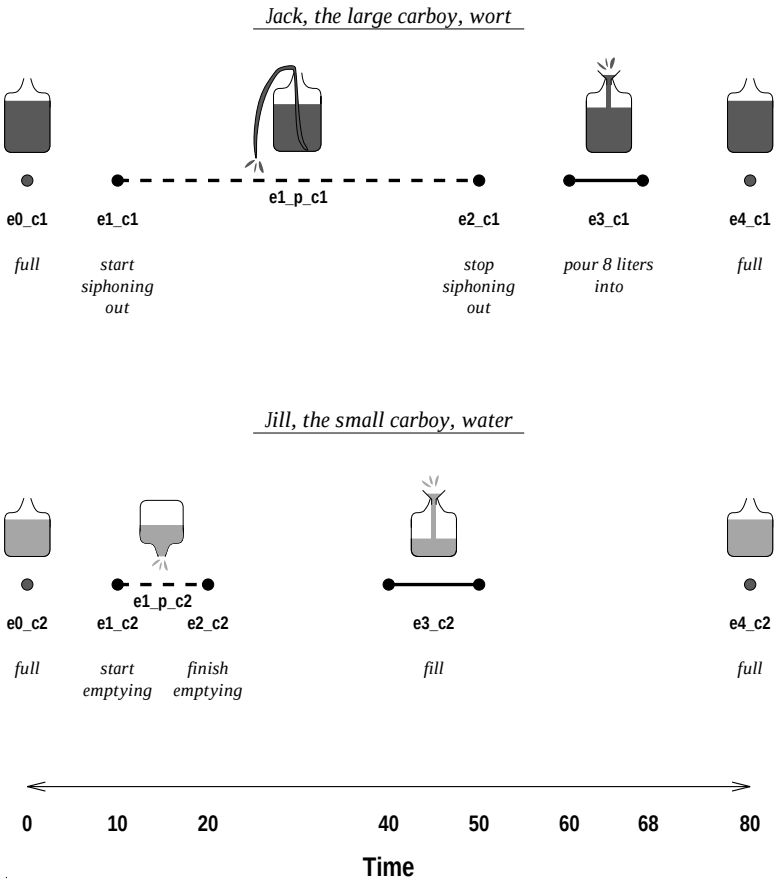


Figure 5.5: The explicitly mentioned eventualities.

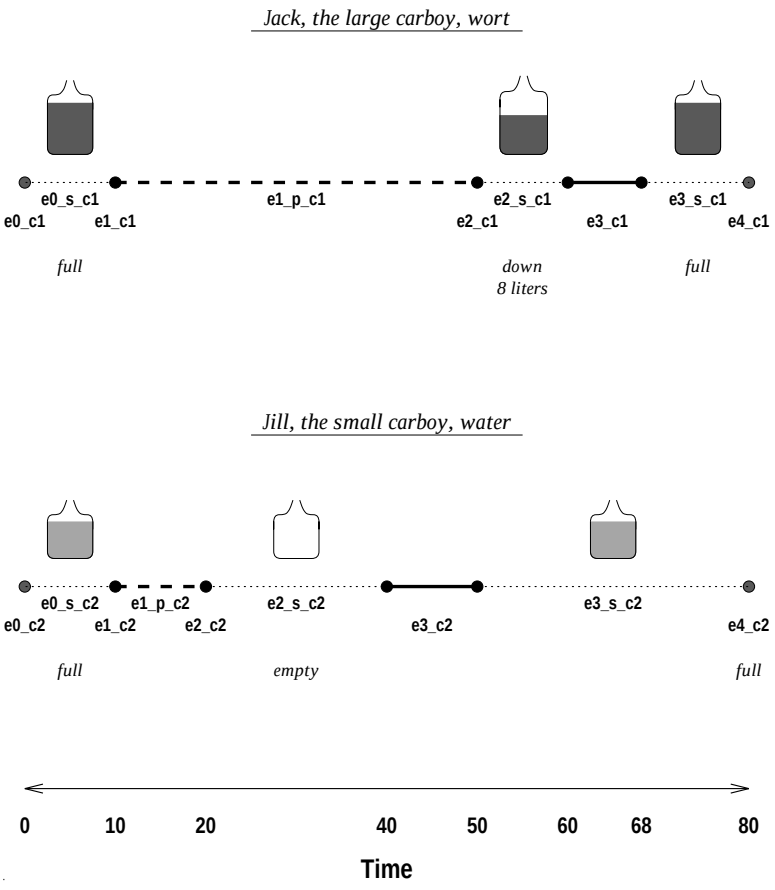


Figure 5.6: The implicit eventualities.

```
| ?- q(db).
|: The large carboy was full of wort for 5 seconds.

u: (((large(_6816), carboy(_6816)),
     (wort(_7983), full_of(_6816,_7983,_8484)),
     for(5.0,_8484,_10521)),
    actual(_10521)) : t

((large(carboy1), carboy(carboy1)),
 (wort(wort), full_of(carboy1,wort,e0_s_c1)),
 for(5.0,e0_s_c1,comp(e0_s_c1,intv(0.0,5.0)))),
actual(comp(e0_s_c1,intv(0.0,5.0)))

yes


| ?- q(db).
|: Jack started pouring water into the large carboy.

u: ((((large(_9100), carboy(_9100)),
      water(_7903), pour_into(jack,_7903,_9100,_6829)),
     start_plus(_6829,_11334)),
    actual(_11334)) : t

no
```

Figure 5.7: Example queries.

```
%% named eventuality subsorts
state(E)     :- stative(E, +).
canonical(E) :- state(E), delimited(E, -).
moment(E)    :- state(E), delimited(E, +), dimension(E, 0).
period(E)    :- state(E), delimited(E, +), dimension(E, 1).

non_state(E)    :- stative(E, -).
process(E)      :- non_state(E), delimited(E, -).
event(E)        :- non_state(E), delimited(E, +).
momentaneous(E) :- event(E), dimension(E, 0).
protracted(E)   :- event(E), dimension(E, 1).


%% named material subsorts
substance(X) :- material(X), delimited(X, -).
object(X)    :- material(X), delimited(X, +).
```

Figure 5.8: The named sorts.

5.2.3 Queries

The statement verifier takes as input a simple declarative sentence, which it then attempts to determine the truth of with respect to a given database. To do so, the input sentence is first parsed to derive a logical form; this logical form then serves as the goal to be proved using the eventuality calculus. The proof search is implemented via a meta-interpreter differing in only minor ways from standard Prolog; details of the parser and the meta-interpreter appear in the next section.

Figure 5.7 shows two example queries, both of which are evaluated with respect to the database pictured in Figures 5.5 and 5.6. The first query succeeds, since the large carboy is entailed to be full during the five-second interval from time 0 to time 5; the second query fails, however, since it does not follow from the database that Jack ever starts pouring water into the large carboy. Note that the program is not designed to provide a more helpful response; in a practical question-answering system, it would be important for the program to respond by adding clarifying information—e.g., that the large carboy was in fact known to be full for ten seconds, in the first case, or that it was wort rather than water that Jack poured into the carboy, in the second.

```
%% 1. special relations
stative(E, +) :- conseq_st(_, E).
stative(E, -) :- start(_, E).

%% 2. lexical relations
stative(E, +) :- stative_pred(Pred),    Pred*E.
stative(E, -) :- nonstative_pred(Pred), Pred*E.

stative_pred(E^ full_of(_, _, E)).
protracted_pred(E^ fill_with(_, _, _, E)).

nonstative_pred(Pred) :- protracted_pred(Pred).

%% 3. delimitedness and incremental thematic relations
delimited(E, V) :- itr(Rel), Rel*X*E, delimited(X, V).
itr(X^E^ pour_into(_, X, _, E)).
```

Figure 5.9: Sort axioms.

5.2.4 Sort Axioms

The named sorts of entities and the features used to distinguish them are listed in Figure 5.8. To determine the values of these features, the rules shown in Figure 5.9 (and others like them) are invoked. These rules are explained below:

1. The first pair of rules is representative of those that determine sortal features from special relations such as conseq_st and start; the calculus contains similar rules for delimitedness and dimensionality. As an example, consider the eventualities e2_s_c2 and e1_c1. Since e2_s_c2 is the consequent state of e2_c2, the first rule applies, and thus it can be proven to be stative; similarly, since e1_c1 is the event which starts the process e1_p_c1, the second rule can be invoked to prove that it is non-stative.

2. The second group of rules determines sortal features from lexical relations such as full_of or fill_with. As an example, consider the eventuality e0_c1. Since the database contains the fact full_of(carboy1, wort, e0_c1), and since the predicate E^ full_of(carboy1, wort, E) can be shown to be one that selects for states (via the third rule listed), the first rule applies to make e0_c1 provably stative. Note that these rules require manipulating higher-order terms, where λ-abstraction is encoded using an infix ^ operator, and application is encoded using an infix * operator. Similarly, the eventuality e3_c2 can be proven to be non-stative, using the fact fill_with(jill, carboy2,

water, e3_c2) and the rest of the rules in this group.

3. The last group concerns the special case of incremental thematic relations, and how they interact with the delimitedness feature. The first rule states that the delimitedness feature of the eventuality argument to an incremental thematic relation must be the same as that of the material argument; the second rule states that pour_into forms incremental thematic relations. As an example of how these rules determine delimitedness, consider the eventuality e3_c1. Since the database contains the fact pour_into(jack, wort1, carboy1, e3_c1), and since wort1 is provably delimited, these rules can be used to prove that e3_c1 is delimited as well.

5.2.5 Moments and Periods

The calculus contains rules for determining the level of a substance in a container at all relevant instants, i.e. all instants within the time period spanned by the continuous episode involving the given participants. The calculus also contains rules for determining the substance level for all (relevant) intervals where the level remains unchanged. As we shall see below, an important facet of these and other similar rules is the ability to construct eventualities (in this case, moments and periods) entailed to exist by other eventualities already present in the database.

Let us begin by considering the query *The large carboy was full at time 5.* The database contains the moment e0_c1, at time 0, in which the large carboy is full of wort; it also contains the event e1_c1 of Jack starting to siphon wort out of the large carboy, at time 10. Connecting these two eventualities is the canonical state e0_s_c1: this state is known to have e0_c1 as one of its moments, as indicated by the binary moment (-of) relation;[6] in addition, it is also identified as the antecedent state of e1_c1.

From these facts, it should be possible to infer the existence of a moment at time 5 in which the large carboy is full. Carrying out this inference requires constructing a moment not already present in the database; since the desired moment is the only one which is both a moment of the canonical state e0_s_c1 and at time 5, it can be uniquely identified by the term moment(e0_s_c1, 5).[7] This is pictured in Figure 5.10.

The key rules necessary to verify the query in this way are listed in Figure 5.11, divided into four groups. The role that each of the groups plays in the proof process is explained below:

[6]This relation is a slight twist on the atomic quantity part-of relation $\sqsubseteq_{aq}$ discussed section 3.3.1, since it holds between a moment and a canonical state or process, rather than a period or protracted event.

[7]Note that the functor moment is overloaded, having both relation and constructor uses. For simplicity, I will continue to overload relation and constructor symbols in the sequel.

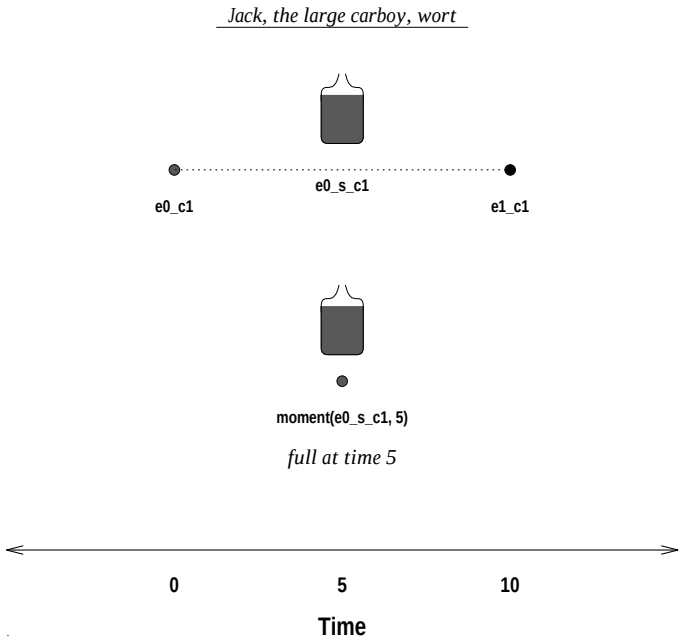


Figure 5.10: Verifying the query *The large carboy was full at time 5.*

```
%% 1.
full(C, E)       :- full_of(C, X, E).
full_of(C, X, E) :- level(C, X, 1.0, E).

%% 2.
level(C, X, N, E) :-
  canonical(ES), level(C, X, N, ES), moment(ES, E).
level(C, X, N, ES) :-
  moment(ES, E0), level(C, X, N, E0).

%% 3.
moment(ES, moment(ES, T)) :-
  comp(ES, EA), time(EA, TA), betw(TA, T).
comp(ES, comp(ES, I)) :-
  moment_comp(ES, E0, EA), time(EA, IA), subintv(IA, I).
moment_comp(ES, E0, comp(E, intv(T0, T1))) :-
  moment(ES, E0), time(E0, T0), end(ES, E1), time(E1, T1).
end(ES, E1) :-
  antec_st(E1, ES), dimension(E1, 0).

%% 4.
at(T, E) :- dimension(E, 0), time(E, T).
```

Figure 5.11: Rules used in verifying the query full(carboy1, E), at(E, 5).

1. The first group establishes the connection between the predicate full and the predicate level. The first rule reduces full(carboy1, E) to the predicate full_of(carboy1, X, E), for some substance X; the second rule reduces this latter predicate to level(carboy1, X, 1.0, E). These reductions enable a uniform treatment of predicates such as full and empty.

2. The second group grounds the proof in the canonical state e0_s_c1, of which E will turn out to be a moment. The first rule finds a canonical state ES which has the same level of the substance X in carboy1, by proving the subgoal level(carboy1, X, 1.0, ES); it then shows that E is a moment of ES by proving the subgoal moment(ES, E). The second rule in this group applies to the former of these two subgoals; it succeeds when ES is bound to e0_s_c1, and binds E0 to e0_c1 and X to wort.

3. The third group completes the proof of full(carboy1, E) by proving the goal moment(e0_s_c1, E). The first rule constructs the desired moment by binding E to moment(e0_s_c1, T), and then shows that the time T of this moment falls within the interval TA of some period EA of e0_s_c1; note that this temporal restriction is necessary to prevent the query *The large carboy was full at time 15* from succeeding, for example. To find an appropriate period EA, the composed-of relation comp is invoked, via the subgoal comp(e0_s_c1, EA). The latter three rules suffice to prove this goal, by constructing the period named comp(e0_s_c1, intv(0, 10)).

4. The final group consists of a single rule which proves the second conjunct of the original goal by binding the time T to 5. This yields the moment named moment(e0_s_c1, 5), as pictured in Figure 5.10.

The previous example concerned derived moments of canonical states. Next we will consider an example requiring derived moments of processes, namely the query *The small carboy was half full at time 45.* Here the relevant eventuality in the database is the event e3_c2 of Jill filling the small carboy. Since this event takes from time 40 to time 50, and the carboy is empty beforehand, this query should be verified given our constant rate of transfer assumption. This is pictured in Figure 5.12.

The key additional rules necessary to verify this query are listed in Figure 5.13. The first rule is invoked in proving the goal level(carboy2, X, 0.5, E). This rule looks for a protracted event EA in which some agent adds some quantity XA of the substance X to the small carboy. As suggested above, in this case EA is bound to e3_c2, using a rule not listed. The temporal interval is then bound to intv(40, 50); since the particular substance does not matter, I will leave this part of the proof aside. The next step in the proof is to construct the desired moment of the process which makes up e3_c2. This is achieved by first invoking the ground-from relation gr, which constructs the process gr(e3_c2) via the second rule listed; this process is then passed to the same moment-of rule

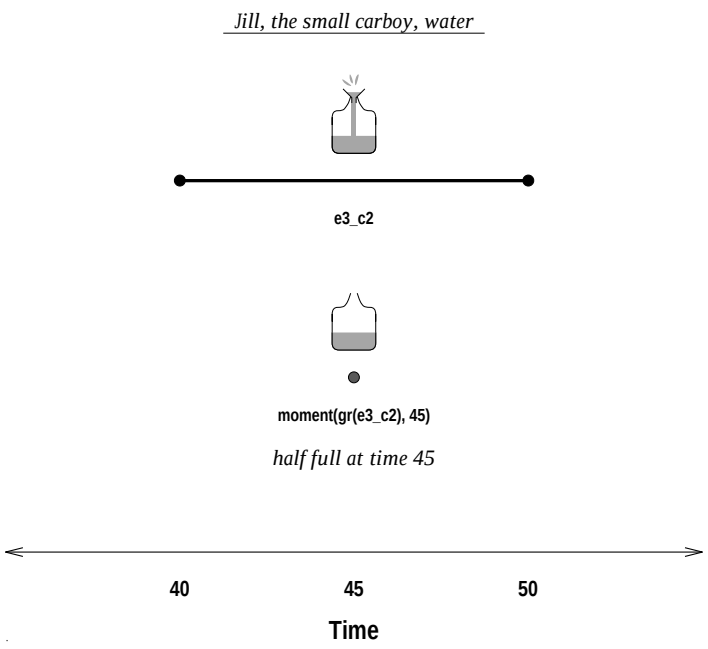


Figure 5.12: Verifying the query *The small carboy was half full at time 45.*

```
%% 1.
level(C, X, N, E) :-
  protracted(EA),
  add_to(_, XA, C, EA), comp(X, XA), time(EA, intv(T0, T1)),
  gr(EA, EP), moment(EP, E), time(E, T),
  antec_st(EA, ES0),  level(C, X, N0, ES0),
  conseq_st(EA, ES1), level(C, X, N1, ES1),
  N = N0 + (N1 - N0) * (T - T0) / (T1 - T0).

%% 2.
gr(E, gr(E)) :- protracted(E).

%% 3.
level(C, X, 1.0, E) :-
  conseq_st(EA, E), fill_with(_, C, X, EA).
level(C, X, 0.0, E) :-
  conseq_st(EA, E), empty_of(_, C, X, EA).
```

Figure 5.13: Rules used in verifying the query half_full(carboy2, E), at(E, 45).

as before, which yields the moment moment(gr(e3_c2), T). The last step is to calculate the value of T by linear interpolation, using the initial and final levels N0 and N1; these values are established to be 0.0 and 1.0, respectively, by the last pair of rules listed. Since T0 and T1 are bound to 40 and 50, T turns out to be 45 and the query succeeds.

Having discussed the rules for moments, we now turn to periods. The first query we will discuss is *The large carboy was full from time 2 to time 8*. This inference just requires constructing a period from the state e0_s_c1 which is a proper part of the maximal (known) period, as pictured in Figure 5.14. The important part of the translation of this query is shown below:

(5.6) `full(carboy1, E), from_to(2, 8, E, EA)`

To prove the first conjunct, the eventuality E is bound to the canonical state e0_s_c1, which is determined to be a state in which the large carboy is full as described previously. To prove the second conjunct, just one further rule is needed:

(5.7)
```
from_to(T0, T1, E, EA) :-
    comp(E, EA), time(EA, intv(T0, T1)).
```

Using the same rule for the composed-of relation as before, this goal succeeds by binding EA to the constructed period comp(e0_s_c1, intv(2, 8)). Note that this

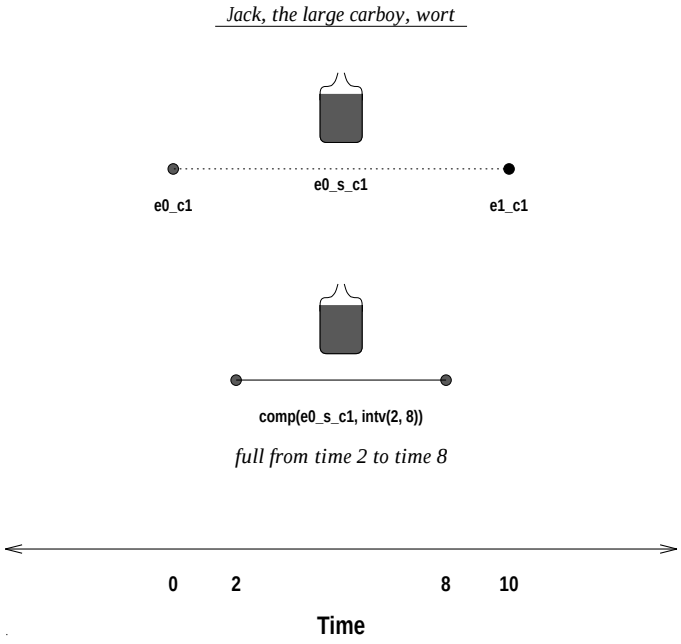


Figure 5.14: Verifying the query *The large carboy was full from time 2 to time 8.*

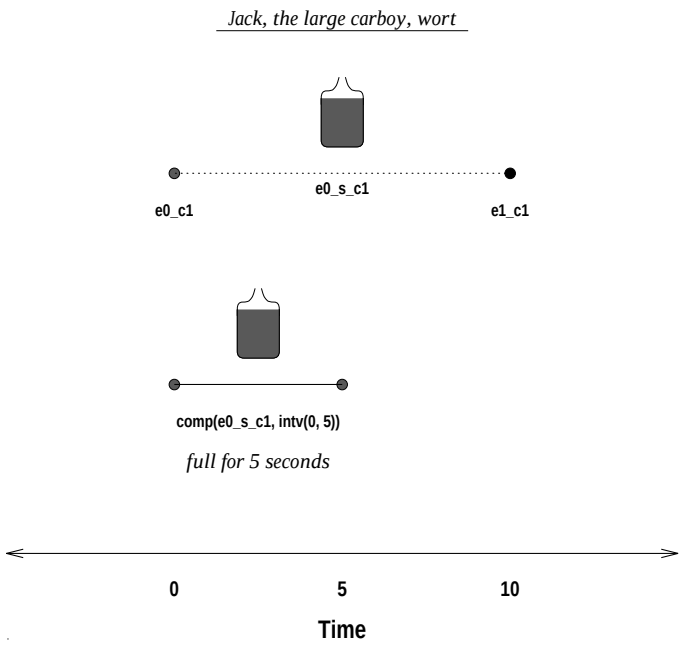


Figure 5.15: Verifying the query *The large carboy was full for 5 seconds.*

period has as its time the interval from time 2 to time 8, which is a subinterval of the one from time 0 to time 10, as required.

A slight variation on this example is the query *The large carboy was full for 5 seconds*; the important part of the translation appears below:

(5.8)
```
full(carboy1, E), for(5, E, EA)
```

This query again requires constructing a period of e0_s_c1 which is a proper subpart of the maximal known period; this time, however, the exact period in question is not completely determined. Since this choice makes no difference, the initial one may be chosen by default, as pictured in Figure 5.15. The additional rules for completing the proof are listed below:[8]

(5.9)
```
for(D, E, EA) :-
  comp(E, EA), dur(EA, D).
dur(E, D) :-
  time(E, intv(T0, T1)), D = T1 - T0.
```

[8]The careful reader may have noticed that the composed-of relation is introduced in the rule for for, rather than in the translation; this difference is not taken to be significant here.

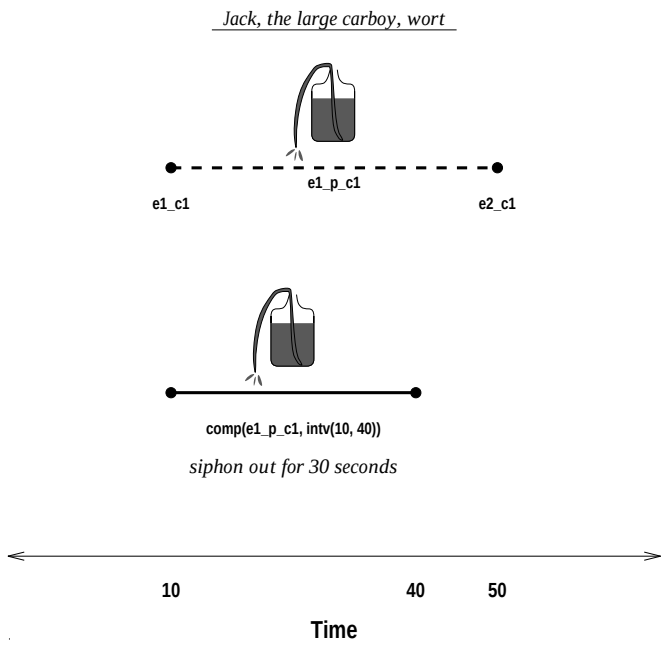


Figure 5.16: Verifying the query *Jack siphoned wort out of the large carboy for 30 seconds.*

5.2.6 Downward Entailments

The next examples concern downward entailments with processes and events. First we will consider how the process/event distinction determines downward entailments with respect to time; then we will consider how incremental thematic relations determine downward entailments with respect to material entities.

