12-1987

A Queueing Analysis of Hashing With Lazy Deletion

John A. Morrison

Larry A. Shepp
University of Pennsylvania

Christopher J. Van Wyk

Recommended Citation

Morrison, J. A., Shepp, L. A., & Van Wyk, C. J. (1987). A Queueing Analysis of Hashing With Lazy Deletion. *SIAM Journal on Computing, 16* (6), 1155-1164. http://dx.doi.org/10.1137/0216073

A Queueing Analysis of Hashing With Lazy Deletion

Abstract
Hashing with lazy deletion is a simple method for maintaining a dynamic dictionary: items are inserted and sought as usual in a separate-chaining hash table; however, items that no longer need to be in the data structure remain until a later insertion operation stumbles on them and removes them from the table. Because hashing with lazy deletion does not delete items as soon as possible, it keeps more items in the dictionary than methods that use more careful deletion strategies. On the other hand, its space overhead is much smaller than those more careful methods, so if the number of extra items is not too large, hashing with lazy deletion can be a practical algorithm when space is scarce. In this paper, we analyze the expected amount of excess space used by hashing with lazy deletion.

Keywords
dynamic dictionary, first passage times, storage

SIAM J. COMPUT.
Vol. 16, No. 6, December 1987

012

A QUEUEING ANALYSIS OF HASHING WITH LAZY DELETION*

JOHN A. MORRISON†, LARRY A. SHEPP† AND CHRISTOPHER J. VAN WYK†

Abstract. Hashing with lazy deletion is a simple method for maintaining a dynamic dictionary: items are inserted and sought as usual in a separate-chaining hash table; however, items that no longer need to be in the data structure remain until a later insertion operation stumbles on them and removes them from the table. Because hashing with lazy deletion does not delete items as soon as possible, it keeps more items in the dictionary than methods that use more careful deletion strategies. On the other hand, its space overhead is much smaller than those more careful methods, so if the number of extra items is not too large, hashing with lazy deletion can be a practical algorithm when space is scarce. In this paper, we analyze the expected amount of excess space used by hashing with lazy deletion.

Key words. dynamic dictionary, first passage times, storage

AMS(MOS) subject classifications. 60K30, 68A50

1. Introduction.

The strategy of hashing with lazy deletion was proposed by Van Wyk and Vitter [1] for use in the following situation:

> A sequence of items is given; each item includes a search key, a starting time, and an expiration time. The items arrive in the order of their starting times, and each item must be kept in a dynamic dictionary (available for searching) until the arrival of an item whose starting time is later than the item's expiration time.

Algorithms that delete items as soon as possible have unacceptably high overhead in space [1], even though they require less storage space for the items themselves. Hashing with lazy deletion means keeping the items hashed by search key in a table of linked lists (separate chains); each time an item is added to a list, any items on that list that the new item shows to be expired are deleted from that list. This deletion procedure is "lazy" because there is no separate operation associated with clearing expired items out of the table: expired items are only deleted when they are encountered during an insertion operation.

Hashing with lazy deletion was used to solve a geometric adjacency problem that arose in the analysis of integrated circuits [2]. The items were line segments in the plane: starting time corresponded to minimum x-coordinate, expiration time to maximum x-coordinate, and search key to y-intercept. At any abscissa x all segments that cross the vertical line at x must be in the dictionary. The artwork for a chip of modest size (around 100,000 transistors) can involve three to ten million line segments. But it is reasonable to expect that at most 50,000 segments cross any vertical line, so the dictionary only needs to store about 50,000 segments. Hashing with lazy deletion uses 24 bytes per segment, while hashing with careful deletion requires 36 bytes per segment. Thus, using careful deletion, the dictionary can fill around 1.8 megabytes; even if hashing with lazy deletion keeps 20% more segments in the dictionary than are necessary, and thus requires 1.44 megabytes plus the size of the hash table, it still uses less space than the careful deletion strategy.

Given a sequence of items, let $N(t)$ be the number of items that start at or before time t and expire at or after time t. The statement of the problem requires that these items be in the dictionary at time t, so any algorithm used to solve this problem must

* Received by the editors February 3, 1986; accepted for publication (in revised form) March 24, 1987.
† AT&T Bell Laboratories, Murray Hill, New Jersey 07974.

use space proportional to $N(t)$ at time t; in this sense, an algorithm whose space complexity is $N(t)$ for all t has *optimum* space complexity. Given the same sequence of items and the number H of buckets in a hash table being managed with lazy deletion, let $U_H(t)$ be the number of items present in the table at time t; $U_H(t)$ is the space complexity of hashing with lazy deletion. Note that for all H and all t, $U_H(t) \geqq N(t)$.