Let us begin by considering the query *Jack siphoned wort out of the large carboy for 40 seconds*. Since the database contains the process `e1_p_c1` of Jack siphoning wort out of the large carboy, and the events `e1_c1` at time 10 and `e1_c2` at time 50 in which this process starts and stops (respectivey), this query should be verified. The important part of the translation appears below:

(5.10) `siphon_out_of(jack, wort, carboy1, E), for(40, E, EA)`

To make this query go through, an additional case of the composed-of relation needs to be considered, namely the case in which the maximal delimited counterpart of an undelimited entity is constructed:

(5.11)
```
comp(E, comp(E, I)) :-
  max_comp(E, EA), time(EA, IA), subintv(IA, I).
max_comp(E, comp(E, intv(T0, T1))) :-
  start(E, E0), time(E0, T0),
  stop(E, E1),  time(E1, T1).
```

These rules suffice to prove the query above, by binding E to e1_p_c1 and EA to comp(e1_p_c1, intv(10, 50)). Note that these rules also suffice to verify similar queries with lesser durations; for example, the query *Jack siphoned wort out of the carboy for 30 seconds* can be proved by constructing the protracted event comp(e1_p_c1, intv(10, 40)), as shown in Figure 5.16.

Next we turn to an example involving an event rather than a process. Consider the query *Jill filled the small carboy in 10 seconds*. Since the database contains the event e3_c2 of Jill filling the small carboy, which takes exactly ten seconds, this query should be verified. Using the rule for in listed below, a proof for this query is easily found:

(5.12)
```
in(D, E) :-
  protracted(E), dur(E, D0), D0 =< D.
```

Now consider what happens when we try a query with a lesser duration, such as *Jill filled the small carboy in 8 seconds*. In this case, the durational constraint provided by the *in*-adverbial will simply fail to be satisfied.

Having examined downward entailments with respect to time, we turn now to downward entailments with respect to material entities, in the case of incremental thematic relations. As our example, we will consider the query *Jack poured 6 liters of wort into the large carboy*; since the database contains an event e3_c1 of Jack pouring eight liters of wort into the large carboy, this query should be verified. To do so, this inference requires constructing a subevent of e3_c1 in which three-fourths of the eight liters is transferred; given our constant rate assumption, this subevent will of course take three-fourths as long as e3_c1, as pictured in Figure 5.17.

Carrying out this inference requires the additional rules shown in Figure 5.18. These are explained below:

1. The first pair of rules is used to connect the goal pour_into(jack, XB, carboy1, EB) to a schema for downward entailments with incremental thematic relations. In the second rule, the relation Rel is bound to X^E^ pour_into(jack, X, carboy1, E).

2. The first rule in the second group serves to construct the desired subevent. The first subgoal binds EA to e3_c1 and XA to wort1 using the fact pour_into(jack, wort1, carboy1, e3_c1). The rest of the subgoals construct a subpart of wort1, XB, and a subpart of e3_c1, EB, such that the quantity-duration ratio (i.e., the rate) remains constant. The second rule in this

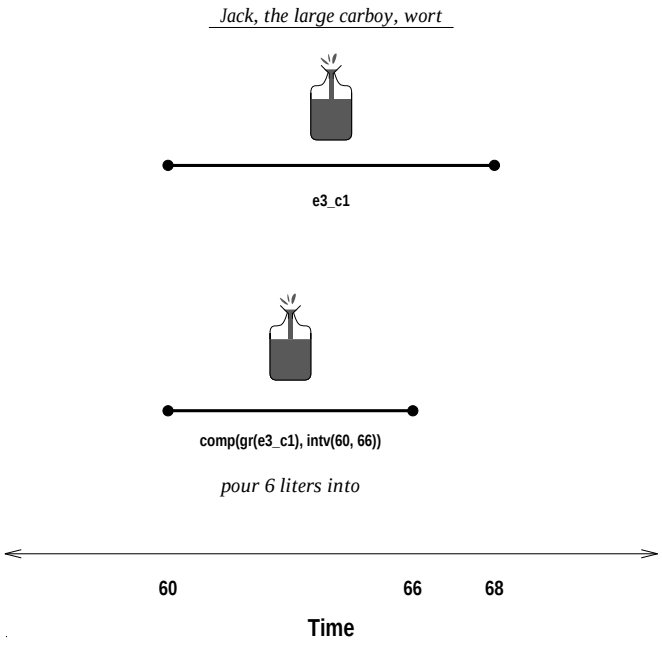


Figure 5.17: Verifying the query *Jack poured 6 liters of wort into the large carboy.*

```
%% 1.
pour_into(A, X, C, E) :- itr_schema(pour_into(A, X, C, E)).
itr_schema(Prop) :-
  itr(Rel), Rel*X*E = Prop, ev_ev(Rel, X, E).

%% 2.
ev_ev(Rel, XB, EB) :-
  Rel*XA*EA,
  partof(XA, XB), partof(EA, EB),
  quant( XA, QA), dur(EA, DA),
  quant( XB, QB), dur(EB, DB),
  QA / DA = QB / DB.

partof(EA, EB) :- gr(EA, E), comp(E, EB).

%% 3.
comp(X, comp(X, _, _)) :- substance(X).
quant(comp(_, Q, _), Q).

%% 4.
wort(XA) :- comp(X, XA), wort(X).
wort(X)  :- gr(XA, X),   wort(XA).
```

Figure 5.18: Rules used in verifying the query wort(X), comp(X, XB), quant(XB, 6), pour_into(jack, XB, carboy1, EB).

group reduces the part-of relation to the composition of the composed-of and ground-from relations (cf. section 4.2). In the case of EB, this yields comp(gr(e3_c1), intv(60, 66)) as the desired subpart to which it is bound; as for XB, this case is treated below.

3. The third group constructs particular objects composed of a given substance. Note that a substance and a quantity do not uniquely determine a particular object; this contrasts with a process and an interval, which do uniquely determine a particular protracted event. For this reason, the name for the constructed object contains a free variable, in order to avoid unwanted identities. In the example at hand, the composition of comp and gr yields the subpart comp(gr(wort1), 6, _) of wort1, which is bound to XB.

4. The last group establishes that the predicate wort does in fact hold of the substance gr(wort1). It does so by requiring this predicate to be preserved by both comp and gr, which in two steps leads back to the fact wort(wort) present in the database.

5.2.7 Existential Entailments

The previous example involved the downward entailments which are particular to incremental thematic relations; the next examples again involve incremental thematic relations, this time focusing on their existential entailments.

First we will consider the query *Jack siphoned some amount of wort out of the large carboy*, which can be verified as pictured in Figure 5.19. What is interesting to observe here is that the event which verifies this query also verifies the query *Jack siphoned wort out of the large carboy for 40 seconds*, using the rules introduced so far. As such, this example illustrates how the present approach resolves Oberlander and Dale's problem of missing identities.

The additional rules invoked in proving this query are listed in Figure 5.20 and explained below:

1. The first pair of rules is used to connect the goal siphon_out_of(jack, XA, carboy1, EA) to a schema for existential entailments with incremental thematic relations. In the second rule, the relation Rel is bound to X^E^ siphon_out_of(jack, X, carboy1, E).

2. The single rule in the second group serves to construct the desired event. First X and E are bound to the substance wort and the process e1_p_c1 (respectively), using the fact siphon_out_of(jack, wort, carboy1, e1_p_c1). Then the composed-of relation is invoked to construct the particular object XA and event EA involved in the relation: using the previously introduce rules for comp, XA is bound to comp(wort, QA, _) and EA to comp(e1_p_c1, intv(10, 50)). Note that the particular quantity (or amount) QA involved in this event is constrained to be equal to the rate times the duration.

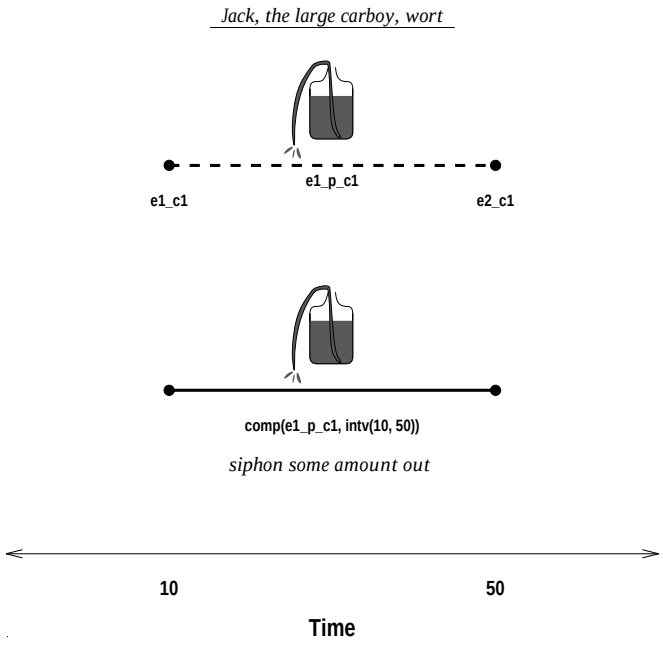


Figure 5.19: Verifying the query *Jack siphoned some amount of wort out of the large carboy.*

```
%% 1.
siphon_out_of(A, X, C, E) :- itr_schema(siphon_out_of(A, X, C, E)).
itr_schema(Prop) :-
  itr(Rel), Rel*X*E = Prop, proc_ev(Rel, X, E).


%% 2.
proc_ev(Rel, XA, EA) :-
  Rel*X*E, rate(E, R),
  comp(  X, XA), comp( E, EA),
  quant(XA, QA), dur( EA, DA),
  QA = DA * R.

%% 3.
rate(E, R) :-
  rate(E, X, R0), viscosity(X, V),
  R = R0 * V.
rate(E, X, 0.4)   :- siphon_out_of(_, X, _, E).
viscosity(X, 0.5) :- wort(X).
```

Figure 5.20: Rules used in verifying the query comp(wort, XA), quant(XA, _), siphon_out_of(jack, XA, carboy1, EA).

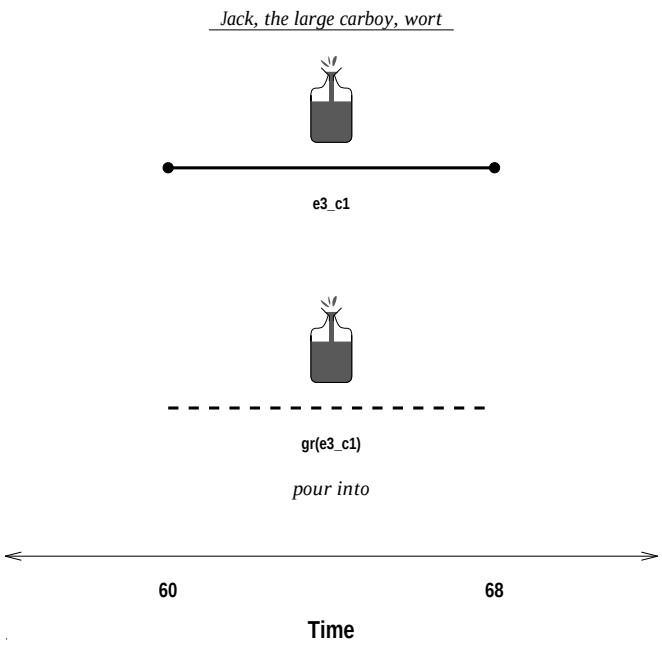


Figure 5.21: Verifying the query *Jack poured wort into the large carboy (for a while).*

3. The last group of rules establishes the rate of transfer depending on the method used and the viscosity of the substance transferred. Naturally, siphoning is slower than pouring, and wort is more viscous than water.

The process-to-event mapping just considered also has a natural converse, which is used to verify the query *Jack poured wort into the carboy (for a while).* Since the database contains an event `e3_c1` of Jack pouring eight liters of wort into the large carboy, this query can be verified as pictured in Figure 5.21. The important part of the translation follows:

(5.13)
```
wort(X), pour_into(jack, X, carboy1, E), comp(E, _)
```

This goal can be proved using the last case of the schema for incremental thematic relations, which appears below:

(5.14)
```
pour_into(A, X, C, E) :- itr_schema(pour_into(A, X, C, E)).
itr_schema(Prop) :-
  itr(Rel), Rel*X*E = Prop, ev_proc(Rel, X, E).
```

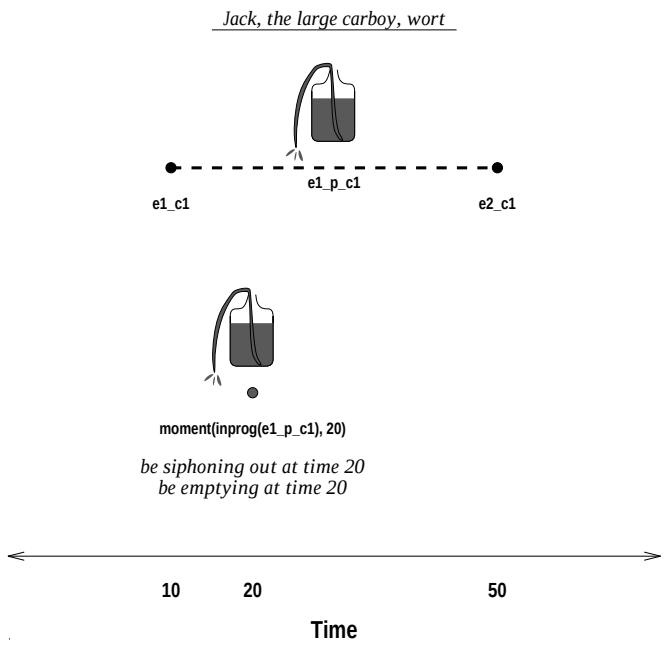


Figure 5.22: Verifying the queries *Jack was siphoning wort out of the large carboy at time 20* and *Jack was emptying the large carboy at time 20.*

```
ev_proc(Rel, X, E) :-
  Rel*XA*EA, gr(XA, X), gr(EA, E).
```

These rules serve to construct the process gr(e3_c1) and the substance gr(wort1), which suffice to complete the desired inference.

5.2.8 The Imperfective Paradox

We turn now to the imperfective paradox. The first two queries we will consider here concern the process e1_p_c1 of Jack siphoning wort out of the large carboy. From what is known about this process in the database, the query *Jack was siphoning wort out of the large carboy at time 20* should clearly be verified, as pictured in Figure 5.22. Now consider the query *Jack was emptying the large carboy at time 20*. Whether or not this query should be verified is somewhat unclear, since the truth of this statement would seem to depend on Jack's intentions; for example, if Jack never intended to remove more than the eight liters of wort he actually siphoned out, this statement should be false. If, on the other

```
%% 1.
inprog_plus(E, ES) :-
  inprog(E, ES0),
  (ES = ES0 ; moment(ES0, ES)).
inprog(E, inprog(E)) :- process(E).

%% 2.
actual(moment(E, T)) :-
  comp(E, EA), time(EA, TA), betw(TA, T),
  actual(EA).
actual(comp(E, intv(_, T1))) :-
  end(E, E1), time(E1, T1),
  actual(E1).

end(inprog(E), E1) :- stop(E, E1).
```

Figure 5.23: Rules used in verifying the query siphon_out_of(jack, wort, carboy1, E), inprog_plus(E, ES), at(20, ES), actual(ES).

hand, we know that Jack does (initially) intend to empty the carboy, then this latter query should be verified too, as is also pictured in Figure 5.22.

This additional knowledge is captured here by adding to the database that the process `e1_p_c1` is one that makes up (or, is 'ground from') an emptying event, as shown below:[9]

(5.15)
```
gr(e1_e_c1, e1_p_c1).
empty_of(jack, carboy1, wort, e1_e_c1).
```

These additional facts suffice to verify both queries. Note, however, that unlike the other events in the database, the emptying event `e1_e_c1` need not be an actual one; thus it is possible for *Jack was emptying the large carboy at time 20* to be true, but *Jack emptied the large carboy* to be false. Furthermore, the falsity of this latter statement is not inconsistent with the truth of statements such as *Jack siphoned 8 liters of wort out of the large carboy*, which illustrates how the present account avoids the problem that Zucchi's paradox reveals in Moens' analysis.[10]

Let us now examine the two queries above in detail. The additional rules used in verifying the first query are listed in Figure 5.23 and explained below:

[9] Cf. sections 2.2 and 4.2 for a more complete discussion of this move.

[10] Here we are letting *Jack empties the large carboy* be the analogue of *Gianni travels to Florence*, and *Jack siphons 8 liters of wort out of the large carboy* be the analogue of *Gianni travels to Piacenza*.

1. The first rule in this group, the relation inprog_plus, translates the progressive. It allows for either a process or an event as its input argument E, and yields either a canonical state or a moment as its result argument ES; only the process case is shown here.[11] After E is bound to the process e1_p_c1, the progressive state of this process inprog(e1_p_c1) is constructed using the second rule. This canonical state is then input to moment-of rule, ultimately yielding the moment moment(inprog(e1_p_c1), 20).

2. The next group of rules establishes that this moment satisifies the special predicate actual (which has been ignored to this point). The first rule is used to prove the actuality of the moment moment(inprog(e1_p_c1), 20) in terms of the actuality of some temporally inclusive period of the same canonical state inprog(e1_p_c1). The next two rules suffice to reduce the actuality of the temporally inclusive period comp(inprog(e1_p_c1), intv(10, 50)) to the fact actual(e2_c1) present in the database, where e2_c1 is the event in which the process e1_p_c1 stops.

Turning now to the second query, the important part of the translation is shown below:

(5.16)
```
empty(jack, carboy1, E), inprog_plus(E, ES),
  at(20.0, ES), actual(ES)
```

To verify this query, the following two additional rules are invoked:

(5.17)
```
empty(A, C, E) :- empty_of(A, C, _, E).
inprog_plus(E, ES) :-
  gr(E, EP), inprog(EP, ES0),
  (ES = ES0 ; moment(ES0, ES)).
```

The first rule simply reduces the predicate empty to the predicate empty_of for some substance. The second rule is the second case of inprog_plus mentioned above; unlike the previous one, this rule introduces the ground-from relation gr mapping the event E to the process EP. In this example, E is bound to the event e1_e_c1 of Jack emptying the carboy of wort, and EP to the process e1_p_c1 as before; from this point, the rest of the proof is the same as for the previous query, including the subproof of the goal actual(ES). It is this commonality which provides the key to the resolution of the imperfective paradox: since the proof does not depend on the actuality of the event e1_e_c1, this query can be verified, despite the fact that its non-progressive counterpart (*Jack emptied the large carboy*) cannot.

In the preceding example, the fact that the siphoning process in which Jack was engaged could make up an emptying event was not deemed sufficient evidence to determine that it in fact did. This case can be contrasted with one

[11] The careful reader may have noticed that ambiguity in the translation has been traded for disjunction in the introduced rule; this difference is not taken to be significant.

in which a process continues until the relevant state change actually occurs, as Landman (1992) points out (cf. also section 2.2). For example, consider the event e3_c1 of Jack pouring the 8 liters of wort back into the large carboy: because this event results in the large carboy being full again, it should be judged a filling event.[12] The following rule suffices to make this inference:

(5.18)
```
fill_with(A, C, X, E) :-
    add_to(A, XA, C, E), comp(X, XA),
    conseq_st(E, ES), level(C, X, 1.0, ES),
    actual(E).
```

This rule is all that needs to be added to get the query *Jack filled the large carboy* to go through.[13]

5.2.9 Aspectual Verbs

To complete this section, we will consider three examples involving the aspectual verbs *start, stop,* and *finish*. The first example involves the events e1_c1 and e2_c1, in which Jack starts siphoning wort out of the large carboy at time 10 and stops at time 50, respectively. From the facts in the database concerning these events, it should be possible to verify the query *Jack siphoned wort out of the large carboy for 40 seconds*, as pictured in Figure 5.24. Since this query involves an inference that we have already needed in the case of downward entailments, this query can in fact be verified using no additional rules.

The second example involves the events e1_c2 and e2_c2, in which Jill starts and finishes emptying the small carboy, respectively. The facts concerning these events should suffice to verify the query *Jill emptied the small carboy*, as pictured in Figure 5.25. Doing so requires only one additional rule, which derives the actuality of the emptying event e1_e_c2 from the actuality of the culmination e2_c2:

(5.19)
```
actual(E) :-
    gr(E, EP), cul(EP, E1), actual(E1).
```

The final example is essentially the converse of the preceding one. From the event e3_c2 of Jill filling the small carboy, which takes place during the interval from time 40 to time 50, it should be possible to verify both the query *Jill started filling the small carboy at time 40* and the query *Jill finished filling the small carboy at time 50*, as pictured in Figure 5.26. These queries involve two additional rules each, one for constructing the desired momentaneous event and one for deriving its actuality; these rules are listed below:

[12] From this it follows, of course, that the process gr(e3_c1) makes up a filling event (to complete the analogy between the two cases).

[13] Note that because of the restriction that the event E be an actual one, the query *Jack was emptying the large carboy at time 20* will still fail in the absence of the additional facts gr(e1_e_c1, e1_p_c1) and empty_of(jack, carboy1, wort, e1_e_c1).