Under the assumptions that the starting times of items form a Poisson process, that item life-times are independent and exponentially distributed, and that the hash function is uniform on the search keys, Van Wyk and Vitter showed that the expected value of $U_H(t)$ is H more than the expected value of $N(t)$. This suggests that it is possible to choose H so that the expected search time is constant while space complexity is little more than optimum.

In this paper we obtain more information about the space complexity of hashing with lazy deletion by studying the values of $\max_t N(t)$ and $\max_t U_H(t)$, where $0 \leqq t \leqq T$. The quantity $\max_t N(t)$ is a lower bound on the space required by any algorithm that solves the problem, while $\max_t U_H(t)$ is the space required by hashing with lazy deletion using H buckets. Suppose the amount of space available to solve the problem is S. If $\max_t U_H(t) > S \geqq \max_t N(t)$, then hashing with lazy deletion cannot solve the problem even though an algorithm with a more careful deletion strategy could solve the problem in space S. Hence we wish to study the amount by which $\max_t U_H(t)$ exceeds $\max_t N(t)$; note that the expressions $N(t)$ and $U_H(t)$ may attain their maxima at different values of t.

In § 2 we derive an expression for $\max_t N(t)$ using techniques from queueing theory. In § 3 we derive an expression for $\max_t U_1(t)$ by studying the following queueing problem:

> The door of a bank is locked from the inside, so customers can enter at any time, but customers who have completed their transactions must wait in the lobby for an arriving customer before they can leave; assuming that customers are served immediately, what is the total number of customers—in service and waiting to leave—in the bank?

This queueing problem directly models the behavior of hashing with lazy deletion using a single bucket ($H = 1$). Section 4 shows how the results in §§ 2 and 3 provide bounds on the space complexity of hashing with lazy deletion using H buckets. Section 5 presents some numerical results.

2. Analysis of $N(t)$. We first consider the case of an M/M/∞ queue, with Poisson arrival rate λ, and mean service time $1/\mu$. The equilibrium probability p_j that there are j customers in the system is [3]

$$p_j = \frac{a^j}{j!} e^{-a}, \qquad a = \frac{\lambda}{\mu}. \tag{2.1}$$

Let $N(t)$ denote the number of customers in the system at time t. We are interested in the distribution of $\max_{0 \leqq t \leqq T} N(t)$, given that the system is in equilibrium at $t = 0$. Let $s_{j,k}(t)$ denote the density of the passage time from j to k. Now, $\max_{0 \leqq t \leqq T} N(t)$ will be less than k if, and only if, the system is initially in some state $j < k$, and the passage time from j to k is greater than T. Hence,

$$\begin{aligned} \Pi_k(T) &\equiv \Pr\left\{ \max_{0 \leqq t \leqq T} N(t) < k \,\middle|\, \text{equilibrium at } t = 0 \right\} \\ &= \sum_{j=0}^{k-1} p_j \int_T^\infty s_{j,k}(t)\, dt. \end{aligned} \tag{2.2}$$

We now turn our attention to the calculation of $s_{j,k}(t)$. We use the method of Siegert (cf. [4] and [5]), which is possible because $N(t)$ increases in unit steps. Let

$$\sigma_m(\xi) = \int_0^\infty e^{-\xi t} s_{m,m+1}(t)\, dt \tag{2.3}$$

be the Laplace transform of the upward passage time density at m. Then [6],

$$\int_0^\infty e^{-\xi t} s_{j,k}(t)\, dt = \prod_{m=j}^{k-1} \sigma_m(\xi), \qquad k > j. \tag{2.4}$$

Also [6], since the departure rate from state m is $m\mu$,

$$(\xi + \lambda + m\mu)\sigma_m = \lambda + m\mu\sigma_{m-1}\sigma_m, \qquad m = 0, 1, \cdots. \tag{2.5}$$

We define

$$\omega_{m+1} = \left(\prod_{l=0}^{m} \sigma_l \right)^{-1}, \qquad m = 0, 1, \cdots, \quad \omega_0 = 1, \tag{2.6}$$

and let

$$\eta = \frac{\xi}{\mu}, \qquad a = \frac{\lambda}{\mu}. \tag{2.7}$$

Then, from (2.5), we obtain

$$(\eta + a + m)\omega_m = a\omega_{m+1} + m\omega_{m-1}, \qquad m = 0, 1, \cdots. \tag{2.8}$$