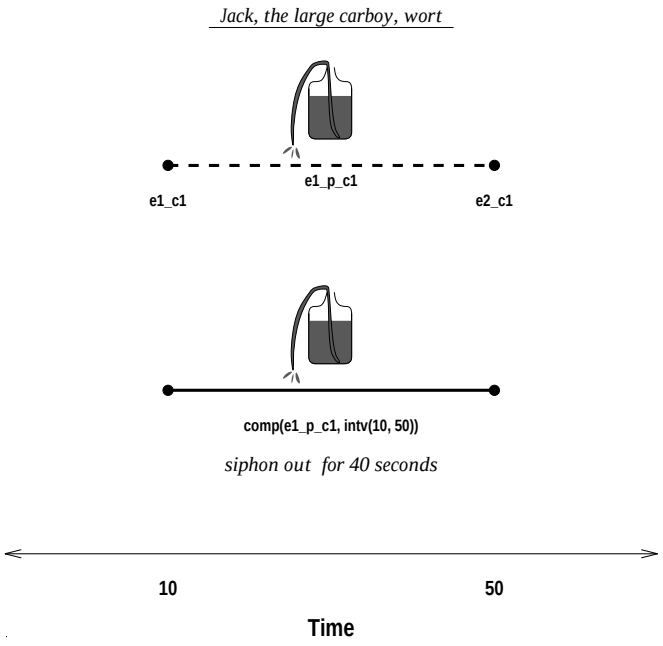


Figure 5.24: Verifying the query *Jack siphoned wort out of the large carboy for 40 seconds.*

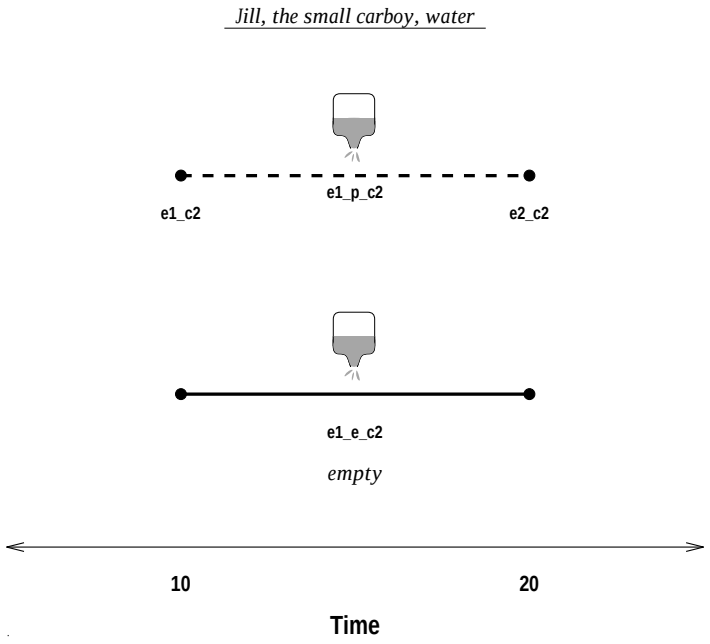


Figure 5.25: Verifying the query *Jill emptied the small carboy.*

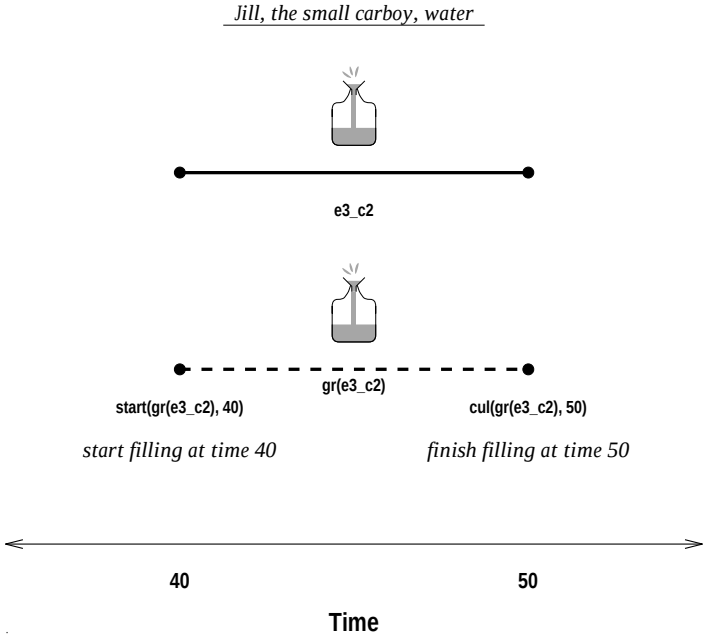


Figure 5.26: Verifying the queries *Jill started filling the small carboy at time 40* and *Jill finished filling the small carboy at time 50*.

```
(5.20)  start(E, start(E, T)) :- gr(EA, E), time(EA, intv(T, _)).
        cul(  E, cul(E, T)  ) :- gr(EA, E), time(EA, intv(_, T)).
        actual(start(gr(E), _)) :- actual(E).
        actual(cul(gr(E), _)  ) :- actual(E).
```

5.3 Implementation

This section discusses the implementation details omitted from the previous section, including the meta-interpreter, the parser, and efficiency concerns.

5.3.1 The Meta-Interpreter

As mentioned in the introduction to this chapter, Kowalski and Sergot take events (rather than states or situations) to be basic, which sets their calculus apart from the well-known situation calculus of McCarthy and Hayes. A problem with this approach is that a state cannot be represented in the database without making reference to an event which initiates or terminates it; thus, for example, their calculus cannot explicitly represent a moment in which the large carboy is full, in the absence of a known prior event in which this container becomes full.

In the present calculus, neither events nor states are taken to be basic, i.e. both can be present in the database, and both can be derived; moreover, the same is true for protracted events and momentaneous ones. Eschewing normal forms in this way allows for increased representational flexibility at only a small price—namely, some mechanism must be added to avoid infinite loops in the search procedure.

As an example of this problem, consider the following two rules for determining the actuality of some eventuality:

```
(5.21)  actual(E) :-
          gr(E, EP), cul(EP, E1), actual(E1).
        actual(cul(gr(E), _)) :-
          actual(E).
```

The first rule above is used to reduce the actuality of a protracted event to the actuality of its culmination, as in Figure 5.25; the second rule is used for the converse case, as in Figure 5.26. Clearly, this pair of rules will lead to an infinite loop under the normal top-down interpretation regime found in standard Prolog.

One way to get around this problem and others like it would be to employ a top-down technique augmented with memo-ization, as in the OLDT resolution framework discussed in (Tamaki and Sato, 1986) and (Kanamori and Kawamura, 1993). For convenience, a more easily implemented method is used here, as part of a simple meta-interpreter. To break loops induced by rules such as those in

(5.21), a higher-order operator may be employed which restricts the proof of a goal to database lookup, as shown below:

```
(5.22)  actual(E) :-
          known gr(E, EP), known cul(EP, E1),
          actual(E1).
        actual(cul(gr(E), _)) :-
          known actual(E).
```

Here the operator known prevents the top-down search from looping by ensuring that only one rule in this pair will ever apply.

A related problem which does not seem amenable to memo-ization techniques alone (Kanamori and Kawamura, 1993, p. 12) is the presence of rules which generate an infinite number of solutions. For example, consider the following triple of rules for determining that an eventuality EA is non-stative:

```
(5.23)  stative(EA, V) :- comp(E, EA), stative(E, V).
        stative(EP, -) :- gr(_, EP).
        gr(EA, gr(EA)) :- protracted(EA).
```

Since the goal protracted(EA) includes stative(EA, -) as a subgoal, this chain can be used to generate the infinite series of constructed events e, comp(gr(e),_), ...To resolve this problem, sort predicates can be restricted to generate only those eventualities present in the database, as shown below:

```
(5.24)  stative(comp(E, _), V) :- not var(E),        stative(E, V).
        stative(EA,         V) :- known comp(E, EA), stative(E, V).
        stative(gr(E),      -) :- not var(E).
        stative(EP,         -) :- known gr(_, EP).
```

By employing the var and known operators in this way, loops caused by sort predicates can be broken, while still allowing all the possible constructed events to be effectively filtered.

A different problem that arises in the implementation of the calculus is how to treat higher-order terms. Because the rules for incremental thematic relations require a sound treatment of bound variables, the first-order simulation of β-reduction discussed in (Pereira and Shieber, 1987) will not work; such an approach would fall prey to the **variable poisoning** problem discussed in (Moore, 1989) and (Pereira, 1990). On the other hand, the limited use of higher-order terms in the calculus does not warrant the full power of higher-order unification, as found in Nadathur and Miller's (1988) language λProlog; the limited (and decidable) extension to first-order unification found in Miller's (1991) L_λ would suffice. For convenience, however, the present implementation is in standard Prolog, necessitating the explicit manipulation of the higher-order terms in the few places in which they arise.

```
%% shift-reduce parser
parse([Result], [], Result).
parse([Cat1|Stack], Buffer, Result) :-
  null(Cat1, Cat2),
  parse([Cat2|Stack], Buffer, Result).
parse([Cat2, Cat1|Stack], Buffer, Result) :-
  reduce(Cat1, Cat2, Cat3),
  parse([Cat3|Stack], Buffer, Result).
parse(Stack, [Word|Buffer], Result) :-
  category(Word, Cat),
  parse([Cat|Stack], Buffer, Result).

%%>
reduce(X/Y : M  : A --> B,
       Y   : N  : A,
       X   : MN : B)
:-
  bnf(M*N, MN).

%%<
reduce(Y   : N  : A,
       X\Y : M  : A --> B,
       X   : MN : B)
:-
  bnf(M*N, MN).
```

Figure 5.27: The shift-reduce parser and reduction rules.

```
%% bare NP
null(n  : P : e --> t,
     np : Q^ (P*X, Q*X) : (e --> t) --> t).

%% subj lift
null(s\np : P : e --> e --> t,
     s\np : R : ((e --> t) --> t) --> e --> t)
:-
  bnf(Q^E^ (Q* (X^ P*X*E)), R).

%% obj lift
null(s\np/np : P : e --> e --> e --> t,
     s\np/np : R : ((e --> t) --> t) --> e --> e --> t)
:-
  bnf(Q^Y^E^ (Q* (X^ P*X*Y*E)), R).
```

Figure 5.28: Nullary lifting rules.

5.3.2 The Parser

The parser used to derive the logical forms is a variant of the backtracking shift-reduce parser found in (Steedman, 1991b).[14] As shown in Figure 5.27, at each step the parser can shift, reduce or use a nullary rule; since type-raising and coordination are not covered, only the forward and backward application rules are required.

The categories manipulated by the parser consist of a triple including a syntactic type, a λ-term, and a semantic type. Two categories can be reduced only when the syntactic and semantic types match. The semantics of the resulting category is the β-normal form of the application of the functor term to the argument one, which is derived via an explicit call to the normalization routine `bnf`.

As discussed in section 3.2, lifting rules are employed in the grammar to make nouns into bare NPs and to let verbs combine with generalized quantifiers. In the parser, the applicability of these rules is controlled by the semantic type of the categories in question, as shown by the example rules listed in Figure 5.28. Note that for simplicity, fresh variables are used instead of existentially quantified variables ones.

Finally, some example lexical items appear in Figure 5.29. As noted in the last section, ambiguity in the translation of VP-modifiers has been traded for disjunction in the introduced rule; for example, the progressive is given a unique

[14] Note that this parser has been chosen for its perspicuity, rather than for its efficiency.

```
%% proper nouns
cat(jack, np : jack : e).

%% determiners
cat(a, np/n : P^Q^ (P*X, Q*X) : (e --> t) --> (e --> t) --> t).

%% nouns
cat(carboy,  n : X^ carboy(X) : e --> t).

%% verbs
cat(fill, s\np/np :
          X^A^E^ fill(A, X, E) :
          e --> e --> e --> t).

%% vp modifiers
cat(prog, (s\np) / (s\np) :
          P^X^E^ (P*X*E0, inprog_plus(E0, E)) :
          (e --> e --> t) --> (e --> e --> t)).
cat(at, (s\np) \ (s\np) / time :
        T^P^X^E^ (P*X*E, at(T, E)) :
        e --> (e --> e --> t) --> (e --> e --> t)).
```

Figure 5.29: Example lexical items.

translation using the disjunctive predicate inprog_plus, rather than several lexical entries (again, this difference is not taken to be significant). Otherwise, the lexicon differs little from the one developed in Part I.

5.3.3 Efficiency Concerns

The logic program directly implementing the present calculus works reasonably quickly for small examples. The two most important ways to improve its efficiency appear to be the following ones. First, in order to realize the gains in efficiency made possible in an order-sorted framework, one should either (i) incorporate the resolution of the sort axioms into the unification algorithm, as suggested by Walther (1990) and others, or (ii) investigate more digestible representations—such as those found in (Dale, 1992)—which by collecting constraints on an entity into a single structure to be unified against should have much the same effect. Second, one should use the constraints provided by the temporal adverbials at a much earlier stage, in order to avoid wasting computations involving eventualities on the wrong part of the time line.

5.4 Summary

In this chapter I have presented a calculus of eventualities which covers a subset of the linguistic account developed in Part I. Following Moens (1987), I have shown that the calculus facilitates the implementation of a simple statement verifier which allows for a much greater range of natural language queries than is usually the case with temporal databases. Going beyond Moens, I have also addressed the problem of aspectual composition, and resolved some problematic aspects of his treatment of the imperfective paradox. In particular, I have shown how an explicit representation of subevents allows a natural resolution of Zucchi's paradox, and how the present treatment of process and event expressions resolves Oberlander and Dale's problem in representing different 'perspectives' on the same event. Note, however, that the relative importance of these issues to natural language processing tasks has not been directly addressed; doing so would have required a corpus study of an appropriate task domain, which was beyond the scope of this study.

The calculus has been implemented using a Prolog meta-interpreter which allows both events and states to be basic, and thus provides a greater degree of representational flexibility than the event calculus of Kowalski and Sergot (1986). While the implementation works reasonably quickly for small examples, its efficiency could be improved by taking full advantage of the order-sorted framework and making better use of the constraints provided by the temporal adverbials in the search process.

Chapter 6

A MicroWorld Study of Discourse Interpretation

In this chapter I discuss a preliminary study of the relevance of the spatio-temporal aspects of the model-theoretic analysis to discourse interpretation. In a sense, this chapter begins where the preceding one left off: whereas the preceding chapter did not address the problem of how to extract an event-based representation of a text's meaning, and likewise avoided issues of spatio-temporal representation, these topics form the focus of attention here.

The chapter is organized as follows: section 6.1 discusses the motivations behind the study; section 6.2 presents the most interesting findings discovered in the research carried out to date; section 6.3 focuses on the merits and shortcomings of the constraint-based abductive interpretation procedure utilized in the investigation; finally, section 6.4 summarizes the study and reiterates its conclusions.

6.1 Motivation

In recent years, the issue of how aspect affects discourse interpretation has received increasing attention;[1] quite independently, so has the problem of how to represent the meanings of locative prepositions at a fine-grained level.[2] Al-

[1] Cf. (Dry, 1981; Smith, 1981; Steedman, 1982; Kamp and Rohrer, 1983; Partee, 1984; Reinhart, 1984; Hinrichs, 1986; Dowty, 1986; Eberle, 1992; Moens, 1987; Hinrichs, 1988; Moens and Steedman, 1988; Nakhimovksy, 1988; Webber, 1988; Caenepeel, 1989; Hatav, 1989; Lascarides and Asher, 1991; Lascarides, Asher, and Oberlander, 1992; Hwang and Schubert, 1992; Sandström, 1992; Sandström, 1993).

[2] Cf. (Badler, 1975; Fillmore, 1982; Herskovits, 1986; Thibadeau, 1986; Schirra et al., 1987; Garrod and Sanford, 1988; Retz-Schmidt, 1988; Hays, 1989; Mayer, 1989; Badler et al., 1990; Habel, 1990; Kalita, 1990; Jackendoff and Landau, 1991; Webber et al., 1991; Regier, 1992; Siskind, 1992).

though these two issues have yet to be studied in a single setting, they are clearly related: as we saw in section 4.3, locative prepositions are aspectually significant, in ways consonant with their spatial meanings; if aspect affects discourse interpretation, then so must locative prepositions, at least indirectly.

The aim of this chapter is to examine the relevance of the linguistic account developed in Part I to these two issues; as such, it represents a first attempt to examine them in a unified setting. Because these issues are so large, however, the study is necessarily a preliminary one.

The study is set within the context of the following task: to devise a program which allows a short narrative description within a small English fragment to be used as a specification for a simple microworld animation. Such a program has been sufficiently implemented to illustrate the findings discussed in the next section; details of the implementation appear in section 6.3.

Given the ambiguous and underspecified nature of narrative specifications, the present task context requires one to investigate how to balance defaults and preferences with both linguistically and situationally derived constraints. As such, it serves as an excellent framework for examining the fine-grained meanings of locative prepositions and the communicative functionality of aspect in narrative discourse.

The particular microworld employed in the study is shown in Figure 6.1. The protagonist is Dante, an affable sort who tries to avoid getting shot by his rather dull antagonist, The IRCS (an evil triangle ship). Dante can seek protection behind any of the three islands, or behind one of the available mines. He can also pump goo into the mines, which protects them from The IRCS' shells. Finally, Dante can try to get rid of the IRCS by leading her[3] into a mine.[4]

In outline form, the program works as follows. The initial scene is specified independently, including the locations, orientations, and so forth of the movable objects. The narrative is then interpreted to produce a sequence of events grounded in this initial scene, using the procedure described in section 6.3. For each event explicitly mentioned in the text, there are often several unmentioned events inferred to take place; consequently, the resulting collection generally contains numerous temporally overlapping events. The entire collection of events is then used to build the sequence of frames that make up the animation, using parametric interpolation to compute the location, orientation, and so on of each movable object at each time slice.[5]

An example output animation is pictured in Figure 6.2, for the initial scene shown in the first frame and the narrative specification listed in the caption

[3]The use of feminine pronoun for The IRCS is a mere convenience in accord with nautical tradition, and should not be interpreted otherwise.

[4]Note that not all of these elements of the microworld have been realized in the current prototype implementation.

[5]For simplicity, linear interpolation is used to compute the inbetween values, though more sophisticated techniques could also be employed.

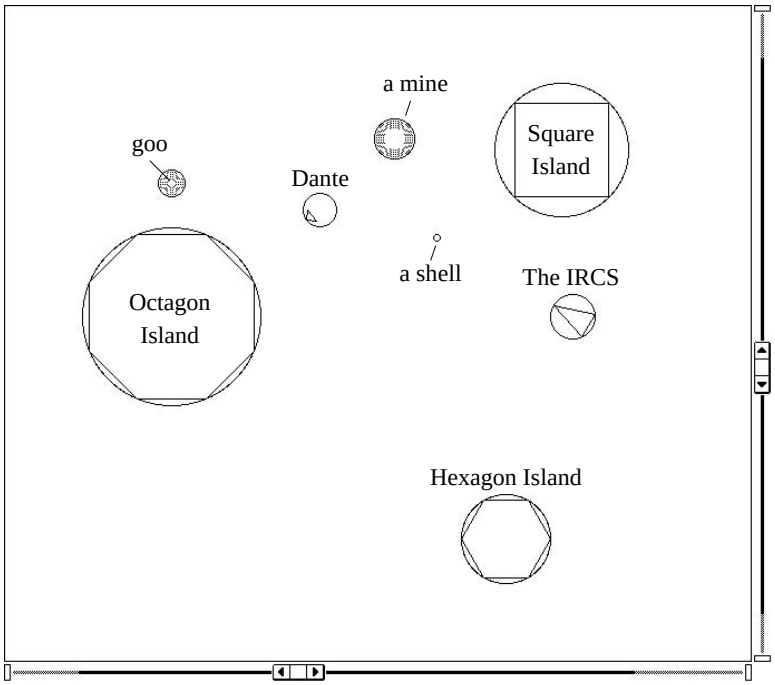


Figure 6.1: The microworld.

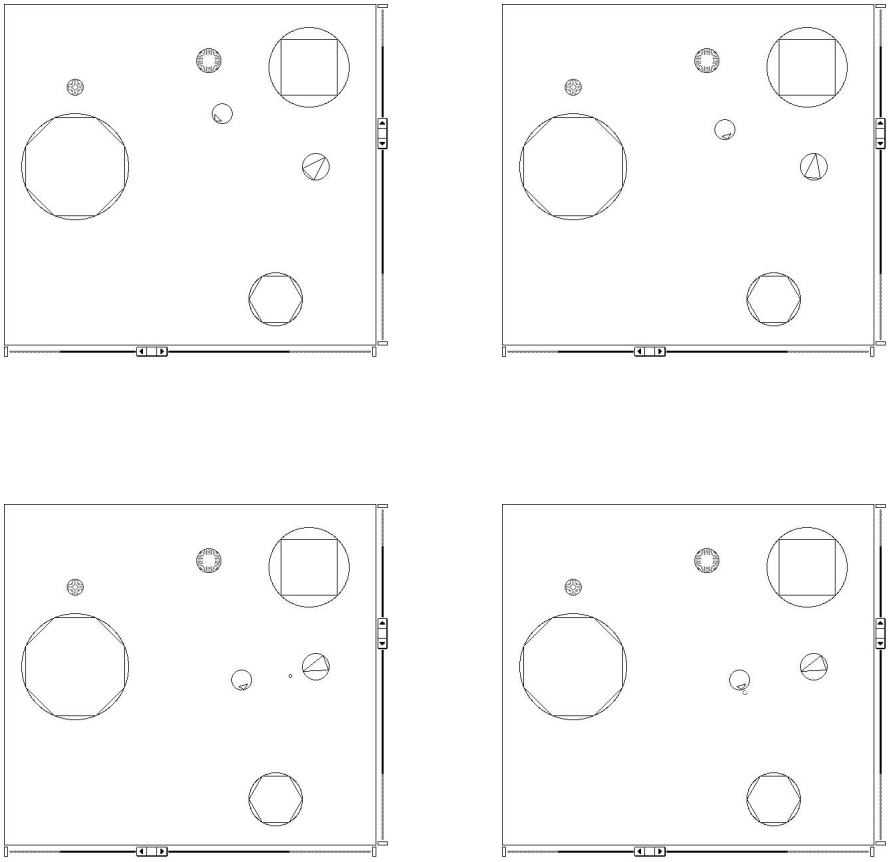


Figure 6.2: The animation specified by *Dante zipped towards Hexagon Island. When she fired, he stopped. The shell missed him.*

(these frames are just four of many). In addition to the explicitly mentioned events, the interpretive process also yields an event of Dante turning around, an event of the IRCS aiming, an event of the shell being launched, and an event in which the shell explodes. Note that many of these events overlap temporally; for example, Dante turns around and starts moving towards Hexagon Island at the same time as The IRCS turns to aim.

In aiming at Dante, The IRCS has two options: to aim directly, or to lead him according to his current velocity. In this example, The IRCS misses because she chooses to lead him, and he stops in time. To see the other possible outcome, the user could simply change the verb *miss* to *hit*; in this case, the input specification would force The IRCS to take the alternative strategy of aiming directly, which would foil Dante's attempt to dodge the shell.

It is worth noting at this point that The IRCS' aiming strategy does not always determine whether she hits or misses Dante. For example, consider the animation pictured in Figure 6.3, for the input specification listed therein. Because Dante is moving directly away from The IRCS, it does not matter which aiming strategy she chooses—where she points when she fires will be the same in either case. For this reason, if the user changes the input specification by replacing *miss* with *hit*, an error will be detected in the interpretation process, and the user will be informed that no hitting event could be constructed. Conversely, consider now the example pictured in Figure 6.4. Because of the relatively slow speed and shallow angle involved in this situation, both aiming strategies will result in The IRCS hitting Dante, and thus replacing *hit* by *miss* here will again lead to a detectable error.

These three examples point to the interest in examining the fine-grained meanings of locative prepositions in a dynamic context, as is done in this preliminary study. It is only by grounding descriptions in an actual scene that these qualitatively different cases can be discovered—and doing so, of course, requires appropriately differentiating the meanings of terms such as *towards* and *away from*.

6.2 Findings

6.2.1 Discourse Interpretation and the Event/Process Distinction

Since the work of Hinrichs (1986), Kamp and Rohrer (1983) and Partee (1984), aspectual class has been recognized as an important factor in discourse interpretation, especially in the case of narrative.[6] The best studied distinction is that between stative and eventive sentences: whereas sentences describing events tend to carry the narrative forward in time, those describing states sentences

[6] Although more recent work stresses the primacy of discourse relations, aspectual class has continued to play a significant role.