Next, we let

$$\theta_m = a^m \omega_m / m!. \tag{2.9}$$

Hence, from (2.6) and (2.8), we find that

$$(\eta + a + m)\theta_m = (m+1)\theta_{m+1} + a(1 - \delta_{m0})\theta_{m-1}, \qquad \theta_0 = 1, \tag{2.10}$$

where δ denotes the Kronecker symbol. It follows from (2.10) that θ_m is a polynomial of degree m in both η and a, and it can be related to a Poisson-Charlier polynomial. In the notation of [7],

$$\theta_m = s_m(\eta, a). \tag{2.11}$$

From (2.6), (2.7), (2.9) and (2.11), we obtain

$$\prod_{m=j}^{k-1} \sigma_m(\xi) = \frac{j!\, a^k s_j(\xi/\mu, a)}{k!\, a^j s_k(\xi/\mu, a)}, \qquad 0 \leqq j < k. \tag{2.12}$$

The zeros of $s_k(\eta, a)$, regarded as a function of η, are negative and distinct [7], and we denote them by $\eta_l = -\kappa_l(a)$, $l = 1, \cdots, k$. Then

$$\lim_{\xi \to -\mu\kappa_l} (\xi + \mu\kappa_l) \frac{s_j(\xi/\mu, a)}{s_k(\xi/\mu, a)} = \frac{\mu s_j(-\kappa_l, a)}{s_k'(-\kappa_l, a)}, \tag{2.13}$$

where the prime denotes derivative with respect to the first argument. It follows, from (2.4), (2.12) and (2.13), that

$$s_{j,k}(t) = \mu \frac{j!\, a^k}{k!\, a^j} \sum_{l=1}^{k} \frac{s_j(-\kappa_l, a)}{s_k'(-\kappa_l, a)} e^{-\mu\kappa_l t}, \qquad 0 \leqq j < k. \tag{2.14}$$

But [7], for $|u| < 1$,

$$\sum_{k=0}^{\infty} s_k(x, a) u^k = e^{au}(1-u)^{-x} = e^{au}\, e^{-x \log(1-u)}. \tag{2.15}$$

Hence,

$$\sum_{k=0}^{\infty} s'_k(x, a)u^k = -\log(1-u)\, e^{au}\, e^{-x\log(1-u)} = \sum_{r=1}^{\infty} \frac{u^r}{r} \sum_{m=0}^{\infty} s_m(x, a)u^m. \tag{2.16}$$

This implies that

$$s'_k(x, a) = \sum_{m=0}^{k-1} \frac{s_m(x, a)}{(k-m)}. \tag{2.17}$$

We may now obtain an expression for $\Pi_k(T)$. From (2.1), (2.2) and (2.14), it is found that

$$\Pi_k(T) = \frac{a^k}{k!} e^{-a} \sum_{j=0}^{k-1} \sum_{l=1}^{k} \frac{s_j(-\kappa_l, a)}{\kappa_l s'_k(-\kappa_l, a)} e^{-\mu\kappa_l T}. \tag{2.18}$$

But [7],

$$\sum_{j=0}^{n} s_j(x, a) = s_n(1+x, a). \tag{2.19}$$

Hence

$$\Pi_k(T) = \frac{a^k}{k!} e^{-a} \sum_{l=1}^{k} \frac{s_{k-1}(1-\kappa_l, a)}{\kappa_l s'_k(-\kappa_l, a)} e^{-\mu\kappa_l T}. \tag{2.20}$$

Besides $\Pi_k(T)$ we are also interested in

$$\Lambda(T) \equiv E\left\{ \max_{0\leqq t\leqq T} N(t) \,\middle|\, \text{equilibrium at } t=0 \right\} = \sum_{k=1}^{\infty} k[\Pi_{k+1}(T) - \Pi_k(T)] = \sum_{k=1}^{\infty} [1-\Pi_k(T)], \tag{2.21}$$

where we have used (2.2) and the fact that $\lim_{k\to\infty} \Pi_k(T) = 1$.