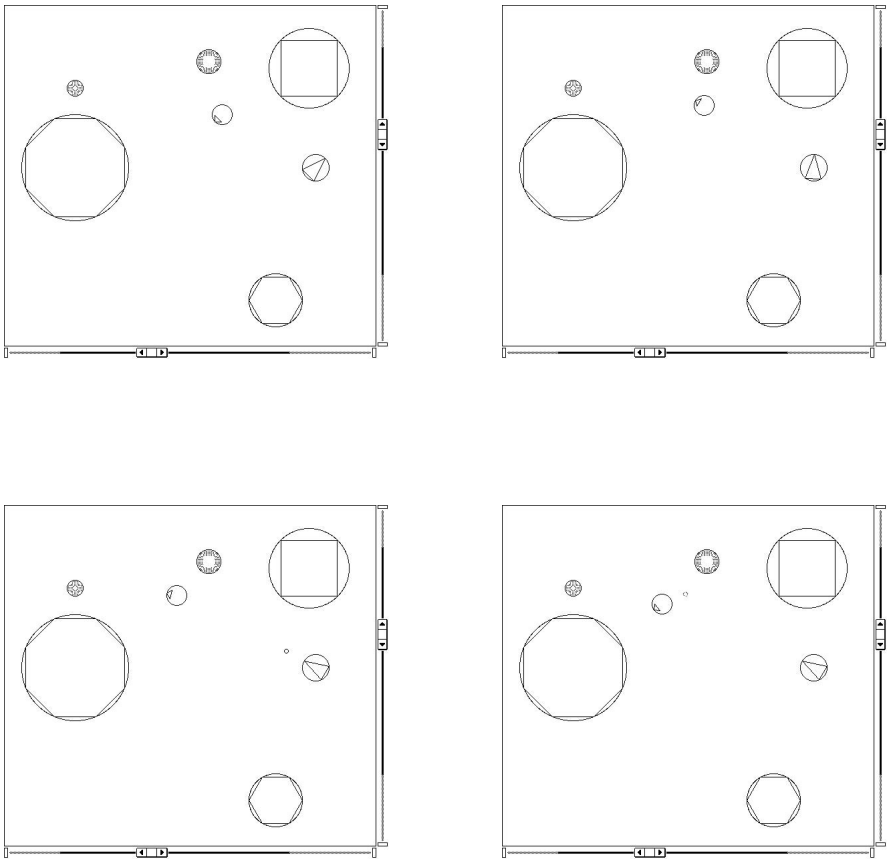


Figure 6.3: Four frames from the animation specified by *Dante zipped away from The IRCS. When she fired, he jetted towards Octagon Island. The shell missed him.*

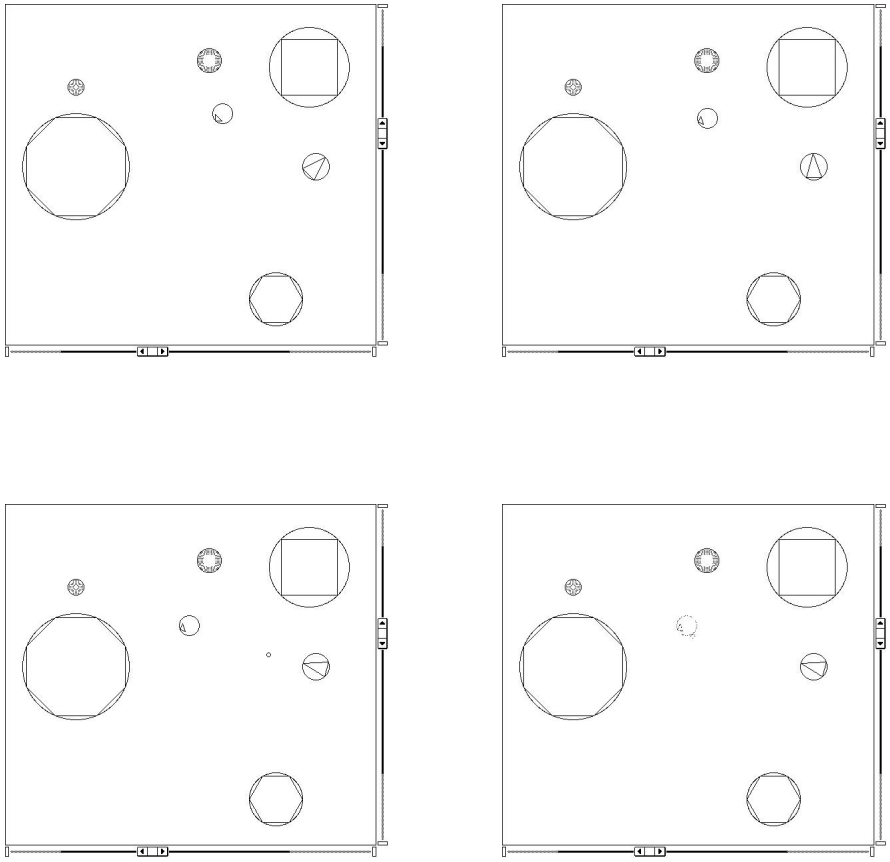


Figure 6.4: The animation specified by *Dante glided towards Octagon Island. When she fired, he stopped. The shell hit him.*

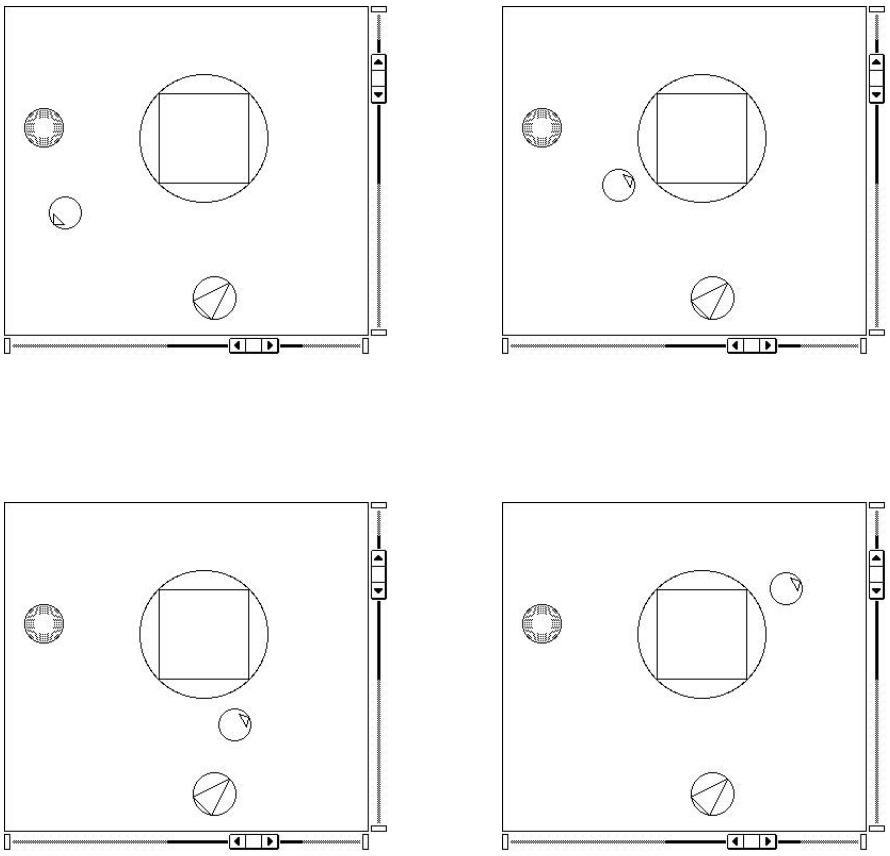


Figure 6.5: The animation specified by *Dante glided over to Square Island. Then he zipped around it.*

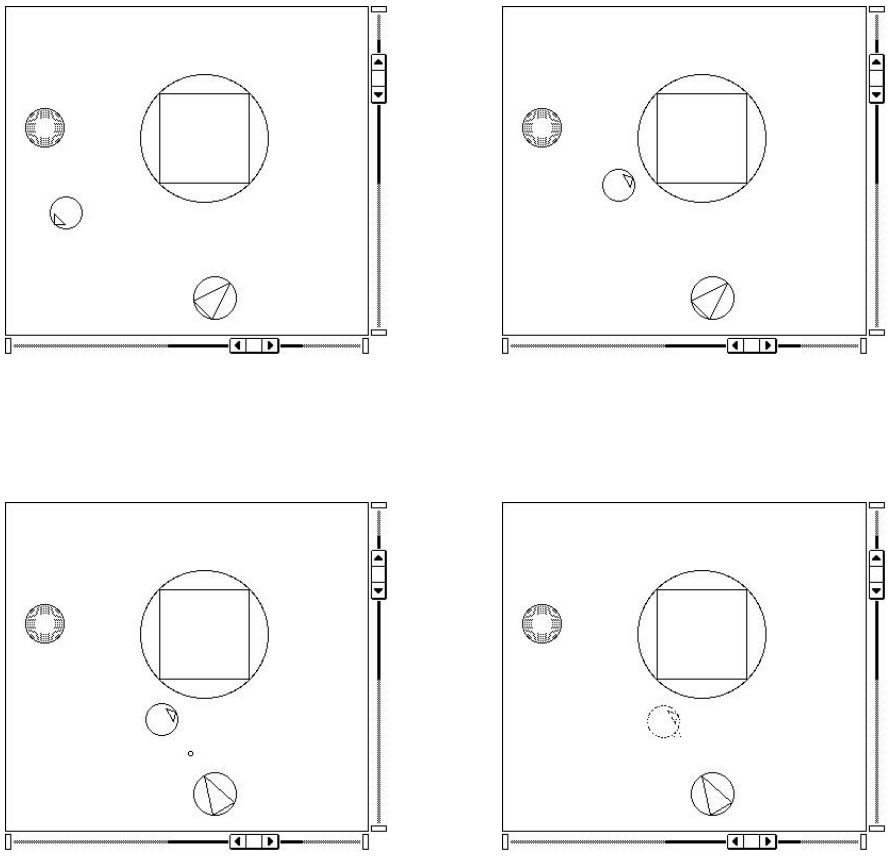


Figure 6.6: The animation specified by *Dante glided over to Square Island. When he started zipping around it, The IRCS fired. The shell hit him.*

generally do not, providing background information instead. Much less attention has been paid, in contrast, to the event/process distinction, which turned out to be more relevant to the present study.

As has often been observed, process expressions are like stative ones in that they can be used to provide background information; in the present context, however, their most natural use is under their inchoative or onset readings. For example, consider again the narrative specification listed in Figure 6.2. In the first sentence, the process expression *Dante zipped towards Hexagon Island* is interpreted as meaning that Dante starts zipping towards Hexagon Island at the scene-initial time; as a result, the next sentence is interpreted as following the onset of Dante's motion (rather than its completion). Note that the capacity to introduce onset events is essential if one wants to specify concurrent events via a narrative sequence: if The IRCS' firing had to be interpreted as following the completion of Dante's motion, the scene would be much less interesting.

With event expressions, inchoative readings are not possible, in contrast to the preceding case. For example, consider the specification for the animation pictured in Figure 6.5. Because the locative prepositions *over to* and *around*[7] only yield event expressions (unlike *towards*), the two sentences are necessarily interpreted as describing a completed sequence. If the user wants to add an event in the middle of this sequence, an onset event must therefore be mentioned explicitly; of course, introducing such an onset event opens up the possibility that the process which it begins never culminates (the imperfective paradox), as is the case in the example pictured in Figure 6.6.

6.2.2 Definite Reference and Incremental Interpretation

Haddock (1987) points out a curious problem for theories of definite reference, which goes roughly as follows (Haddock's example actually involves rabbits and hats). Suppose one wants to refer to Dante in the context pictured in Figure 6.7, but cannot remember his name. One way to refer to him would be to use the definite NP *the character near the shore*, since Dante is much closer to the shore of Square Island than is The IRCS. What is curious about this sort of definite expression is that it seems perfectly natural despite the fact that the embedded definite NP (*the shore*) does not refer uniquely. Intuitively, this NP is deemed acceptable because once one realizes that the NP is supposed to refer to a character near some shore, the set of contextually appropriate shores is implicitly restricted to those that have a character near them, which is in fact a singleton set.

To account for this sort of example, Haddock proposes a model of interpretation whereby constraints on referents are built up incrementally, along the lines suggested above. Interestingly, this same technique appears to be appropriate for more difficult examples requiring a fine-grained treatment of locative prepo-

[7] Here *around* is interpreted as meaning around to the other side.

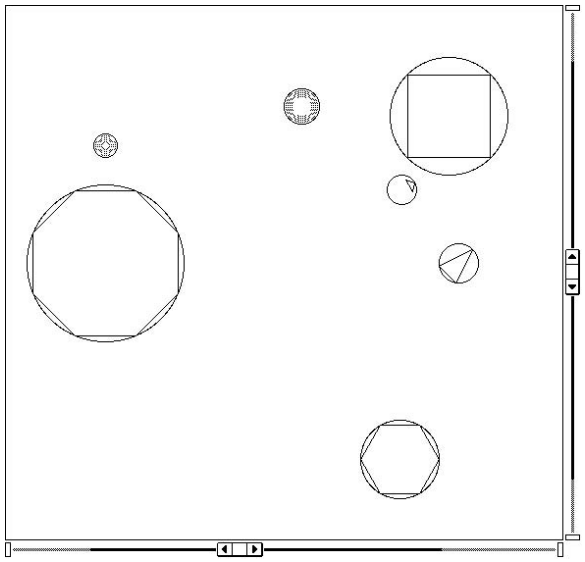


Figure 6.7: The context for interpreting the definite NP *the character near the shore*.

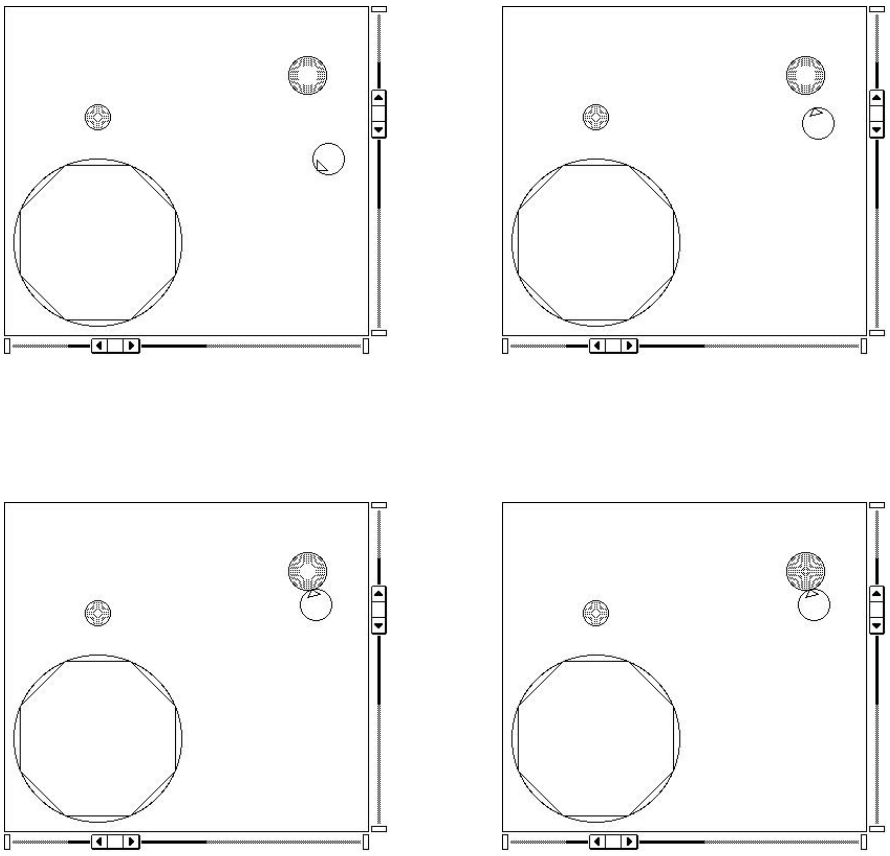


Figure 6.8: The animation specified by *Dante glided up to the mine. Then he filled it with goo.*

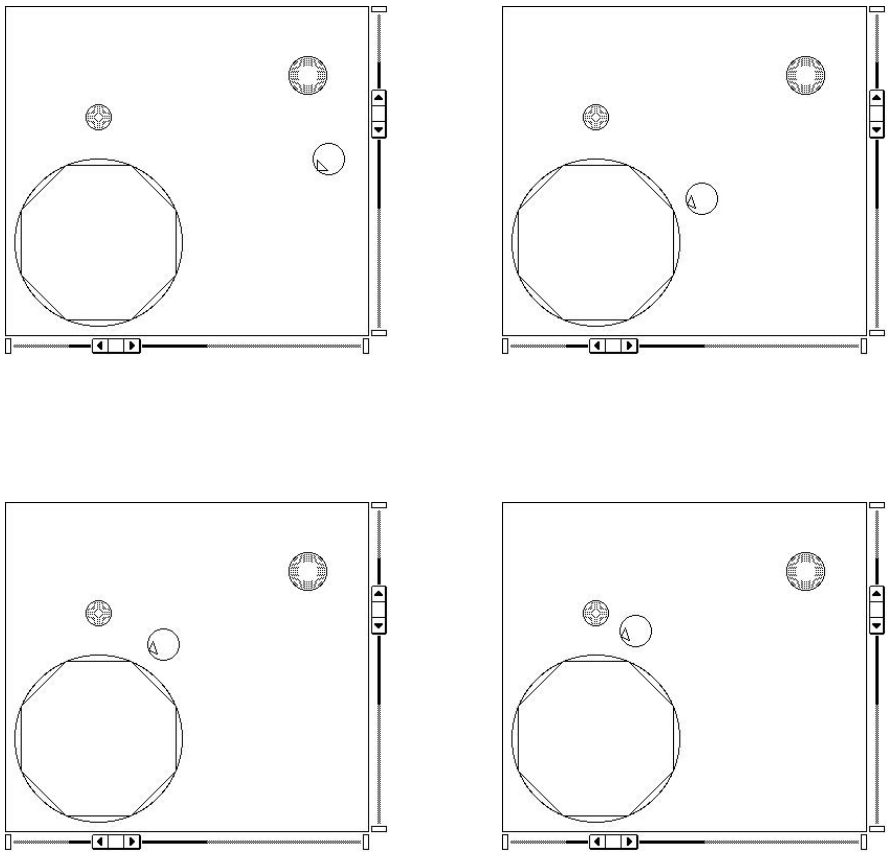


Figure 6.9: The animation specified by *Dante zipped over to Octagon Island. Then he glided along the shore until he reached the mine.*

sitions in a dynamic context. Consider first the specification listed in Figure 6.8. Because the preposition *up to* is taken to specify a path which begins near the reference object (and ends in contact with it), the smaller mine may be removed from further consideration, which then satisfies the uniqueness requirement of the definite NP *the mine*.[8] Now consider the specification listed in Figure 6.9. In this example, the preposition *along* specifies a path which follows the Octagon Island shoreline; because the larger mine is offshore, it could not possibly be reached by such a path, and thus can safely be eliminated as a candidate referent. As a result, the definite NP *the mine* again refers uniquely, but this time to a different mine. It is important to note that in both of these examples, it is only by grounding the constraints supplied by these path prepositions in the initial context that the definite references may be successfully resolved.

6.2.3 Equivalence Classes and Path Representation

As we saw in section 4.3, the mathematical notion of equivalence class can be usefully employed in formalizing paths, in the following two ways:

1. As Habel (1990) suggests, by taking paths to be equivalence classes of functions from times to locations, extraneous information about velocity can be eliminated, while still preserving information such as sequential order and direction.

2. By taking undelimited paths to be equivalence classes of their delimited counterparts, extraneous information about specific endpoints can be eliminated, while still preserving information such as direction and proximal distance. Moreover, this treatment allows undelimited paths to optionally possess maximal or minimal points.

At this point the reader might be troubled by the perception that employing equivalence classes in this way seems to add complexity in order to achieve simplicity. This apparent paradox can be resolved by observing that equivalence classes can often be represented at least as succinctly as their individual elements, if not more so. For example, consider the use of a tree to represent a derivation in a context-free grammar. Since, strictly speaking, derivations in a context-free grammar are defined to be sequences of rule applications, such a tree should be considered an equivalence class of derivations, where each element of the equivalence class imposes a particular ordering on the rule applications. Suppose now that we want to represent a particular element of this equivalence class; if we begin with the derivation tree, doing so would actually require adding information to the representation, namely a sequential ordering on the non-terminal nodes.

[8] Note that this uniqueness requirement is not actually enforced in the current implementation.

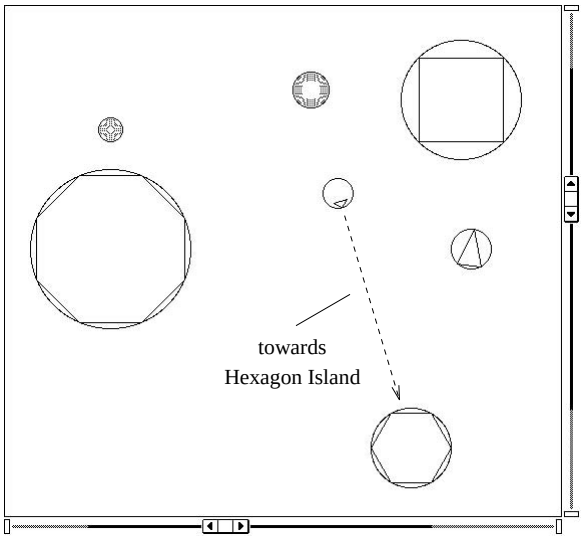


Figure 6.10: Representing an undelimited path towards Hexagon Island.

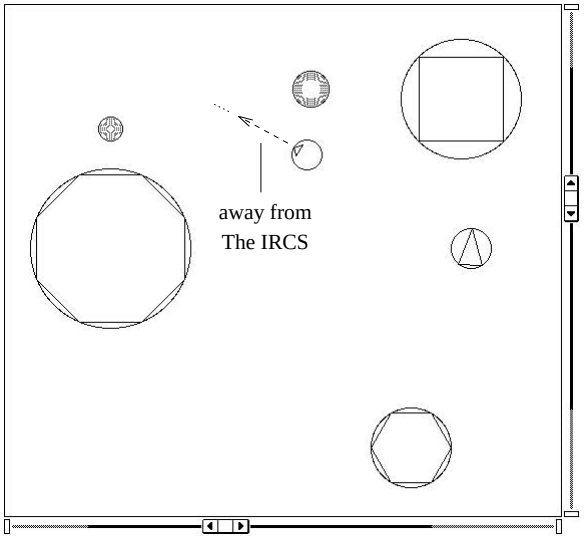


Figure 6.11: Representing an undelimited path away from The IRCS.

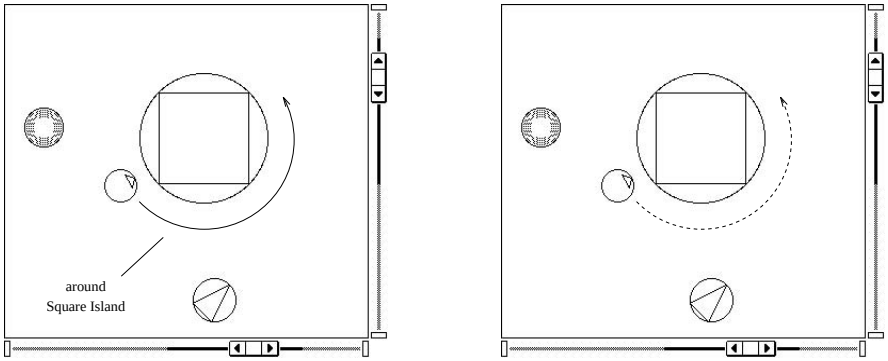


Figure 6.12: A delimited path around Square Island, and its undelimited counterpart.

As might be expected, a similar situation arises in representing paths and eventualities. In the case of delimited paths, this should already be familiar; for concreteness, let us take the case of polylines, as is done in the current implementation. Representing a path using a polyline requires no more than a simple list of points; to represent a motion event involving this path, particular times must be paired with these points, in accordance with the speed of the agent or other factors of interest.

Turning now to the undelimited case, let us first examine the directional prepositions *towards* and *away from*. Figure 6.10 depicts the path specified by the PP *towards Hexagon Island*, as used in the example in Figure 6.2; following the conventions of the preceding chapter, a dashed line is used to indicate its undelimited status. This path may be represented using just a minimal point and maximal point, where the minimal point is Dante's initial location and the maximal point where he would reach Hexagon Island. This representation turns out to be no more complex than the minimal representation of the delimited path actually traversed, which has a start point equal to the minimal point and an end point about halfway to Hexagon Island.

Consider now the path specified by the PP *away from The IRCS*, as used in the example in Figure 6.3. Unlike the previous case, this path does not have a maximal point; to suggest this, the dashed line is extended by three dots in Figure 6.11. Despite the fact that this path represents an unbounded collection of delimited paths, representing it again requires just two pieces of information: a minimal point, as before, and a direction, calculated using the vector from the IRCS' initial location to Dante's initial location.

The preceding two examples both involved mapping an undelimited path to one of its delimited counterparts, once the specific endpoints became known. To complete this section, let us now reconsider the example in Figure 6.6, which additionally involves the reverse mapping. The path specified by the PP *around Square Island* is depicted in the left portion of Figure 6.12 (again following the conventions of the preceding chapter, a solid line is used to indicate its delimited status); this path is the one involved in the event specified by *Dante zip around Square Island*. Since the aspectual verb *start* requires a process as input, this event must first be mapped to the process which makes it up; it is this process whose onset is then asserted to actually occur. As explained in section 4.3, this event-to-process mapping concomitantly maps the delimited path involved in the event to the undelimited one involved in the process; the resulting undelimited path is depicted in the right portion of Figure 6.12. In representational terms, this mapping simply equates the start and end points of the delimited path with the minimal and maximal points of the undelimited one (resp.), leaving the intermediate points the same. Once this mapping is completed, the resulting undelimited path can then be mapped to one of its delimited counterparts, just as in the preceding examples; of course, this delimited path need not be the maximal one, as the example in Figure 6.6 demonstrates.[9]

6.3 Implementation

6.3.1 Interpretation as Abduction

As a mode of inference, abduction is best understood in comparison to deduction and induction:[10]

$$\frac{p \rightarrow q \quad p}{q} \text{ Deduction} \qquad \frac{p \quad q}{p \rightarrow q} \text{ Induction}$$

$$\frac{q \quad p \rightarrow q}{p} \text{ Abduction}$$

In deduction, one infers the result from the rule and the case. In induction, from the case and the result—or from many such instances—one infers the rule. Abduction is the third possiblity: from the result and the rule, one infers the case.

[9] Note that this case requires eliminating some of the intermediate points, since the path is not a straight one.

[10] This brief introduction is adapted from (Kakas, Kowalski, and Toni, 1992) and (Hobbs et al., 1993).

Abduction can be viewed as inference to the best explanation. Given the evidence q, one infers the underlying cause or explanation p, using the rule $p \rightarrow q$. For example, suppose the evidence q is the proposition that it is wet outside, and $p \rightarrow q$ the rule that if it rains then it becomes wet outside: from these we may abductively infer the proposition p, i.e. that it must have rained. Of course, this mode of inference is not a valid one—someone might have turned on the sprinkler instead. As such, abduction necessarily involves choosing amongst various alternatives, according to criteria such as likelihood and consistency with the rest of what one knows.

In (Hobbs et al., 1993) and earlier work, Hobbs and his colleagues have argued that their approach to abductive inference results in a significant simplification of how the problem of interpreting texts is conceptualized. For them, the interpretation of a text is the minimal explanation of why the text would be true. This suggests an interpretation procedure where one abductively proves the logical form of the text from what is mutually known, merging redundancies where possible and making assumptions when necessary.

The purported advantages of their approach include its declarative and unifying nature and its flexible treatment of given and new information. Regarding the former, Hobbs and his colleagues note that their view of **interpretation as abduction** can be combined with the older view of **parsing as deduction** (Kowalski, 1979; Pereira and Warren, 1983) to yield an elegant and thorough integration of syntax, semantics and pragmatics. As for the latter, they suggest that in seeking to minimize the new information which must be assumed, their approach is in the spirit of Crain and Steedman's (1985) view of discourse.

Before continuing, let us briefly look at an example of how an abductive interpretation procedure allows for a flexible treatment of given and new information. Consider the following pair:

(6.1) (a) Jack jogged from the bridge to the museum. He left at noon and arrived thirty minutes later.

(b) Jack jogged from the bridge to the museum. He left the museum at one and arrived at the cafe sweaty and ten minutes late.

In (6.1a), the leaving and arriving events mentioned in the second sentence are most naturally interpreted as providing further information about the jogging event mentioned in the first sentence. In contrast, the leaving and arriving events in (6.1b) cannot be interpreted in this manner; instead, they must be interpreted as newly mentioned events.

This pattern can be made to fall out of the abductive interpretation procedure in the following way. Suppose each verb introduces a predicate over an eventuality variable into the logical form, together with some cost for assuming the predicate as new information. Suppose further that we have rules which state that an event of going from one place to another entails the existence of an event of leaving the first place and an event of arriving at the second. Given

this, the leaving and arriving predicates in (6.1a) can be proved by backchaining on the rules linking these events to the jogging event introduced in the first sentence; since proofs grounded in known facts cost nothing, this is the preferred interpretation. In (6.1b), however, these rules cannot be used, since the points of departure and arrival mentioned in the second sentence are inconsistent with those of the jogging event introduced in the first; for this reason, the higher-cost proof where these events are taken to be newly mentioned is the only available one.

6.3.2 Abduction Plus Constraints

Because of the compelling arguments for the 'interpretation as abduction' view put forth by Hobbs and his colleagues, it seemed worthwhile to investigate their approach within the confines of the present study. For this reason, the present implementation is based upon an extended version of their abductive inference procedure, as detailed in Stickel (1990).

Stickel's abductive theorem prover differs from standard depth-first iterative deepening Prolog meta-interpreters in three principal ways: first, it marks each goal with an assumption cost, which may be passed on to its subgoals according to a weighted scheme; second, it allows goals to be assumed into the context, which increases the total cost of the proof by the assumption cost; and third, its iterative deepening is cost-based rather than depth-based. Although Stickel stresses the importance of efficient but partial checks for consistency, these are not integrated into his abductive inference system. This is in contrast to all of the approaches discussed in Kakas, Kowalski and Toni's (1992) review paper, which interleave abduction with the incremental checking of integrity constraints.[11]

For convenience, a somewhat simpler approach to consistency checking is taken here, namely to incorporate some constraint logic programming techniques into the inference system, along the lines of Jaffar and Michaylov (1987). The idea is to add to the proof context a store of unsolved constraints, which are checked after each proof step. In this way constraints are built up incrementally, and solved as soon as the variables are sufficiently instantiated; furthermore, backtracking occurs as soon as an inconsistency is detected.

In addition to providing a simple mechanism for partial consistency checks, the incremental constraint solving mechanism provides a means for grounding the events in the microworld context. For example, consider the rules shown in Figure 6.13. The first rule is for the path predicate away_from, which takes an object X (the object in motion) and a reference object R as parameters. This rule must be grounded at a particular time, of course, since the objects may change location; rather than letting this time be an extra parameter, the

[11] Cf. Pitt, den Bergh, and Cunningham (1992) for an interesting alternative approach to discourse interpretation based upon the integrity checking techniques developed in Sadri and Kowalski (1988).

```
away_from(X, R, Path) :-
  direction(Path, A), min_point(Path, PX),
  current_rt(RT),
  location_at_time(X, RT, PX),
  location_at_time(R, RT, PR),
  from_to_dir_dist(PR, PX, A, _).

from_to_dir_dist(P0, P1, Dir, Dist) :-
  dist_from_to(P0, P1, Dist),
  to_from_dir_dist(P0, Dir, Dist, P1),
  dir_from_to_dist(P0, P1, Dist, Dir).

dist_from_to(pt(X0,Y0), pt(X1,Y1), pts(N)) :-
  N = sqrt(sqr(X1 - X0) + sqr(Y1 - Y0)).
to_from_dir_dist(pt(X0, Y0), nrads(R), pts(N), pt(X1, Y1)) :-
  X1 = cos(R) * N + X0, Y1 = sin(R) * N + Y0.
dir_from_to_dist(pt(X0,Y0), pt(X1,Y1), pts(N), nrads(R)) :-
  X = X1 - X0, Y = Y1 - Y0, R = acos(X / N) * pol(Y).
```

Figure 6.13: Grounding the path predicate away_from in context.

current reference time RT is used instead.[12] As described in the last section, the away_from rule sets the minimal point of the path to the current location PX of the object in motion, and the direction of the path to the angle A determined by the vector from PR to PX, where PR is the refence object's current location. This value is computed using the auxiliary goal from_to_dir_dist, which builds up enough local constraints to solve for any one of its four arguments from the other three.[13]

As we saw in the examples in Figures 6.3 and 6.4, sometimes the constraints derived from the textual specification cannot be satisfied. In this case it is obviously preferable to relax some of the constraints rather than to fail to find any interpretation at all. By introducing rules with error predicates (at some non-negligible cost), the abductive interpretation procedure can be made to progressively relax particular constraints.[14] As an example, consider again the specification in Figure 6.3. Recall that because of the geometry of the situation, no sequence of events can be found which satisfies the specification where the verb *miss* is replaced by *hit*. This specification can still be given an interpretation, however, if the user supplies a rule which allows the event predicate translating *hit* to be abductively proved by first assuming an error predicate and then proving the event predicate translating *miss* instead.[15]

6.3.3 Limitations and Problems

The current prototype implementation is quite limited in terms of what information is represented in the rule base. For example, no attempt has been made to seriously represent the discourse structure or to systematically capture the normal patterns of behavior of the characters in the microworld. In future work, it would be interesting to find out if the present abductive interpretation procedure could easily accomodate rules encoding this information, as Hobbs and his colleagues suggest is the case.

Another interesting question that arises in this regard is whether a logical approach employing a relatively simple level of geometric reasoning—such as the present one—can be successfully integrated with more sophisticated approaches to path planning. In recent work, Badler, Phillips, and Webber (1993) and Badler et al. (1994) have demonstrated the advantages of employing numerical (behavioral, reactive) techniques in planning the motion of complex figures in dynamic environments. This might lead one to think that geometric reasoning should be eschewed altogether in the process of discourse interpretation; nevertheless, the fact that geometric reasoning can interact with reference reso-

[12] The reference time is updated with each new event added to the discourse context.

[13] Note that it is often useful to provide redundant specifications, since the constraints are only checked locally (and thus efficiently).

[14] This is similar to the treatment of ill-formed utterances in (Hobbs et al., 1993).

[15] Of course, generalizing the treatment of this type of error would require an explicit treatment of negation, or at least antonyms.

lution, as shown in section 6.2, suggests the utility of pursuing such an eventual integration.

While these particular limitations could perhaps be overcome, it should be emphasized that there is still reason to doubt the ultimate viability of the 'interpretation as abduction' approach. This is because there are two serious problems which have yet to be adequately addressed; while both of these problems are acknowledged by Hobbs and his colleagues, neither has been definitely resolved.

The first problem is the inefficiency of the cost-based abduction scheme. While Hobbs and his colleagues do suggest three ways in which their scheme can be made more efficient, they do not address the fundamental intractability of searching for a globally optimal discourse interpretation.[16] This problem was a significant one even for the present small-scale implementation. To mitigate this problem, the assumption cost of predicates introducing new information was reduced to a small but non-zero amount. This yielded a three-way distinction in proofs: those with zero cost, those with negligible cost, and those with a significant cost. The zero-cost proofs turned out to be useful for computing intermediate locations and other implicitly determined parameter values without adding any new information; the negligible-cost and significant-cost proofs were used to distinguish those without errors and those with errors, respectively. Note, however, that this technique is not compatible with the flexible accomodation of given and new information; it was only because of the small range of texts considered that the technique worked in the present study.

The second problem concerns the semantics of the assumption costs. To date the best proposal is that of Charniak and Shimony (1990), who provide a probabilistic semantics for the propositional case of cost-based abduction. While their semantics seems entirely appropriate for applications of abduction such as diagnostic reasoning, it does not suffice to explain what the numbers mean in abductive natural language interpretation. The problem stems from their assumption that a proposition always has the same cost, wherever it occurs in the inference process. This assumption is a reasonable one for them, since they only concern themselves with the probability that a proposition is true. In the interpretive scheme of Hobbs and his colleagues, however, high assumption costs are sometimes equated with the disutility of not proving a certain proposition, rather than its improbability; this is done to avoid assuming what should be given information. On the surface, at least, it seems that the use of assumption costs for both probabilistic reasoning and flexible information accomodation leads one to conflate two different metrics; whether or not this problem can be successfully resolved remains to be seen.

[16] In (Hobbs et al., 1988), this problem is mentioned as one in need of further study.

6.4 Summary

In this chapter I have reviewed a preliminary study of the relevance of the spatio-temporal aspects of the linguistic account to discourse interpretation. This study is set within the context of devising a program which allows a short narrative description within a small English fragment to be used as a specification for a simple microworld animation.

The findings of the study are threefold:

- The event/process distinction, which depends on the choice of locative preposition, can be used to discern complete events from those that mark the onset of a particular process; moreover, onset events, whether explicitly mentioned or implict, are particulary useful in specifying temporally overlapping events within the confines of a narrative sequence.

- The effect of incremental interpretation upon definite reference first observed by Haddock (1987) also appears in more difficult examples requiring a fine-grained treatment of locative prepositions in a dynamic context. For example, in *Dante glided along the shore until he reached the mine*, determining which of the mines present in the context is the intended one involves calculating which one could be reached by following the specified path.

- The use of equivalence classes in formalizing paths, which might seem to add complexity to achieve simplicity, supports representations which are at least as succinct for the equivalence classes as for their individual elements.

These findings have been illustrated by a program implemented using a constraint-based abductive interpretation procedure extending that of Stickel (1990); as such it falls within the **interpretation as abduction** paradigm set forth by Hobbs and his colleagues (1993). In brief, the algorithm seeks to balance defaults and preferences with both linguistically and situationally derived constraints in order to yield a collection of eventualities which fully determines the animation.

While the approach holds some promise, its ultimate viability suffers from two unresolved problems with this paradigm, namely the intractability of global optimization and the lack of an adequate semantics for the assumption costs. Furthermore, it also remains to be seen to what extent the approach can be successfully generalized; in particular, it is an open question whether a logical approach employing a relatively simple level of geometric reasoning can be successfully integrated with more sophisticated approaches to path planning involving numerical (behavioral, reactive) techniques, as in e.g. Badler, Phillips, and Webber (1993) and Badler et al. (1994).

Chapter 7

Epilogue

In recent years, it has become common in the linguistics and philosophy literature to assume that **events** and **processes** are ontologically distinct entities, on a par with **objects** and **substances**. At the same time, the idea that time-based (episodic) knowledge should be represented as a collection of interrelated **eventualities** has gained increasing acceptance in the computational linguistics and artificial intelligence literature.

Contrary to what one might expect, a search through the prior literature in linguistics and philosophy reveals no account in which these sortal distinctions play a direct role in adequately explaining the problem of **aspectual composition** and the closely related **imperfective paradox**. In the computational linguistics and artificial intelligence literature, moreover, relatively little attention has been paid to either problem.

The first part of the dissertation investigates, from a model-theoretic perspective, the hypothesis that the parallel ontological distinctions introduced above may be directly employed in an explanatory formal account of the problem of aspectual composition and the imperfective paradox. The aim of this part is to show that a sortal account, developed in the spirit of the eventuality-based work on temporal (episodic) representation, can indeed be explanatory. The second part of the dissertation explores the potential computational applications of the linguistic account, by way of two case studies; here the aim is to begin to demonstrate the suitability of the model-theoretic analysis to computational purposes.

Part I develops a synthesis of proposals by Hinrichs (1985), Krifka (1989; 1992) and Jackendoff (1991) which makes correct predictions in many cases not considered by these authors (cf. section 4.9). In particular, the account is the first to adequately explain the syntactic and semantic behavior of **non-individuating accomplishment expressions**, such as *Jack pour some amount of wort into the carboy*. These expressions are troublesome for earlier analyses which mistakenly assume that tests for individuation correctly identify the class

of accomplishment expressions; because the present account does not rely on such tests, these expressions are unproblematic.

Of the various components of the account, two merit explicit mention here. The first is its use of an **order-sorted logic** as the translation language, which enables one to distinguish formulas that are merely false from those that are not well-sorted, and thus semantically anomalous. The second is its treatment of delimitedness, i.e. that feature which distinguishes substances and processes (undelimited) from objects and events (delimited). In the present account, the undelimited entities—viz., the substances, processes, and their plural and path analogues—are given a natural and uniform treatment in terms of **equivalence classes**, a notion that also plays a central role in the present adaptation of Habel's (1990) approach to formalizing paths. In a nutshell, the idea is simply to treat an undelimited entity as a reified continuum of its delimited counterparts. For example, rather than letting a substance (i.e., an entity of sort substance) be a particular quantity of stuff (here, an entity of sort object), as in e.g. Bach (1986), the present account requires a substance to be a continuum of such quantities; similarly, a process (i.e., an entity of sort process) is not taken to be an eventuality of particular duration (here, an event) but a continuum of such eventualities. This abstract treatment of the generalized count/mass distinction enables the definition of the broadly applicable notion of an **incremental thematic relation**—i.e. a relation that links a continuum in the eventuality domain with one in either the material domain or path domain—which forms the basis of the present explanation of the problem of non-individuating accomplishment expressions and many other interesting cases.

The first case study in part II follows Moens (1987) in showing how a **calculus of eventualities** can facilitate the implementation of a simple statement verifier which allows for a much greater range of natural language queries than is usually the case with temporal databases; it also goes beyond Moens to address the problem of aspectual composition, and resolves some problematic aspects of his treatment of the imperfective paradox (cf. section 5.4). The calculus has been implemented using a Prolog meta-interpreter which allows both events and states to be basic, thereby providing a greater degree of representational flexibility than the event calculus of Kowalski and Sergot (1986).

The second, more preliminary case study examines the relevance of the model-theoretic analysis to discourse interpretation. This study is set within the context of devising a program which allows a short narrative description within a small English fragment to be used as a specification for a simple microworld animation. The findings of the study are threefold (cf. section 6.4): first, the event/process distinction plays a significant role in increasing the efficiency and expressive possibilities of a narrative sequence; second, the effect of incremental interpretation upon definite reference first observed by Haddock (1987) also appears in more difficult examples requiring a fine-grained treatment of locative prepositions in a dynamic context; and third, the use of equivalence classes in formalizing paths, which might seem to add complexity to achieve simplic-

ity, supports representations which are at least as succinct for the equivalence classes as for their individual elements. These findings have been illustrated by a program implemented using a constraint-based abductive interpretation procedure falling within the **interpretation as abduction** paradigm set forth by Hobbs et al. (1993).

Naturally, many interesting questions remain for further inquiry. As discussed in section 4.2, the present treatment of undelimited entities turns out to be quite similar to Hinrichs' treatment of Carlsonian **kinds**—as such, it would do much to bolster the present approach if it could be shown to be compatible with recent work on **genericity**, where abstract kinds are usually employed. Another interesting question that arises in this regard is how the present account could be made to fit with recent work on dynamically-changing discourse contexts; not surprisingly, there are several places in the account where a static semantics turns out to be inadequate to explain the phenomena at hand, the most notable one being the interaction of *in*-adverbials with activity expressions.

On the computational side, a corpus study of an appropriate task domain would be quite useful in establishing the relative importance of the problem of aspectual composition and the imperfective paradox to natural language processing tasks. Finally, it would also be interesting to see to what extent representations such as those found in (Dale, 1992), which collect constraints on an entity into a single, easily digested structure, could also realize the gains in efficiency promised by the order-sorted framework.

Appendix A

Eventuality Calculus Code

The code for the calculus of eventualities consists of a set of meta-interpreted Horn clauses, represented using the entailment operator `<-:`. The meta-interpreter, which is the same one used in the microworld implementation, appears separately in appendix C. The parser and lexicon, described in chapter 5, have been omitted.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                                eventualities
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
eventuality(E) <-: stative(E, _).

%% named subsorts
state(E)     <-: stative(E, +).
canonical(E) <-: state(E), delimited(E, -).
moment(E)    <-: state(E), delimited(E, +), dimension(E, 0).
period(E)    <-: state(E), delimited(E, +), dimension(E, 1).
non_state(E)    <-: stative(E, -).
process(E)      <-: non_state(E), delimited(E, -).
event(E)        <-: non_state(E), delimited(E, +).
momentaneous(E) <-: event(E), dimension(E, 0).
protracted(E)   <-: event(E), dimension(E, 1).

%%
stative(E, V) <-:
  if(var(E),
     (setof(E0, stative0(E0, V), S), member(E, S)),
     once stative0(E, V)).
stative0(E, +) <-:
  stative_pred(Pred),
  bnf(Pred*E, Prop),
  known Prop.
stative0(E, -) <-:
  nonstative_pred(Pred),
```

```
  bnf(Pred*E, Prop),
  known Prop.

delimited(E, V) <-:
  if(var(E),
     (setof(E0, delimited0(E0, V), S), member(E, S)),
     once delimited0(E, V)).
delimited0(E, +) <-:
  delimited_pred(Pred),
  bnf(Pred*E, Prop),
  known Prop.
delimited0(E, +) <-: dimension(E, 0).
delimited0(E, -) <-: not delimited(E, +).
delimited0(E, V) <-:
  itr(Rel), bnf(Rel*X*E, Prop),
  known Prop,
  delimited(X, V).

dimension(E, V) <-:
  if(var(E),
     (setof(E0, dimension0(E0, V), S), member(E, S)),
     once dimension0(E, V)).
dimension0(E, 0) <-:
  dimension0_pred(Pred),
  bnf(Pred*E, Prop),
  known Prop.
dimension0(E, 1) <-:
  dimension1_pred(Pred),
  bnf(Pred*E, Prop),
  known Prop.
dimension0(E, 1) <-: delimited(E, -), eventuality(E).

%% conseq, antec state
conseq_st(EA, E) <-:
  known conseq_st(E1, E),
  known cul(EP, E1), known gr(EA, EP).
conseq_st(E1, E) <-:
  known conseq_st(EA, E),
  gr(EA, EP), cul(EP, E1).
conseq_st(comp(EP, intv(T0, T1)), E) <-:
  known conseq_st(E1, E),
  stop( EP, E1), time(E1, T1),
  start(EP, E0), time(E0, T0).
antec_st(EA, E) <-:
  known antec_st(E0, E),
  known start(EP, E0), known gr(EA, EP).
antec_st(E0, E) <-:
  known antec_st(EA, E),
  gr(EA, EP), start(EP, E0).
antec_st(comp(EP, intv(T0, T1)), E) <-:
  known antec_st(E0, E),
  start(EP, E0), time(E0, T0),
  stop( EP, E1), time(E1, T1).
stative0(E, +) <-: known conseq_st(_, E).
stative0(E, +) <-: known antec_st( _, E).
```

```
%% partof
partof(EA, EB) <-: gr(EA, E), comp(E, EB).

%% comp
comp(E, comp(E, I)) <-:
  (known comp(E, EA) ; max_comp(E, EA) ; moment_comp(E, _, EA)),
  time(EA, IA), subintv(IA, I).
time0(comp(_, I), I) <-: true.
max_comp(E, comp(E, intv(T0, T1))) <-:
  bd_l(E, E0), time(E0, T0),
  bd_r(E, E1), time(E1, T1).
moment_comp(E, E0, comp(E, intv(T0, T1))) <-:
  known moment(E, E0), time(E0, T0),
  end(         E, E1), time(E1, T1).
moment_comp(E, E1, comp(E, intv(T0, T1))) <-:
  known moment(E, E1), time(E1, T1),
  begin(       E, E0), time(E0, T0).
stative0(  comp(E, _), V) <-: not var(E),        stative(E, V).
stative0(  EA,         V) <-: known comp(E, EA), stative(E, V).
delimited0(comp(E, _), +) <-: not var(E).
delimited0(EA,         +) <-: known comp(_, EA).
dimension0(comp(E, _), 1) <-: not var(E),        eventuality(E).
dimension0(EA,         1) <-: known comp(E, EA), eventuality(E).

%% moment
moment(E, moment(E, T)) <-:
  comp(E, EA), time(EA, TA),
  betw(TA, T).
time0(moment(_, T), T) <-: true.
stative0(  moment(E, _), +) <-: not var(E).
stative0(  ES,           +) <-: known moment(_, ES).
dimension0(moment(E, _), 0) <-: not var(E).
dimension0(ES,           0) <-: known moment(_, ES).


%% inprog
inprog(E, inprog(E)) <-: not known inprog(E, _), process(E).
stative0(  inprog(E), +) <-: not var(E).
stative0(  ES,        +) <-: known inprog(_, ES).
delimited0(inprog(E), -) <-: not var(E).
delimited0(ES,        -) <-: known inprog(_, ES).

%% gr
gr(E, gr(E)) <-: not known gr(E, _), protracted(E).
max_comp(gr(E), E) <-: true.
stative0(  gr(E), -) <-: not var(E).
stative0(  EP,    -) <-: known gr(_, EP).
delimited0(gr(E), -) <-: not var(E).
delimited0(EP,    -) <-: known gr(_, EP).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                        incremental thematic relations
*/
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
itr_schema(Prop) <-:
  itr(Rel), bnf(Rel*X*E, Prop),
  (ev_proc(Rel, X, E) ;
   proc_ev(Rel, X, E) ;
   ev_ev(Rel, X, E)).

ev_proc(Rel, X, E) <-:
  bnf(Rel*XA*EA, Prop),
  known Prop,
  gr(XA, X), gr(EA, E).
proc_ev(Rel, XA, EA) <-:
  bnf(Rel*X*E, Prop),
  known Prop,
  comp(  X, XA), comp( E, EA),
  quant(XA, QA), dur( EA, DA),
  rate(E, R),
  QA = DA * R.
ev_ev(Rel, XB, EB) <-:
  bnf(Rel*XA*EA, Prop),
  known Prop,
  partof(XA, XB), partof(EA, EB),
  quant( XA, QA), dur(EA, DA),
  quant( XB, QB), dur(EB, DB),
  QA =\= QB, DA =\= DB,
  QA / DA = QB / DB.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                        begin, end, start, stop, cul
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
bd_l(E, E0) <-: start(E, E0) ; begin(E, E0).
bd_r(E, E1) <-: stop( E, E1) ; end(  E, E1).

begin(E, E0) <-: conseq_st(E0, E), dimension(E0, 0).
end(  E, E1) <-: antec_st( E1, E), dimension(E1, 0).
begin(inprog(E), E0) <-: start(E, E0).
end(  inprog(E), E1) <-: stop( E, E1).
start(E, start(E, T)) <-: gr(EA, E), time(EA, intv(T, _)).
cul(  E, cul(E, T)  ) <-: gr(EA, E), time(EA, intv(_, T)).
time0(start(_, T), T) <-: true.
time0(cul(  _, T), T) <-: true.
stop(E, E1) <-: cul(E, E1).

stative0(start(E, _), -) <-: not var(E).
stative0(cul(  E, _), -) <-: not var(E).
stative0(ET,          -) <-: known start(_, ET).
stative0(ET,          -) <-: known stop( _, ET).
stative0(ET,          -) <-: known cul(  _, ET).
dimension0(start(E, _), 0) <-: not var(E).
dimension0(cul(  E, _), 0) <-: not var(E).
dimension0(ET,          0) <-: known start(_, ET).
dimension0(ET,          0) <-: known stop( _, ET).
dimension0(ET,          0) <-: known cul(  _, ET).
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                aspectual verbs, inprog plus optional coercion
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
start_plus(E, E0) <-: process(E), start(E, E0).
start_plus(E, E0) <-: protracted(E), gr(E, EP), start(EP, E0).
stop_plus(E, E1) <-: process(E), stop(E, E1).
stop_plus(E, E1) <-: protracted(E), gr(E, EP), stop(EP, E1).
finish_plus(E, E1) <-: process(E), cul(E, E1).
finish_plus(E, E1) <-: protracted(E), gr(E, EP), cul(EP, E1).

inprog_plus(E, ES) <-: process(E), inprog_m(E, ES).
inprog_plus(E, ES) <-: protracted(E), gr(E, EP), inprog_m(EP, ES).
inprog_m(E, ES) <-: inprog(E, ES).
inprog_m(E, ES) <-: inprog(E, ES0), moment(ES0, ES).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                              actual eventualities
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
actual(E) <-:
  not known actual(E),
  once actual0(E).
actual0(E) <-:
  known gr(E, EP), known cul(EP, E1),
  actual(E1).
actual0(start(gr(E), _)) <-:
  known actual(E).
actual0(cul(gr(E), _)) <-:
  known actual(E).
actual0(moment(E, T)) <-:
  comp(E, EA), time(EA, TA),
  betw(TA, T),
  actual(EA).
actual0(comp(E, intv(_, T1))) <-:
  bd_r(E, E1), time(E1, T1),
  actual(E1).
actual0(comp(E, IB)) <-:
  comp(E, EA), time(EA, IA), not eq(IA, IB), subintv(IA, IB),
  actual(EA).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                               temporal adverbials
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
at(T, E) <-:
  dimension(E, 0),
  time(E, T).
from_to(T0, T1, E, EA) <-:
  comp(E, EA),
  time(EA, intv(T0, T1)).
```

```
for(D, E, EA) <-:
  comp(E, EA),
  dur(EA, D).
in(D, E) <-:
  protracted(E),
  dur(E, D0),
  D0 =< D.
dur(E, D) <-:
  time(E, intv(T0, T1)),
  D = T1 - T0.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                                    times
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
time(E, T) <-: not known time(E, T), once time0(E, T).
time0(E, intv(T0, T1)) <-:
  known gr(E, EP),
  known start(EP, E0), time(E0, T0),
  known cul(  EP, E1), time(E1, T1).
subintv(intv(T0, T1), intv(T0, T1)) <-: true.
subintv(intv(T0, T1A), intv(T0, T1B)) <-:
  T0 =< T1B, T1B =< T1A.
subintv(intv(T0A, T1A), intv(T0B, T1B)) <-:
  T0A =< T0B, T0B =< T1A,
  T0A =< T1B, T1B =< T1A.
betw(intv(T0, T1), T) <-:
  T0 =< T, T =< T1.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                      level of a substance in a container
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
full_of(     C, X, E) <-: level(C, X, 1.0, E), not known full_of(     C, X, E).
half_full_of(C, X, E) <-: level(C, X, 0.5, E), not known half_full_of(C, X, E).
empty_of(    C, X, E) <-: level(C, X, 0.0, E), not known empty_of(    C, X, E).
full(     C, E) <-: full_of(     C, _, E).
half_full(C, E) <-: half_full_of(C, _, E).
empty(    C, E) <-: empty_of(    C, _, E).

level(C, X, N, E) <-:
  level0(C, X, N, E).
level(C, X, N, E0) <-:
  canonical(E),
  known level(C, X, N, E),
  moment(E, E0).
level(C, X, N, E) <-:
  process(EP),
  once level1a(C, X, N, E, EP).
level(C, X, N, E) <-:
  protracted(EA),
  once level1b(C, X, N, E, EA).
```

```
level0(C, X, 1.0, E) <-: known full_of(     C, X, E).
level0(C, X, 0.5, E) <-: known half_full_of(C, X, E).
level0(C, X, 0.0, E) <-: known empty_of(    C, X, E).

level1a(C, X, N, E, EP) <-:
  once ((add_to(_, X, C, EP), Pol = 1) ;
        (remove_from(_, X, C, EP), Pol = -1)),
  moment(EP, E), time(E, T),
  rate(EP, R), vol(C, V),
  start(EP, E0), time(E0, T0),
  antec_st(E0, ES),
  known level(C, X, N0, ES),
  Q = R * (T - T0),
  N = N0 + Pol * Q / V.
level1b(C, X, N, E, EA) <-:
  once (add_to(_, XA, C, EA) ; remove_from(_, XA, C, EA)), comp(X, XA),
  gr(EA, EP), moment(EP, E), time(E, T),
  time(EA, intv(T0, T1)),
  antec_st(EA, ES0),
  known level(C, X, N0, ES0),
  conseq_st(EA, ES1),
  known level(C, X, N1, ES1),
  N = N0 + (N1 - N0) * (T - T0) / (T1 - T0).

level_aug <-:
  setof(E, canonical(E), S),
  map(E, level_aug(E), S).
level_aug(E) <-:
  once level2(C, X, N, E),
  assume level(C, X, N, E).

level2(C, X, N, E) <-:
  known moment(E, E0),
  level0(C, X, N, E0).
level2(C, X, 1.0, E) <-:
  conseq_st(EA, E),
  fill_with(_, C, X, EA).
level2(C, X, 0.0, E) <-:
  conseq_st(EA, E),
  empty_of(_, C, X, EA).
level2(C, X, N, E) <-:
  conseq_st(EA, E),
  once ((add_to(_, XA, C, EA), Pol = 1) ;
        (remove_from(_, XA, C, EA), Pol = -1)),
  comp(X, XA),
  quant(XA, Q), vol(C, V),
  antec_st(EA, E0),
  once level2(C, X, N0, E0),
  N = N0 + Pol * Q / V.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                            lexical predicate types
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
nonstative_pred(Pred) <-: protracted_pred(Pred).
delimited_pred( Pred) <-: protracted_pred(Pred).
dimension1_pred(Pred) <-: protracted_pred(Pred).
nonstative_pred(Pred) <-: incremental_pred(Pred).
dimension1_pred(Pred) <-: incremental_pred(Pred).
incremental_pred(Pred) <-: itr(Rel), bnf(Rel*_, Pred).

stative_pred(E^ full_of(     _, _, E)) <-: true.
stative_pred(E^ half_full_of(_, _, E)) <-: true.
stative_pred(E^ empty_of(    _, _, E)) <-: true.
protracted_pred(E^ fill_with(_, _, _, E)) <-: true.
protracted_pred(E^ empty_of( _, _, _, E)) <-: true.
itr(X^E^ pour_into(_, X, _, E))     <-: true.
itr(X^E^ pour_out_of(_, X, _, E))   <-: true.
itr(X^E^ siphon_into(_, X, _, E))   <-: true.
itr(X^E^ siphon_out_of(_, X, _, E)) <-: true.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                              pouring and siphoning
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
pour_into(A, X, C, E)     <-: itr_schema(pour_into(A, X, C, E)).
pour_out_of(A, X, C, E)   <-: itr_schema(pour_out_of(A, X, C, E)).
siphon_into(A, X, C, E)   <-: itr_schema(siphon_into(A, X, C, E)).
siphon_out_of(A, X, C, E) <-: itr_schema(siphon_out_of(A, X, C, E)).

rate(E, R) <-:
  rate0(E, X, R0),
  viscosity(X, V),
  R = R0 * V.
rate0(E, X, 2.0) <-: known pour_into(    _, X, _, E).
rate0(E, X, 2.0) <-: known pour_out_of(  _, X, _, E).
rate0(E, X, 0.4) <-: known siphon_into(  _, X, _, E).
rate0(E, X, 0.4) <-: known siphon_out_of(_, X, _, E).

add_to(A, X, C, E) <-: pour_into(  A, X, C, E).
add_to(A, X, C, E) <-: siphon_into(A, X, C, E).
remove_from(A, X, C, E) <-: pour_out_of(  A, X, C, E).
remove_from(A, X, C, E) <-: siphon_out_of(A, X, C, E).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                              filling and emptying
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
fill(A, C, E) <-: fill_with(A, C, _, E).
empty(A, C, E) <-: empty_of(A, C, _, E).

add_to(A, XA, C, E) <-:
  known fill_with(A, C, X, E), comp(X, XA),
  antec_st(E, E0), known level(C, X, N0, E0),
  quant(XA, QA), vol(C, V),
  QA = V * (1.0 - N0).
remove_from(A, XA, C, E) <-:
```

```
  known empty_of(A, C, X, E), comp(X, XA),
  antec_st(E, E0), known level(C, X, N0, E0),
  quant(XA, QA), vol(C, V),
  QA = V * N0.

%% nb. actual restriction!
fill_with(A, C, X, E) <-:
  add_to(A, XA, C, E), comp(X, XA),
  conseq_st(E, ES), known level(C, X, 1.0, ES),
  actual(E).
empty_of(A, C, X, E) <-:
  remove_from(A, XA, C, E), comp(X, XA),
  conseq_st(E, ES), known level(C, X, 0.0, ES),
  actual(E).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                        lexically-named material entities
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
material0(X) <-: known wort(X).
material0(X) <-: known water(X).

wort(XA)  <-: material(X),  comp(X, XA), wort(X).
water(XA) <-: material(X),  comp(X, XA), water(X).
wort(X)   <-: material(XA), gr(XA, X),   wort(XA).
water(X)  <-: material(XA), gr(XA, X),   water(XA).
viscosity(X, 0.5) <-: wort(X).
viscosity(X, 1.0) <-: water(X).

material0(X)  <-: carboy(X).
delimited0(X) <-: carboy(X).

large(X) <-: carboy(X), vol(X, V), V >= 26.
small(X) <-: carboy(X), vol(X, V), V =< 22.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                                material entities
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
substance(X) <-: material(X), delimited(X, -).
object(X)    <-: material(X), delimited(X, +).
material(X) <-:
  if(var(X),
     (setof(X0, material0(X0), S), member(X, S)),
     once material0(X)).

%% comp
comp(X, comp(X, _, _)) <-: substance(X).
quant(comp(_, Q, _), Q) <-: true.
material0(comp(X, _, _)) <-: not var(X),        material(X).
material0(XA           ) <-: known comp(X, XA), material(X).
delimited0(comp(X, _, _), +) <-: not var(X).
```

```
%% gr
gr(X, gr(X)) <-: not known gr(X, _), object(X).
material0(gr(XA)) <-: not var(XA),     material(XA).
material0(X     ) <-: known gr(XA, X), material(XA).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                            keepers and assumables
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
assumable(_) :- fail.
keeper(_) :- fail.
```

Appendix B

Microworld Code

The code for the microworld consists of a set of modules centered around the constraint-based abductive interpretation meta-interpreter, which appears separately in appendix C. Those that concern the Quintus Prowindows interface have been omitted, as have the parser and lexicon, leaving just the clauses involved in using the logical form output by the parser to compute a collection of eventualities which fully determines the animation.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                                    scenes
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
scene(Snapshots) <-:
  setof(LM, landmark(LM), LMs),
  map(LM, P, snapshot_anytime(LM, P), LMs, Ps),
  map(P, R, eq([(P, 0.0)], R), Ps, Snapshots0),
  snapshots(Snapshots1),
  append(Snapshots0, Snapshots1, Snapshots2),
  delete(Snapshots2, [], Snapshots).
snapshots(Ss) <-:
  setof(X, object(X), Xs),
  map(X, STs, snapshots_times(X, STs), Xs, Ss).
snapshots_times(X, STs) <-:
  print('tracing '), print_nl(X),
  setof(T, snapshot_time(X, T), Ts),
  map(T, S, snapshot_at_time(X, T, S), Ts, Ss),
  nums(Ts, Ns),
  pair(Ss, Ns, STs).
snapshot_time(X, T) <-:
  snapshot_evty(E), actual(E),
  object(E, X),
  evty_time(E, T).

evty_time(E, T) <-: time(E, T).
evty_time(E, T) <-: interval(E, I), interval_time(I, T).
```

```
evty_time(E, T) <-: middle_point_time(E, T).

interval_time(intv(T, _), T) <-: true.
interval_time(intv(_, T), T) <-: true.

middle_point_time(E, T) <-:
  path(E, Path),
  middle_points(Path, Ps),
  start_point(Path, P0),
  speed(E, Speed),
  interval(E, intv(T0, _)),
  middle_points_times([P0 | Ps], T0, Speed, [T0 | Ts]),
  member(T, Ts).
middle_point_time(E0, T) <-:
  start(E, E0),  stop(E, E1),
  start_stop_event_cs(E0, E1, E01),
  comp(E, E01),
  interval(E01, intv(T0,  _)),
  path(E01, Path),
  start_point(Path, P0),
  middle_points(Path, Ps),
  speed(E01, Speed),
  middle_points_times([P0 | Ps], T0, Speed, [T0 | Ts]),
  member(T, Ts).
middle_points_times([_], Tn, _, [Tn]) <-: true.
middle_points_times([Pi, Pj | RestPs], Ti, Speed, [Ti | RestTs]) <-:
  dist_from_to(Pi, Pj, Dist),
  dist_speed_dur(Dist, Speed, Dur), intv_dur(intv(Ti, Tj), Dur),
  middle_points_times([Pj | RestPs], Tj, Speed, RestTs).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                              parameter calculation
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
parameter_at_time(E, MatchP, Time, Parameter) <-:
  once freeze parameter_at_time0(E, MatchP, Time, Parameter).

%% last mentioned evty
parameter_at_time0(E, MatchP, secs(T), Parameter) <-:
  once (local_entity(E), call(MatchP), actual(E)),
  (time(E, secs(TE)) ; interval(E, intv(_, secs(TE)))),
  T >= TE,
  evty_parameter_at_time(E, secs(T), Parameter).

%% during or at evty
parameter_at_time0(E, MatchP, Time, Parameter) <-:
  freeze call(MatchP), actual(E),
  dist_to_time(E, Time, secs(0.0)),
  evty_parameter_at_time(E, Time, Parameter).

%% closest evty
parameter_at_time0(E, MatchP, Time, Parameter) <-:
  findall(E, (call(MatchP), actual(E)), [Best | Rest]),
  closest_evty_to_time(Best, Rest, Time, E),
```

```
  evty_parameter_at_time(E, Time, Parameter).

closest_evty_to_time(Best, [], _, Best) <-: true.
closest_evty_to_time(Best, [First | Rest], Time, E) <-:
  closest_to_time(Best, First, Time, Next),
  closest_evty_to_time(Next, Rest, Time, E).
closest_to_time(E1, E2, T, E) <-:
  dist_to_time(E1, T, secs(D1)),
  dist_to_time(E2, T, secs(D2)),
  if(((0.0 =< D2, D2 < D1) ; (D1 = D2, structure(start, E2))),
     eq(E, E2),
     eq(E, E1)).

%% distance to evty time
dist_to_time(E, secs(T), secs(D)) <-:
  time(E, secs(T0)),
  D = T - T0.
dist_to_time(E, secs(T), secs(D)) <-:
  interval(E, intv(secs(T0), secs(T1))),
  if((T0 =< T, T =< T1),
     D = 0.0,
     D = T - T1).

%% state
evty_parameter_at_time(E, _, P) <-: freeze state(E), point(E, P).

%% start, stop
evty_parameter_at_time(E, secs(T), P) <-:
  structure(start, E),
  time(E, secs(TE)), T = TE,
  point(E, P).
evty_parameter_at_time(E, secs(T), P) <-:
  structure(start, E),
  time(E, secs(TE)), T > TE,
  point(E, PE), process(E, E0),
  start_point(PathE1, PE), end_point(PathE1, P),
  path(E1, PathE1), interval(E1, intv(secs(TE), secs(T))),
  comp(E0, E1).
evty_parameter_at_time(E, secs(T), P) <-:
  structure(stop, E),
  time(E, secs(TE)), T >= TE,
  point(E, P).

%% event
evty_parameter_at_time(E, secs(T), P) <-:
  interval(E, intv(secs(T0), _)),
  T =< T0,
  path(E, Path),
  start_point(Path, P).
evty_parameter_at_time(E, secs(T), P) <-:
  interval(E, intv(_, secs(T1))),
  T >= T1,
  path(E, Path),
  end_point(Path, P).
evty_parameter_at_time(E, secs(T), P) <-:
```

```
  interval(E, intv(secs(T0), secs(T1))),
  T0 < T, T < T1,
  path(E, Path), start_point(Path, P0),
  start_point(Path1, P0), end_point(Path1, P),
  path(E1, Path1), interval(E1, intv(secs(T0), secs(T))),
  subevent(E, E1).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                    declaratives, later, when, until
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
decl(E, PE) <-:
  call(PE),
  (time(E, T) ;
   interval(E, intv(_, T)) ;
   (start(E, E0) $ 0.001, time(E0, T))),
  current_rt(T) $ 0.001.
later(msecs(D)) <-:
  current_rt(secs(T0)),
  T1 = T0 + D / 1000,
  current_rt(secs(T1)) $ 0.001.
when(E1, PE1, _E2, PE2) <-:
  call(PE1),
  time(E1, T1),
  current_rt(T1) $ 0.001,
  call(PE2).
until(E, PE, E2, PE2, E01) <-:
  call(PE),
  start(E, E0) $ 0.001,
  call(PE2),
  time(E2, T2),
  current_rt(T2) $ 0.001,
  stop(E, E1) $ 0.001,
  start_stop_event_cs(E0, E1, E01),
  comp(E, E01).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                                for/in duration
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for(E, msecs(D), E01) <-:
  DS = D / 1000,
  start(E, E0) $ 0.001,
  time(E0, secs(T0)), T1 = T0 + DS,
  current_rt(secs(T1)) $ 0.001,
  stop(E, E1) $ 0.001,
  start_stop_event_cs(E0, E1, E01),
  comp(E, E01).
in(msecs(D), E) <-:
  DS =< D / 1000,
  duration(E, secs(DS)).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
/*
                                locomotion
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
glide(Agent, P, E) <-:
  gliding_speed(Agent, Speed),
  speed(E, Speed),
  translation(Agent, P, E) $ 0.001.
zip(Agent, P, E) <-:
  zipping_speed(Agent, Speed),
  speed(E, Speed),
  translation(Agent, P, E) $ 0.001.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                          aiming, firing, reaching, etc.
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
aim(X, R, E) <-: aim(X, R, lead, E) ; aim(X, R, direct, E).
aim(X, R, W, E) <-:
  current_rt(RT),
  location_at_time(X, RT, PX),
  location_at_time(R, RT, PR0),
  orientation_at_time(X, RT, AX0),
  if(eq(W, direct),
     from_to_dir_dist(PX, PR0, A0, _),
     (structure(shell, S), translation_speed(S, TS),
      dist_from_to(PX, PR0, DL),
      dist_speed_dur(DL, TS, DurL),
      intv_dur(intv(RT, TL), DurL),
      location_at_time(R, TL, PRL),
      from_to_dir_dist(PX, PRL, A0, _))),
  from_to_dir_dist(AX0, A0, Dir, rads(D0)),
  rotation_speed(X, RS),
  print_nl((W, A0, D0)),
  minimize([AX0, A0, RT], D0, D,
           [AX,  A,  T],
           aim_dist(AX, A, rads(D)),
           [AX1, A1, T1],
           aim_update(D, AX, T, W, PX, R, RS, Dir, AX1, A1, T1),
           0.01,
           [AXm, _, _], _),
  end_point(Path, AXm),
  rotation(X, Path, E) $ 0.001.
aim_dist(AX, A, D) <-: from_to_dir_dist(AX, A, _, D).
aim_update(D, AX, T, W, PX, R, RS, Dir, AX1, A1, T1) <-:
  D1 = D / 2,
  dist_speed_dur(rads(D1), RS, Dur),
  intv_dur(intv(T, T1), Dur),
  from_to_dir_dist(AX, AX1, Dir, rads(D1)),
  location_at_time(R, T1, PR1),
  if(eq(W, direct),
     from_to_dir_dist(PX, PR1, A1, _),
     (structure(shell, S), translation_speed(S, TS),
      dist_from_to(PX, PR1, DL),
```

```
      dist_speed_dur(DL, TS, DurL),
      intv_dur(intv(T, TL), DurL),
      location_at_time(R, TL, PRL),
      from_to_dir_dist(PX, PRL, A1, _))).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                        locations, orientations, etc.
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
state(E)         <-: location(_, E).
snapshot_evty(E) <-: location(_, E).

structure(loc, loc(_, _, _, _)) <-: true.
object(loc(X, _, _, _), X) <-: true.
point( loc(_, P, _, _), P) <-: true.
time(  loc(_, _, T, _), T) <-: true.
id(    loc(_, _, _, I), I) <-: true.

location(X, E) <-:
  new(loc, E),
  object(E, X),
  no_simult_evty(E0, location(X, E0), E).
location_at_time(X, T, P) <-:
  parameter_at_time(E, location_evty(X, E), T, P).
location_evty(X, E) <-: location(X, E).
location_evty(X, E) <-: translation(X, E), delimited(E, +delim).
location_evty(X, E) <-:
  aspectual_event(E), object(E, X),
  process(E, E0), translation(X, E0).

state(E)         <-: orientation(_, E).
snapshot_evty(E) <-: orientation(_, E).

structure(ort, ort(_, _, _, _)) <-: true.
object(ort(X, _, _, _), X) <-: true.
point( ort(_, P, _, _), P) <-: true.
time(  ort(_, _, T, _), T) <-: true.
id(    ort(_, _, _, I), I) <-: true.

orientation(X, E) <-:
  new(ort, E),
  object(E, X),
  no_simult_evty(E0, orientation(X, E0), E).
orientation_at_time(X, T, P) <-:
  parameter_at_time(E, orientation_evty(X, E), T, P).
orientation_evty(X, E) <-: orientation(X, E).
orientation_evty(X, E) <-: rotation(X, E), delimited(E, +delim).
orientation_evty(X, E) <-:
  aspectual_event(E), object(E, X),
  process(E, E0), rotation(X, E0).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
```

```
                              start, etc.
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
aspectual_event(E) <-: start(_, E).
snapshot_evty(E)   <-: start(_, E).

structure(start, start(_, _, _, _)) <-: true.
object( start(E, _, _, _), X) <-: object(E, X).
process(start(E, _, _, _), E) <-: delimited(E, -delim).
point(  start(_, P, _, _), P) <-: true.
time(   start(_, _, T, _), T) <-: true.
id(     start(_, _, _, I), I) <-: true.

start(E0, E) <-:
  new(start, E),
  process(E, E0),
  current_rt(RT),
  parameter_match_p(E0, E1, MatchP),
  parameter_at_time(E1, MatchP, RT, P),
  point(E, P), time(E, RT),
  init_rotn_c(E0).
start(E0, E) <-:
  new(start, E),
  process(E, E0),
  current_rt(RT),
  time(E, RT),
  causes(E0, E1),
  start(E1, _).
start(_X, E0, E) <-:
  (eq(E0, E1) ;
   gr(E0, E1)),
  start(E1, E) $ 0.001.
start_stop_event_cs(E0, E1, E01) <-:
  time(E0, T0), time(E1, T1),
  interval(E01, intv(T0, T1)),
  point(E0, P0), point(E1, P1),
  start_point(Path, P0), end_point(Path, P1),
  path(E01, Path).
start_stop_event_cs(E0, E1, E01) <-:
  time(E0, T0), time(E1, T1),
  interval(E01, intv(T0, T1)),
  process(E0, E),
  causes(E, EC),
  start(EC, EC0), stop(EC, EC1),
  causes(E01, EC01),
  start_stop_event_cs(EC0, EC1, EC01).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                          dynamic eventuality basics
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
snapshot_evty(E) <-: dynamic_evty(E), delimited(E, +delim).
dynamic_evty_cs(E) <-:
  delimited(E, -delim).
```

```
dynamic_evty_cs(E) <-:
  interval(E, Intv), duration(E, Dur),
  intv_dur(Intv, Dur),
  speed(E, Speed),
  ((path(E, Path), distance(Path, Dist),
    dist_speed_dur(Dist, Speed, Dur)) ;
   (substance(E, Subst), amount(Subst, Amt),
    amt_speed_dur(Amt, Speed, Dur))).
init_cs(E) <-:
  delimited(E, -delim).
init_cs(E) <-:
  current_rt(RT),
  interval(E, intv(RT, _)),
  parameter_match_p(E, E1, MatchP),
  parameter_at_time(E1, MatchP, RT, P0),
  path(E, Path), start_point(Path, P0),
  init_rotn_c(E).
init_rotn_c(E) <-:
  if((structure(transl, E), object(E, X), directed(X)),
     (path(E, Path), direction(Path, A1), end_point(Path1, A1),
      rotation(X, Path1, _)),
     true).

%%
comp(E0, E) <-:
  structure(Struct, E0), structure(Struct, E),
  delimited(E0, -delim), delimited(E, +delim),
  object(E0, X),         object(E, X),
  speed(E0, S),          speed(E, S),
  path(E0, P0),          path(E, P),
  dynamic_evty_cs(E0),   dynamic_evty_cs(E),
  comp(P0, P).

%%
gr(E0, E) <-:
  structure(Struct, E0), structure(Struct, E),
  delimited(E0, +delim), delimited(E, -delim),
  object(E0, X),         object(E, X),
  speed(E0, S),          speed(E, S),
  path(E0, P0),          path(E, P),
  dynamic_evty_cs(E0),   dynamic_evty_cs(E),
  gr(P0, P),
  platonic(E0) $ 0.001,
  dynamic_evty(E) $ 0.001.

%%
subevent(EA, EB) <-:
  structure(Struct, EA), structure(Struct, EB),
  delimited(EA, +delim), delimited(EB, +delim),
  dynamic_evty_cs(EA),   dynamic_evty_cs(EB),
  object(EA, X),         object(EB, X),
  speed(EA, S),          speed(EB, S),
  path(EA, PA),          path(EB, PB),
  interval(EA, IA),      interval(EB, IB),
  subinterval(IA, IB),
```

```
  subpath(PA, PB).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                              translations, etc.
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
dynamic_evty(E) <-: translation(_, E).

structure(transl, transl(_, _, _, _, _, _, _)) <-: true.
delimited(transl(D, _, _, _, _, _, _), D) <-: true.
object(   transl(_, O, _, _, _, _, _), O) <-: true.
path(     transl(D, _, P, _, _, _, _), P) <-: delimited(P, D).
speed(    transl(_, _, _, S, _, _, _), S) <-: true.
interval( transl(D, _, _, _, I, _, _), I) <-: eq(D, +delim).
duration( transl(D, _, _, _, _, R, _), R) <-: eq(D, +delim).
id(       transl(_, _, _, _, _, _, I), I) <-: true.

translation(X, E) <-: translation(X, _, E).
translation(X, P, E) <-:
  new(transl, E),
  object(E, X), path(E, P),
  translation_speed(X, S), speed(E, S),
  init_cs(E), dynamic_evty_cs(E), path_cs(P).
parameter_match_p(E, E0, location_evty(X, E0)) <-:
  structure(transl, E),
  object(E, X).
translation_in_progress(X, E0) <-:
  once freeze (local_entity(E), location_evty(X, E), actual(E)),
  if(structure(start, E),
     process(E, E0),
     fail).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                                    paths
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
structure(path, path(_, _, _, _, _, _, _, _)) <-: true.
delimited(    path(D, _, _, _, _, _, _, _), D) <-: true.
min_point(    path(D, P, _, _, _, _, _, _), P) <-: eq(D, -delim).
start_point(  path(D, _, P, _, _, _, _, _), P) <-: eq(D, +delim).
middle_points(path(_, _, _, M, _, _, _, _), M) <-: true.
end_point(    path(D, _, _, _, P, _, _, _), P) <-: eq(D, +delim).
max_point(    path(D, _, _, _, _, P, _, _), P) <-: eq(D, -delim).
direction(    path(_, _, _, _, _, _, R, _), R) <-: true.
distance(     path(D, _, _, _, _, _, _, S), S) <-: eq(D, +delim).

path_cs(Path) <-:
  min_point(Path, P0), max_point(Path, P1),
  direction(Path, Dir),
  if((inst_point(P0), inst_point(P1)),
     from_to_dir_dist(P0, P1, Dir, _),
     true).
path_cs(Path) <-:
```

```
  start_point(Path, P0), end_point(Path, P1),
  middle_points(Path, Ps),
  direction(Path, Dir),
  distance(Path, Dist),
  if(eq(Ps, []),
     from_to_dir_dist(P0, P1, Dir, Dist),
     (from_to_dir_dist(P0, P1, Dir, _),
      distance_sum([P0 | Ps], P1, Dist))).
distance_sum([Pn], P1, Dist) <-:
  dist_from_to(Pn, P1, Dist).
distance_sum([Pi, Pj | RestPs], P1, Dist) <-:
  dist_from_to(Pi, Pj, Dist0),
  distance_sum([Pj | RestPs], P1, Dist1),
  num(Dist0, D0), num(Dist1, D1), num(Dist, D),
  functor(Dist0, F, _), functor(Dist, F, _),
  D = D0 + D1.

%%
comp(Path0, Path) <-:
  min_point(Path0, P0), max_point(Path0, P1),
  inst_point(P0), not inst_point(P1),
  direction(Path0, Dir), direction(Path, Dir),
  start_point(Path, P),
  (eq(P0, P) ; from_to_dir_dist(P0, P, Dir, _)),
  path_cs(Path).
comp(Path0, Path) <-:
  min_point(Path0, P0), max_point(Path0, P1),
  not inst_point(P0), inst_point(P1),
  direction(Path0, Dir), direction(Path, Dir),
  end_point(Path, P),
  (eq(P1, P) ; from_to_dir_dist(P, P1, Dir, _)),
  path_cs(Path).
comp(Path0, Path) <-:
  min_point(Path0, P0), max_point(Path0, P1),
  inst_point(P0), inst_point(P1),
  start_point(Path1, P0), end_point(Path1, P1),
  middle_points(Path0, Ps), middle_points(Path1, Ps),
  subpath(Path1, Path).

%%
gr(Path0, Path) <-:
  start_point(Path0, P0), end_point(Path0, P1),
  inst_point(P0), inst_point(P1),
  min_point(Path, P0), max_point(Path, P1),
  middle_points(Path0, Ps), middle_points(Path, Ps),
  direction(Path0, Dir), direction(Path, Dir).

%%
subpath(PathA, PathB) <-:
  path_cs(PathA),
  start_point(PathA, P0A),    start_point(PathB, P0B),
  end_point(PathA, P1A),      end_point(PathB, P1B),
  middle_points(PathA, PsA),  middle_points(PathB, PsB),
  distance(PathA, DistA),     distance(PathB, DistB),
  subsegment(P0A, PsA, P1A, DistA, P0B, PsB, P1B, DistB),
```

```
  path_cs(PathB).
subsegment(P0A, PsA, P1A, DistA, P0B, PsB, P1B, DistB) <-:
  num(DistA, DA), num(DistB, DB),
  0.0 =< T0, T0 =< 1.0,
  0.0 =< T1, T1 =< 1.0,
  T0 < T1,
  D0B = T0 * DA, D1B = T1 * DA,
  T1 - T0 = (DB / DA),
  append([[P0A], PsA, [P1A]], Ps01A),
  subsegment_dist(Ps01A, D0B, D1B, P0B, PsB, P1B).
subsegment_dist([PI, PJ | Rest], D0, D1, P0, Ps, P1) <-:
  dist_from_to(PI, PJ, DistIJ), num(DistIJ, DIJ),
  once ((0.0 =< D0, D0 =< DIJ, interpolate_dist(PI, PJ, D0, P0)) ;
        (inst_point(P0), number(D0)) ;
        D0 > DIJ),
  once ((inst_point(P0), number(D0), D1 =< DIJ,
         interpolate_dist(PI, PJ, D1, P1), eq(PsK, [])) ;
        (D1 > DIJ,
         D0K = D0 - DIJ, D1K = D1 - DIJ,
         subsegment_dist([PJ | Rest], D0K, D1K, P0, PsK, P1))),
  if(D0 < 0.0,
     eq(Ps, [PI | PsK]),
     eq(Ps, PsK)).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                      distance, away from, towards, etc.
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
dist(meters(D), Path) <-:
  distance(Path, pts(D)).
away_from(X, R, Path) <-:
  direction(Path, A),
  current_rt(RT),
  location_at_time(X, RT, PX),
  location_at_time(R, RT, PR),
  from_to_dir_dist(PR, PX, A, _),
  (min_point(Path, PR) ;
   start_point(Path, PX)).
towards(X, R, Path) <-:
  direction(Path, A),
  current_rt(RT),
  location_at_time(X, RT, PX),
  location_at_time(R, RT, PR),
  from_to_dir_dist(PX, PR, A, _),
  (max_point(Path, PR) ;
   start_point(Path, PX)).
over_to(X, R, Path) <-:
  start_point(Path, PX), end_point(Path, P1),
  current_rt(RT),
  location_at_time(X, RT, PX),
  location_at_time(R, RT, PR),
  radius(X, RX),
  radius(R, RR),
  from_to_dir_dist(PX, PR, A, _),
```

```
  proximal_dir_dist(P1, RX, PR, RR, A, RX).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                                dante, etc.
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
snapshot(dante, dante(_, _, _, _)) <-: true.
snap_radius(     dante(R, _, _, _), R) <-: true.
snap_existence(  dante(_, E, _, _), E) <-: true.
snap_orientation(dante(_, _, O, _), O) <-: true.
snap_location(   dante(_, _, _, L), L) <-: true.

object(dante)   <-: true.
dante(dante)    <-: true.
radius(dante, pts(15.0)) <-: true.
translation_speed(dante, Speed) <-:
  gliding_speed(dante, Speed) ;
  zipping_speed(dante, Speed) ;
  jetting_speed(dante, Speed).
gliding_speed(dante, pts_sec(40.0)) <-: true.
zipping_speed(dante, pts_sec(60.0)) <-: true.
snapshot_at_time(dante, T, Dante) <-:
  snapshot(dante, Dante),
  radius(dante, R),
  snap_radius(Dante, R),
  existence_at_time(dante, T, E),
  snap_existence(Dante, E),
  orientation_at_time(dante, T, O),
  snap_orientation(Dante, O),
  location_at_time(dante, T, L),
  snap_location(Dante, L).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                       spatial, temporal etc. constraints
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% from_to_dir_dist calculates any 1 from the other 3
from_to_dir_dist(P0, P1, Dir, Dist) <-:
  dist_from_to(P0, P1, Dist),
  to_from_dir_dist(P0, Dir, Dist, P1),
  dir_from_to_dist(P0, P1, Dist, Dir).

%% locations
dist_from_to(pt(X0,Y0), pt(X1,Y1), pts(N)) <-:
  N = sqrt(sqr(X1 - X0) + sqr(Y1 - Y0)).
to_from_dir_dist(pt(X0, Y0), nrads(R), pts(N), pt(X1, Y1)) <-:
  X1 = cos(R) * N + X0,
  Y1 = sin(R) * N + Y0.
dir_from_to_dist(pt(X0,Y0), pt(X1,Y1), pts(N), nrads(R)) <-:
  X = X1 - X0, Y = Y1 - Y0,
  R = acos(X / N) * pol(Y).

%% proximal distance
```

```
proximal_diff(PX, RX, PR, RR, pts(D), pts(PDF)) <-:
  proximal_dist(PX, RX, PR, RR, pts(PD)),
  PDF = PD - D.
proximal_dist(PX, pts(RX), PR, pts(RR), pts(PD)) <-:
  dist_from_to(PX, PR, pts(D)),
  PD = D - RR - RX.
proximal_dir_dist(PX, pts(RX), PR, pts(RR), A, pts(PD)) <-:
  from_to_dir_dist(PX, PR, A, pts(D)),
  PD = D - RR - RX.
close_proximal_dist(pts(RX), pts(PD)) <-:
  PD =< 3.0 * RX.

%%
subinterval(intv(secs(TOA), secs(T1A)), intv(secs(TOB), secs(T1B))) <-:
  TOA =< TOB, TOB =< T1A,
  TOA =< T1B, T1B =< T1A,
  TOB < T1B.
dist_speed_dur(pts(M),  pts_sec(S),   secs(T)) <-:  M = S * T, M =\= 0.0.
dist_speed_dur(rads(R), rads_sec(S),  secs(T)) <-:  R = S * T, R =\= 0.0.
intv_dur(intv(secs(TO), secs(T1)), secs(T)) <-:
  T = T1 - TO, T =\= 0.0.

%%
interpolate_dist(P0, _,  D, P) <-:
  ident(P0, P), D = 0.0.
interpolate_dist(P0, P1, D, P) <-:
  ident(P1, P), dist_from_to(P0, P1, Dist), num(Dist, D).
interpolate_dist(pt(X0, Y0), pt(X1, Y1), D, pt(X, Y)) <-:
  dist_from_to(pt(X0, Y0), pt(X1, Y1), pts(D01)),
  D =< D01,
  T = D / D01,
  X = X0 + (X1 - X0) * T,
  Y = Y0 + (Y1 - Y0) * T.
interpolate_dist(nrads(R0), nrads(R1), D, nrads(R)) <-:
  dist_from_to(nrads(R0), nrads(R1), rads(D01)),
  D =< D01,
  T = D / D01,
  R = R0 nrads_plus (R1 nrads_minus R0) * T.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                                  minimize
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
minimize(Xs, BestD, D, DiffVs, DiffP, UpdateVs, UpdateP, Eps, MinXs, MinD) <-:
  append([BestD | Xs],     Xs1,      BestDXsXs1),
  append([D     | DiffVs], UpdateVs, DDiffVsUpdateVs),
  mult_subst(BestDXsXs1, DDiffVsUpdateVs, UpdateP,  UpdateP1),
  mult_subst([D1 | Xs1], [D | DiffVs],    DiffP,    DiffP1),
  call(UpdateP1),
  call(DiffP1),
  if(D1 >= BestD, (eq(Xs,  MinXs), eq(BestD, MinD)), true),
  if(D1 =< Eps,   (eq(Xs1, MinXs), eq(D1,    MinD)), true),
  if((Eps < D1, D1 < BestD),
     minimize(Xs1, D1, D, DiffVs, DiffP, UpdateVs, UpdateP, Eps, MinXs, MinD),
```

```
    true).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                                miscellaneous
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
new(S, E) <-:
  new,
  structure(S, E),
  functor(E, Prefix, _),
  id(E, Id),
  gensym(Prefix, Id).
current_rt(T) <-:
  once reference_time(T).
reference_time(secs(T)) <-: new, number(T).
actual(E) <-: not platonic(E).
platonic(_) <-: platonic.
error(_) <-: error.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                            keepers and assumables
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
keeper(reference_time/1).
keeper(platonic/1).
keeper(location/2).

assumable(new/0).
assumable(platonic/0).
assumable(error/0).
```

Appendix C

Meta-Interpreter Code

The code for the constraint-based abductive meta-interpreter described in chapter 5 appears below.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                    abductive constraint logic engine
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% global vars, defaults
:- dynamic eps/1, ic/1, mc/1, acc/1.

epsilon(E) :-
  (eps(E) -> true ; E = 0.001).
initial_cost(IC) :-
  (ic(IC) -> true ; IC = 10.0).
max_cost(MC) :-
  (mc(MC) -> true ; MC = 50.0).
accuracy(A) :-
  (acc(A) -> true ; A = 1.0).

%% cost-based divide-and-conquer deepening
abduce(GLCC, MaxCost, Accuracy, LCC) :-
  abduce(GLCC, _, none, 0.0, MaxCost, Accuracy, LCC).
abduce(GLCC, GLCC, LCC0, MinCost, MaxCost, Accuracy, LCC) :-
  MaxCost - MinCost < Accuracy, !, LCC0 = LCC.
abduce(GLCC, GLCC0, LCC0, MinCost, MaxCost, Accuracy, LCC) :-
  print(MinCost), print(' '), print(MaxCost), nl,
  HalfwayCost is MinCost + (MaxCost - MinCost) / 2,
  copy_term(GLCC, GLCC1),
  LCC = lcc(_, Cs, _),
  LCC1 = lcc(_, Cs, ActualCost),
  (abduce_cost(GLCC1, HalfwayCost, LCC1)
    -> abduce(GLCC, GLCC1, LCC1, MinCost,     ActualCost, Accuracy, LCC)
    ;  abduce(GLCC, GLCC0, LCC0, HalfwayCost, MaxCost,    Accuracy, LCC)).

%% thresholded abductive constraint logic engine
```

```
abduce_cost(glcc((Goal, Rest), Ls0, Cs0, C0), CT, LCC) :- !,
  abduce_cost(glcc(Goal, Ls0, Cs0, C0), CT, lcc(Ls1, Cs1, C1)),
  abduce_cost(glcc(Rest, Ls1, Cs1, C1), CT, LCC).
abduce_cost(glcc((GoalA ; GoalB), Ls, Cs, C), CT, LCC) :- !,
  (abduce_cost(glcc(GoalA, Ls, Cs, C), CT, LCC) ;
   abduce_cost(glcc(GoalB, Ls, Cs, C), CT, LCC)).
abduce_cost(GLCC, CostThreshold, lcc(Ls, Cs, C)) :-
  abd_rule(GLCC, CostThreshold, lcc(Ls, Cs, C)),
  C =< CostThreshold.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                  special form abduction rules
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% defined predicates
abd_rule(glcc(Goal $ _, Ls, Cs0, C), _, lcc(Ls, Cs, C)) :-
  unit_goal(Goal),
  defined_predicate(Goal, Call), !,
  call(Call),
  solve_constraints(Cs0, Cs).

%% call
abd_rule(glcc(call(Goal) $ GC, Ls, Cs, C), CT, LCC) :- !,
  propagate_cost(Goal, GC, AbdGoal),
  abduce_cost(glcc(AbdGoal, Ls, Cs, C), CT, LCC).

%% true
abd_rule(glcc(true $ _, Ls, Cs, C), _, lcc(Ls, Cs, C)) :- !.

%% fail
abd_rule(glcc(fail $ _, Ls, Cs, C), _, lcc(Ls, Cs, C)) :- !,
  fail.

%% once
abd_rule(glcc(once Goal $ GC, Ls, Cs, C), CT, LCC) :- !,
  (abduce_cost(glcc(Goal $ GC, Ls, Cs, C), CT, LCC) -> true ; fail).

%% assume
abd_rule(glcc(assume Goal $ _, Ls0, Cs, C), _, lcc(Ls, Cs, C)) :- !,
  Ls = [Goal | Ls0].

%% known
abd_rule(glcc(known Goal $ _, Ls, Cs0, C), _, lcc(Ls, Cs, C)) :- !,
  member(Goal, Ls),
  solve_constraints(Cs0, Cs).

%% local context entity
abd_rule(glcc(local_entity(E) $ _, Ls, Cs, C), _, lcc(Ls, Cs, C)) :- !,
  member(Pred, Ls),
  functor(Pred, _, N),
  arg(N, Pred, E).

%% if-then-else
abd_rule(glcc(if(If, Then, Else) $ GC, Ls, Cs, C), CT, LCC) :- !,
```

```
  propagate_cost(If, GC, AbdIf),
  (abduce_cost(glcc(AbdIf, Ls, Cs, C), CT, lcc(Ls1, Cs1, C1))
     -> (propagate_cost(Then, GC, AbdThen),
         abduce_cost(glcc(AbdThen, Ls1, Cs1, C1), CT, LCC))
     ;  (propagate_cost(Else, GC, AbdElse),
         abduce_cost(glcc(AbdElse, Ls, Cs, C), CT, LCC))).

%% freeze
abd_rule(glcc(freeze Goal $ GC, Ls, Cs, C), _, lcc(Ls, Cs, C)) :- !,
  abduce_cost(glcc(Goal $ GC, Ls, Cs, C), C, _).

%% negation-as-failure
abd_rule(glcc(not Goal $ GC, Ls, Cs, C), _, lcc(Ls, Cs, C)) :- !,
  \+ abduce_cost(glcc(Goal $ GC, Ls, Cs, C), C, _).

%% findall
abd_rule(glcc(findall(X, Goal, S) $ GC, Ls, Cs, C), _, lcc(Ls, Cs, C)) :- !,
  propagate_cost(Goal, GC, AbdGoal),
  my_findall(X, abduce_cost(glcc(AbdGoal, Ls, Cs, C), C, _), S).

%% setof
abd_rule(glcc(setof(X, Goal, S) $ GC, Ls, Cs, C), _, lcc(Ls, Cs, C)) :- !,
  propagate_cost(Goal, GC, AbdGoal),
  my_setof(X, abduce_cost(glcc(AbdGoal, Ls, Cs, C), C, _), S).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                        abduction rules proper
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% constraint as goal
abd_rule(glcc(Constraint $ _, Ls, Cs0, C), _, lcc(Ls, Cs, C)) :-
  unit_goal(Constraint),
  constraint(Constraint), !,
  normal_forms(Constraint, NFs),
  append(NFs, Cs0, Cs1),
  solve_constraints(Cs1, Cs).

%% goal factoring
abd_rule(glcc(Goal $ _, Ls, Cs0, C), _, lcc(Ls, Cs, C)) :-
  unit_goal(Goal),
  member(Goal, Ls),
  solve_constraints(Cs0, Cs).

%% goal assumption
abd_rule(glcc(Goal $ GC, Ls0, Cs, C0), _, lcc(Ls, Cs, C)) :-
  unit_goal(Goal),
  functor(Goal, F, A),
  assumable(F/A),
  (keeper(F/A)
     -> Ls = [Goal | Ls0]
     ;  Ls = Ls0),
  C is C0 + GC.

%% goal reduction
```

```
abd_rule(glcc(Goal $ GC, Ls, Cs, C), CT, LCC) :-
  non_unit_goal(Goal), !,
  propagate_cost(Goal, GC, AbdGoal),
  abduce_cost(glcc(AbdGoal, Ls, Cs, C), CT, LCC).
abd_rule(glcc(Goal $ GC, Ls0, Cs0, C0), CT, lcc(Ls, Cs, C)) :-
  (Goal <-: Body),
  propagate_cost(Body, GC, AbdBody),
  solve_constraints(Cs0, Cs1),
  abduce_cost(glcc(AbdBody, Ls0, Cs1, C0), CT, lcc(Ls1, Cs, C)),
  functor(Goal, F, A),
  (keeper(F/A)
     -> Ls = [Goal | Ls1]
     ;  Ls = Ls1).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                              miscellaneous
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% unit and non-unit goals
unit_goal(G) :-
  \+ non_unit_goal(G).
non_unit_goal((_, _)).
non_unit_goal((_ ; _)).

%% propagate goal cost to subgoals
propagate_cost((Goal0, Rest0), GC, (Goal, Rest)) :- !,
  propagate_cost(Goal0, GC, Goal),
  propagate_cost(Rest0, GC, Rest).
propagate_cost((GoalA0 ; GoalB0), GC, (GoalA ; GoalB)) :- !,
  propagate_cost(GoalA0, GC, GoalA),
  propagate_cost(GoalB0, GC, GoalB).
propagate_cost(Goal $ Weight, GC, Goal $ Cost) :- !,
  Cost is GC * Weight.
propagate_cost(Goal, GC, Goal $ GC).

%% defined predicates
defined_predicate(eq(X, Y),    X = Y).
defined_predicate(ident(X, Y), X == Y).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                           constraint solver
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% identify constraints (etc)
constraint(_  =  _).
constraint(_ >=  _).

%% constraint normal forms
normal_forms(C, Cs) :-
  functor(C, F, 2),
  arg(1, C, Arg1),
  arg(2, C, Arg2),
  reduce_term(Arg1, A, CsA),
```

```
  reduce_term(Arg2, B, CsB),
  functor(NewC, F, 2),
  arg(1, NewC, A),
  arg(2, NewC, B),
  append([CsA, CsB, [NewC]], Cs0),
  unify_simple_eqs(Cs0, Cs).
reduce_term(X, X, []) :- (var(X) ; number(X)), !.
reduce_term(X, A, Cs) :-
  functor(X, F, N),
  functor(Y, F, N),
  reduce_term_n(X, Y, CsY, N),
  Cs = [A = Y | CsY].
reduce_term_n(_, _, [], 0) :- !.
reduce_term_n(X, Y, Cs, N) :-
  arg(N, X, ArgN),
  arg(N, Y, A),
  reduce_term(ArgN, A, CsA),
  N1 is N - 1,
  reduce_term_n(X, Y, CsRest, N1),
  append(CsA, CsRest, Cs).
unify_simple_eqs([], []).
unify_simple_eqs([X = Y | Tail], Rest) :-
  (var(Y) ; number(Y)), !,
  X = Y,
  unify_simple_eqs(Tail, Rest).
unify_simple_eqs([C | Tail], [C | Rest]) :-
  unify_simple_eqs(Tail, Rest).

%% iterate until no more changes or failure
solve_constraints(C0, C) :-
  solve_constraints(C0, C1, Status),
  (Status = unchanged
     -> C = C1
     ;  (Status = changed
           -> solve_constraints(C1, C)
           ;  fail)).

%% invoke local propagation
solve_constraints([], [], unchanged).
solve_constraints([Eq | Tail], Tail, Status) :-
  inst(Eq), !,
  (solve_eq(Eq) -> Status = changed ; Status = failed).
solve_constraints([Eq | Tail], [Eq | Rest], Status) :-
  solve_constraints(Tail, Rest, Status).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
/*
                              local propagation
*/
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% sufficiently instantiated eqs (etc)
inst(_ = pi).
inst(Eq) :-
  ([Eq] = [A = B + C] ;
   [Eq] = [A = B - C] ;
```

```
   [Eq] = [A = B nrads_minus C]),
  ((number(B), number(C)) ;
   (number(A), number(C)) ;
   (number(A), number(B))),
  \+ zero_divisor(Eq).
inst(Eq) :-
  ([Eq] = [A = -B]),
  (number(A) ; number(B)).

%% zero divisor
zero_divisor(A = B * C) :-
  (number(B), number(C))
    -> fail
    ;  ((number(A), number(C), within_epsilon(C, 0.0)) ;
        (number(A), number(B), within_epsilon(B, 0.0))).
zero_divisor(A = B / C) :-
  (number(A), number(C))
    -> fail
    ;  ((number(B), number(C), within_epsilon(C, 0.0)) ;
        (number(A), number(B), within_epsilon(A, 0.0))).

%% solve equations (etc)
solve_eq(A = pi) :-
  pi(rads(A0)),
  nrads_within_epsilon(A0, A).
solve_eq(A = B + C) :-
  number(B), number(C),
  A0 is B + C,
  within_epsilon(A0, A).

%% M within epsilon of N
within_epsilon(M, N) :-
  number(M), var(N), M = N.
within_epsilon(M, N) :-
  var(M), number(N), M = N.
within_epsilon(M, N) :-
  number(M), number(N),
  Diff is M - N,
  abs(Diff, AbsDiff),
  epsilon(E),
  AbsDiff < E.
nrads_within_epsilon(M, N) :-
  within_epsilon(M, N).
nrads_within_epsilon(M, N) :-
  number(M), number(N), pi(rads(Pi)),
  M1 is M + 2 * Pi,
  within_epsilon(M1, N).
nrads_within_epsilon(M, N) :-
  number(M), number(N), pi(rads(Pi)),
  N1 is N + 2 * Pi,
  within_epsilon(M, N1).
```

Bibliography

James Allen. 1984. Towards a general theory of action and time. *Artificial Intelligence*, 23(2).

Emmon Bach. 1986. The algebra of events. *Linguistics and Philosophy*, 9:5–16.

Norman Badler, Bonnie Webber, Jeff Esakov, and Jugal Kalita. 1990. Animation from instructions. In Badler, Barsky, and Zeltzer, editors, *Making them Move*. MIT Press.

Norman I. Badler, Ramamani Bindiganavale, John P. Granieri, Susanna Wei, and Xinmin Zhao. 1994. Posture interpolation with collision avoidance. In *Computer Animation*, pages 13–20, Geneva, Switzerland.

Norman I. Badler, Cary B. Phillips, and Bonnie Lynn Webber. 1993. *Simulating Humans: Computer Graphics, Animation, and Control*. Oxford University Press, June.

Norman I. Badler. 1975. Temporal scene analysis: Conceptual descriptions of object movements. Technical Report 80, University of Toronto.

Jon Barwise and Robin Cooper. 1981. Generalized quantifiers and natural language. *Linguistics and Philosophy*, 4:159–219.

Robert I. Binnick. 1991. *Time and the Verb*. Oxford University Press.

Mario Borillo and Bruno Gaume. 1989. An extension to Kowalski & and Sergot's event calculus. Manuscript.

Mimo Caenepeel. 1989. *Aspect, Temporal Ordering and Perspective in Narrative Fiction*. Ph.D. thesis, University of Edinburgh.

Greg Carlson. 1977a. *Reference to kinds in English*. Ph.D. thesis, University of Massachussets, Amherst.

Greg Carlson. 1977b. A unified analysis of the English bare plural. *Linguistics and Philosophy*, 1:413–457.

Eugene Charniak and Solomon E. Shimony. 1990. Probabilistic semantics for cost-based abduction. In *Proceedings of the 1990 National Conference on Artificial Intelligence*. pp. 106-111.

Stephen Crain and Mark Steedman. 1985. On not being led up the garden path: The use of context by the psychological parser. In D. Dowty, L. Karttunen, and A. Zwicky, editors, *Natural Language Parsing: Psychological, Computational and Theoretical Perspectives*. Cambridge University Press.

Robert Dale. 1992. *Generating Referring Expressions: Constructing Descriptions in a Domain of Objects and Processes.* MIT Press.

David R. Dowty, Robert E. Wall, and Stanley Peters. 1981. *Introduction to Montague Semantics.* Reidel.

David R. Dowty. 1972. *Studies in the Logic of Verb Aspect and Time Reference.* Ph.D. thesis, University of Texas.

David R. Dowty. 1979. *Word Meaning and Montague Grammar.* Reidel.

David R. Dowty. 1982. Tenses, time adverbs, and compositional semantic theory. *Linguistics and Philosophy*, 5:23–55.

David Dowty. 1986. The effects of aspectual class on the temporal structure of discourse: Semantics or pragmatics. *Linguistics and Philosophy*, 9(1).

David Dowty. 1988. Type raising, functional composition, and non-constituent conjunction. In R. Oehrle, E. Bach, and D. Wheeler, editors, *Categorial Grammars and Natural Language Structures.* Reidel.

David Dowty. 1989. On the semantic content of the notion 'thematic role'. In Barbara Partee, Gennaro Chierchia, and Ray Turner, editors, *Properties, Types, and Meanings*, volume 2, pages 69–130. Kluwer, Dordrecht.

David Dowty. 1991. Thematic proto-roles and argument selection. *Language*, 67(3):547–615.

Helen Dry. 1981. Sentence aspect and the movement of narrative time. *Text*, 1(3):233–240.

Kurt Eberle. 1990. Eventualities in natural language understanding systems. In *Sorts and Types in Artificial Intelligence.* Springer Verlag.

Kurt Eberle. 1992. On representing the temporal structure of a natural language text. In *Proceedings of the Fifteenth International Conference on Computational Linguistics, COLING 92.*

Lee Fedder. 1991. *Generating Natural Language Text from the output of an Application Program.* Ph.D. thesis, University of Cambridge.

Charles J. Fillmore. 1982. Towards a descriptive framework for spatial deixis. In Robert J. Jarvella and Wolfgang Klein, editors, *Speech, Place, and Action.* John Wiley and Sons, Chichester.

Simon C. Garrod and Anthony J. Sanford. 1988. Discourse models as interfaces between language and the spatial world. *Journal of Semantics*, 6:147–160.

John Goldsmith and Erich Woisetschlaeger. 1982. The logic of the english progressive. *Linquistic Inquiry*, 13(1):79–89.

H.P. Grice. 1975. Logic and conversation. In P. Cole and J. Morgan, editors, *Syntax and Semantics III - Speech Acts*, pages 41–58. Academic Press, New York.

Carl A. Gunter. 1992. *Semantics of Programming Languages: Structures and Techniques.* The MIT Press.

Christopher Habel. 1990. Propositional and depictorial representations of spatial knowledge: The case of *path*-concepts. In *Natural Language and Logic.* Springer Verlag. Lecture Notes in Artificial Intelligence.

Nicholas J. Haddock. 1987. Incremental interpretation and combinatory categorial grammar. In *Proceedings of the 10th International Joint Conference on Artificial Intelligence*, pages 661–663.

Galia Hatav. 1989. Aspects, aktionsarten, and the time line. *Linguistics*, 27:487–516.

Ellen M. Hays. 1989. On defining motion verbs and spatial relations. Technical Report 61, Universität des Saarlandes. SFB 314 (VITRA).

U. Hedtstück and P. H. Schmitt. 1990. A calculus for order-sorted predicate logic with sort literals. In *Sorts and Types in Artificial Intelligence.* Springer Verlag.

Irene Heim. 1983. File change semantics and the familiarity theory of definiteness. In R. Bauerle, C. Schwarze, and A von Stechow, editors, *Meaning, use, and the interpretation of language.* Walter de Gruyter, Berlin.

Annette Herskovits. 1986. *Language and Spatial Cognition.* Cambridge University Press.

Erhard Hinrichs. 1985. *A Compositional Semantics for Aktionsarten and NP Reference in English.* Ph.D. thesis, The Ohio State University.

Erhard Hinrichs. 1986. Temporal anaphora in discourses of English. *Linguistics and Philosophy*, 9(1).

Erhard Hinrichs. 1988. Tense, quantifiers, and contexts. *Computational Linguistics*, June.

Jerry Hobbs, Mark Stickel, Paul Martin, and Douglas Edwards. 1988. Interpretation as abduction. In *Proceedings of ACL.*

Jerry Hobbs, Mark Stickel, Douglas Appelt, and Paul Martin. 1993. Interpretation as abduction. *Artificial Intelligence Journal*, 63:69–142.

Chung Hee Hwang and Lenhart K. Schubert. 1992. Tense trees as the "fine structure" of discourse. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics.*

Ray Jackendoff and Barbara Landau. 1991. Spatial language and spatial cognition. *LEA.*

Ray Jackendoff. 1990. *Semantic Structures.* MIT Press.

Ray Jackendoff. 1991. Parts and boundaries. *Cognition*, 41:9–45.

Ray Jackendoff. 1993. The proper treatment of measuring out, telicity, and perhaps even quantification in English. Manuscript.

Joxan Jaffar and Jean-Louis Lassez. 1987. Constraint logic programming. In *Proceedings of the 14th ACM Symposium on the Principles of Programming Languages*, pages 111–119.

Joxan Jaffar and Spiro Michaylov. 1987. Methodology and implementation of a CLP system. In Jean-Louis Lassez, editor, *Fourth International Conference on Logic Programming*, pages 196–218. MIT Press.

A. C. Kakas, R. A. Kowalski, and F. Toni. 1992. Abductive logic programming. *J. Logic Computat.*, 2(6):719–770.

Jugal Kumar Kalita. 1990. *Natural Language Control of Animation of Task Performance in a Physical Domain.* Ph.D. thesis, University of Pennsylvania, Philadelphia, PA.

Hans Kamp and Uwe Reyle. 1993. *From Discourse to Logic: Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory.* Kluwer Academic Publishers.

Hans Kamp and Christian Rohrer. 1983. Tense in texts. In R. Bäuerle, C. Schwarze, and A. von Stechow, editors, *Meaning, Use and Interpretation in Language*, pages 250–269. Walter de Gruyter.

Hans Kamp. 1981. A theory of truth and semantic representation. In J. Groenendijk, T. Janssen, and M. Stokhof, editors, *Formal Methods in the Study of Language*, pages 277–322. Mathematical Centre Tracts, Amsterdam.

Tadashi Kanamori and Tadashi Kawamura. 1993. Abstract interpretation based on OLDT resolution. *Journal of Logic Programming*, 15(1 & 2):1–30, January.

Stuart John Harding Kent. 1993. *Modelling Events from Natural Language.* Ph.D. thesis, Imperial College, University of London.

Robert Kowalski and Marek Sergot. 1986. A logic-based calculus of events. *New Generation Computing*, 4.

Robert Kowalski. 1979. *Logic for Problem Solving.* The Computer Science Library, Artificial Intelligence Series. North Holland, New York.

Manfred Krifka, Jeff Pelletier, Greg Carlson, Alice ter Meulen, Godehard Link, and Gennaro Chierchia. 1992. Genericity: An introduction. Manuscript.

Manfred Krifka. 1989. Nominal reference, temporal constitution and quantification in event semantics. In R. Bartsch, J. van Benthem, and P. van Emde Boas, editors, *Semantics and Contextual Expressions*, pages 75–115. Dordrecht.

Manfred Krifka. 1990. Four thousand ships passed through the lock: Object-induced measure functions on events. *Linguistics and Philosophy*, 13:487–520.

Manfred Krifka. 1992. Thematic relations as links between nominal reference and temporal constitution. In Ivan A. Sag and Anna Szabolcsi, editors, *Lexical Matters.* CSLI.

Fred Landman. 1989a. Groups, I. *Linguistics and Philosophy*, 12:559–605.

Fred Landman. 1989b. Groups, II. *Linguistics and Philosophy*, 12:723–744.

Fred Landman. 1991. *Structures for Semantics.* Kluwer Academic Publishers.

Fred Landman. 1992. The progressive. *Natural Language Semantics*, 1:1–32.

Alex Lascarides and Nicholas Asher. 1991. Discourse relations and defeasible knowledge. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics.*

Alex Lascarides, Nicholas Asher, and Jon Oberlander. 1992. Inferring discourse relations in context. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics.*

Beth Levin. 1993. *English Verb Classes and Alternations: A Preliminary Investigation.* The University of Chicago Press.

Gödehard Link. 1983. The logical analysis of plurals and mass terms. In R. Bauerle, C. Schwarze, and A. von Stechow, editors, *Meaning, Use, and Interpretation of Language.* de Gruyter.

Gödehard Link. 1987. Algebraic semantics of event structures. In J. Groenendijk, M. Stokhof, and F. Veltman, editors, *Proceedings of the Sixth Amsterdam Colloquium.*

Rolf Mayer. 1989. Coherence and motion. *Linguistics*, pages 437–485.

John McCarthy and P.J. Hayes. 1969. Some philosphical problems from the standpoint of artificial intelligence. In *Machine Intelligence 4*. Edinburgh University Press, Edinburgh.

Dale Miller. 1991. A logic programming language with lambda abstraction, function variables and simple unification. *Journal of Logic and Computation*, 1(4):497–536.

Anita Mittwoch. 1982. On the difference between *eating* and *eating something*: Activities versus accomplishments. *Linguistic Inquiry*.

Marc Moens and Mark Steedman. 1988. Temporal ontology and temporal reference. *Computational Linguistics*, June.

Marc Moens. 1987. *Tense, Aspect and Temporal Reference*. Ph.D. thesis, University of Edinburgh.

Friederike Moltmann. 1991. Measure adverbials. *Linguistics and Philosophy*, 14:629–660.

Richard Montague. 1974. The proper treatment of quantification in ordinary english. In *Formal Philosophy: Selected Papers of Richard Montague*. Yale University Press.

Robert C. Moore. 1989. Unification-based semantic interpretation. In *Proccedings of the 27th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Gopalan Nadathur and Dale Miller. 1988. An overview of λprolog. Technical Report MS-CIS-88-40, University of Pennsylvania, June.

Alexander Nakhimovksy. 1988. Aspect, aspectual class, and the temporal structure of narrative. *Computational Linguistics*, June.

G. Nunberg. 1979. The nonuniqueness of semantic solutions: Polysemy. *Linguistics and Philosophy*, 3.

Jon Oberlander and Robert Dale. 1991. Generating expressions referring to eventualities. In *Proceedings of the 13th Annual Conference of the Cognitive Science Society, University of Chicago*.

Arnold Oberschelp. 1990. Order sorted predicate logic. In *Sorts and Types in Artificial Intelligence*. Springer Verlag.

Richard T. Oehrle. 1990. Tense in English complement clauses. Manuscript.

Terence Parsons. 1990. *Events in the Semantics of English: A Study in Subatomic Semantics*, volume 21 of *Current Studies in Linguistics*. MIT Press.

Barbara Partee, Alice ter Meulen, and Robert Wall. 1990. *Mathematical Methods in Linguistics*. Kluwer Academic Publishers.

Barbara Partee. 1984. Nominal and temporal anaphora. *Linguistics and Philosophy*, 7(3).

Francis Jeffry Pelletier and Lenhart K. Schubert. 1989. Mass expressions. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, chapter IV.4, pages 327–407. D. Reidel Publishing Company.

Fernando C. N. Pereira and Stuart M. Shieber. 1987. *Prolog and Natural-Language Analysis*. Number 10 in CSLI Lecture Notes. Center for the Study of Language and Information, Stanford, California. Distributed by Chicago University Press.

Fernando C. N. Pereira and David H. D. Warren. 1983. Parsing as deduction. In *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*, pages 137–144.

Fernando C. N. Pereira. 1990. Prolog and natural-language analysis: into the third decade. In Saumya Debray and Manuel Hermenegildo, editors, *Logic Programming: Proceedings of the 1990 North American Conference*. MIT Press, November.

Jeremy V. Pitt, Karl G. Van den Bergh, and Jim Cunningham. 1992. An experiment in discourse understanding. Technical Report DoC92/1, Department of Computing, Imperial College of Science, echnology and Medicine, London, England.

David Poole. 1989. Explanation and prediction: An architecture for default and abductive reasoning. *Computational Intelligence*, 5:97–110.

James Pustejovsky. 1991a. The generative lexicon. *Computational Linguistics*, December.

James Pustejovsky. 1991b. The syntax of event structure. *Cognition*, 41:47–81.

Willard Van Orman Quine. 1960. *Word and Object*. MIT Press, 1985 edition.

Terrance Philip Regier. 1992. *The Acquisition of Lexical Semantics for Spatial Terms: A Connectionist Model of Perceptual Categorization*. Ph.D. thesis, University of California at Berkeley.

Tanya Reinhart. 1984. Principles of gestalt perception in the temporal organization of narrative texts. *Linguistics*, 22:779–809.

Gudula Retz-Schmidt. 1988. Various views on spatial prepositions. *AI Magazine*, 9(2):95–105.

Fariba Sadri and Robert Kowalski. 1988. A theorem-proving approach to database integrity. In Jack Minker, editor, *Foundations of deductive databases and logic programming*, pages 313–362. Morgan Kaufmann.

Görel Sandström. 1992. The temporal interpretation of *when*-clauses in narratives. In *Proceedings of the Thirteenth Scandinavian Conference of Linguistics*.

Görel Sandström. 1993. When-*clauses and the temporal interpretation of narrative discourse*. Ph.D. thesis, University of Umeå.

J.R.J. Schirra, G. Bosch, C.K. Sung, and G. Zimmermann. 1987. From image sequences to natural language: A first step toward automatic perception and description of motions. *Applied Artificial Intelligence*, 1:287–305.

Lenhart K. Schubert and Francis Jeffry Pelletier. 1987. Problems in the representation of the logical form of generics, plurals, and mass nouns. In *New Directions in Semantics*, pages 385–451. Academic Press.

Roger Schwarzschild. 1992. Types of plural individuals. *Linguistics and Philosophy*, 15:641–675.

Yoav Shoham. 1987. Temporal logics in AI: Semantical and ontological considerations. *Artificial Intelligence*, 33(1):89–104.

Jeffrey Mark Siskind and David Allen McAllester. 1993. Nondeterministic LISP as a substrate for constraint logic programming. In *Proc. National Conference on Artificial Intelligence*.

Jeffrey Mark Siskind. 1992. *Naive Physics, Event Perception, Lexical Semantics, and Language Acquisition*. Ph.D. thesis, MIT.

Carlota S. Smith. 1981. Semantic and syntactic constraints on temporal interpretation. In Philip J. Tedeschi and Annie Zaenen, editors, *Tense and Aspect*, volume 14 of *Syntax and Semantics*. Academic Press.

Robert Stalnaker. 1968. A theory of conditionals. In N. Rescher, editor, *Studies in Logical Theory*. Blackwell, London.

Mark Steedman. 1982. Reference to past time. In R. J. Jarvella and W. Klein, editors, *Speech, Place, and Action*, pages 125–157. John Wiley and Sons Ltd.

Mark Steedman. 1991a. Structure and Intonation. *Language*, 68(2):260–296.

Mark Steedman. 1991b. Type-Raising and Directionality in Combinatory Grammar. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics.*

Mark E. Stickel. 1990. Rationale and methods for abductive reasoning in natural language interpretation. In Rudi Studer, editor, *Natural Language and Logic*, pages 233–252. Springer-Verlag. Lecture Notes in Artificial Intelligence.

Mark E. Stickel. 1991. A Prolog-like inference system for computing minimum-cost abductive explanations in natural-language interpretation. *Annals of Mathematics and Artificial Intelligence*, 4:89–106.

Hisao Tamaki and Taisuke Sato. 1986. OLD resolution with tabulation. In Ehud Shapiro, editor, *Proceedings of the 3rd International Conference on Logic Programming*, pages 84–98, London, June.

Carol Tenny. 1992. The aspectual interface hypothesis. In Ivan A. Sag and Anna Szabolcsi, editors, *Lexical Matters.* CSLI.

Robert Thibadeau. 1986. Artificial perception of actions. *Cognitive Science*, 10(2):117–149.

Pascal Van Hentenryck. 1989. *Constraint Satisfaction in Logic Programming.* MIT Press, Cambridge, MA.

Zeno Vendler. 1967. *Linguistics in Philosophy.* Cornell University Press.

Henk Verkuyl and Joost Zwarts. 1992. Time and space in conceptual and logical semantics: the notion of path. *Linguistics*, pages 483–511.

H. J. Verkuyl. 1972. *On the Compositional Nature of the Aspects.* Reidel.

Henk Verkuyl. 1987. Nondurative closure of events. In Jeroen Groenendijk, Dick de Jongh, and Martin Stokhof, editors, *Studies in Discourse Representation Theory and the Theory of Generalized Quantifiers*, pages 87–113. Foris, Dordrecht.

H. J. Verkuyl. 1989. Aspectual classes and aspectual composition. *Linguistics and Philosophy*, 12:39–94.

Henk J. Verkuyl. 1993. *A theory of aspectuality: The interaction between temporal and atemporal structure*, volume 64 of *Cambridge Studies in Linguistics.* Cambridge University Press.

Frank Vlach. 1981. The semantics of the progressive. In P. Tedeschi and A. Zaenen, editors, *Syntax and Semantics 14: Tense and Aspect.* Academic Press, New York.

Christopher Walther. 1990. Many-sorted inferences in automated theorem proving. In *Sorts and Types in Artificial Intelligence*. Springer Verlag.

Bonnie Webber, Norman Badler, Barbara Di Eugenio, Libby Levison, and Michael White. 1991. Instructing Animated Agents. In *Proc. US-Japan Workshop on Integrated Systems in Multi-Media Environments. Las Cruces, NM.*

Bonnie Lynn Webber. 1988. Tense as discourse anaphor. *Computational Linguistics*, June.

Michael White. 1992. Deictic Temporal Locative vs. Temporal Measure *in.* In *Proceedings of the Penn Review of Linguistics*, volume 16.

Alessandro Zucchi. 1993. Aspect shift. Manuscript, August.

Joost Zwarts and Henk Verkuyl. 1994. An algebra of conceptual structure; an investigation into Jackendoff's conceptual semantics. *Linguistics and Philosophy*, 17(1):1–28.