3. Analysis of $U_1(t)$. We now investigate the single-bucket model of a dynamic dictionary considered by Van Wyk and Vitter [1]. This differs from the M/M/∞ queue in that customers whose service has been completed cannot leave until the next customer arrives. We let $N(t)$ and $W(t)$ denote the number of customers at time t which are in service, and waiting to exit (having been served), respectively. It is evident that $N(t)$ is the number of customers in the corresponding M/M/∞ queue. The equilibrium probability $p_{m,n}$ that $N(t) = m$ and $W(t) = n$ satisfies the recurrence relation [1]

$$(a+m)p_{m,n} = (m+1)p_{m+1,n-1} + a\delta_{n0} \sum_{r=0}^{\infty} p_{m-1,r}, \tag{3.1}$$

where $p_{m,n} = 0$ if $\min(m, n) < 0$. We have set $\beta = 1/\mu$ in [1], and $\lambda = a\mu$, as before.

If we set $n = 0$ in (3.1), we obtain

$$(a+m)p_{m,0} = a \sum_{r=0}^{\infty} p_{m-1,r} = ap_{m-1}. \tag{3.2}$$

Hence $p_{0,0} = 0$ and, from (2.1),

$$p_{m,0} = \frac{a^m e^{-a}}{(m-1)!(a+m)}, \qquad m = 1, 2, \cdots. \tag{3.3}$$

For $n \geqq 1$, (3.1) implies that

$$(a+m)p_{m,n} = (m+1)p_{m+1,n-1}. \tag{3.4}$$

It follows by induction on n, from (3.3) and (3.4), that

$$p_{m,n} = \frac{(m+n)a^{m+n}\Gamma(a+m)}{m!\Gamma(a+m+n+1)}\, e^{-a}. \tag{3.5}$$

Note that

$$\begin{aligned} \sum_{n=0}^{\infty} p_{m,n} &= \frac{a^m}{m!}\, e^{-a}\Gamma(a+m) \sum_{n=0}^{\infty} \left[\frac{a^n}{\Gamma(a+m+n)} - \frac{a^{n+1}}{\Gamma(a+m+n+1)}\right] \\ &= \frac{a^m}{m!}\, e^{-a} = p_m, \end{aligned} \tag{3.6}$$

as it should.

We are interested in the distribution of $\max_{0 \leqq t \leqq T} [N(t)+W(t)]$, given that the system is in equilibrium at $t=0$, namely

$$\begin{aligned} P_k(T) &\equiv \Pr\left\{ \max_{0 \leqq t \leqq T} [N(t)+W(t)] < k \,\middle|\, \text{equilibrium at } t=0 \right\} \\ &= \sum_{m+n<k} p_{m,n} \Pr\left\{ \max_{0 \leqq t \leqq T} [N(t)+W(t)] < k \,\middle|\, N(0)=m,\ W(0)=n \right\}. \end{aligned} \tag{3.7}$$

But $N(t)+W(t)$ remains constant between arrivals, and $W(t)=0$ immediately after an arrival. Hence, $N(t)+W(t)$ increases only when $N(t)$ does, and

$$\begin{aligned} P_k(T) &= \sum_{m+n<k} p_{m,n} \Pr\left\{ \max_{0 \leqq t \leqq T} N(t) < k \,\middle|\, N(0)=m \right\} \\ &= \sum_{m=0}^{k-1} \sum_{n=0}^{k-1-m} p_{m,n} \int_T^{\infty} s_{m,k}(t)\, dt. \end{aligned} \tag{3.8}$$

Now, from (3.5),

$$\begin{aligned} \sum_{n=0}^{k-1-m} p_{m,n} &= \frac{a^m}{m!}\, e^{-a}\Gamma(a+m) \sum_{n=0}^{k-1-m} \left[\frac{a^n}{\Gamma(a+m+n)} - \frac{a^{n+1}}{\Gamma(a+m+n+1)}\right] \\ &= \frac{e^{-a}}{m!}\left[a^m - a^k \frac{\Gamma(a+m)}{\Gamma(a+k)}\right]. \end{aligned} \tag{3.9}$$

Hence, from (2.14), (3.8) and (3.9), we obtain

$$P_k(T) = \frac{a^k}{k!}\, e^{-a} \sum_{m=0}^{k-1} \left[1 - a^{k-m}\frac{\Gamma(a+m)}{\Gamma(a+k)}\right] \sum_{l=1}^{k} \frac{s_m(-\kappa_l, a)}{\kappa_l s_k'(-\kappa_l, a)}\, e^{-\mu\kappa_l T}. \tag{3.10}$$

Besides $P_k(T)$ we are also interested in

$$\begin{aligned} \Omega(T) &\equiv E\left\{ \max_{0 \leqq t \leqq T} [N(t)+W(t)] \,\middle|\, \text{equilibrium at } t=0 \right\} \\ &= \sum_{k=1}^{\infty} k[P_{k+1}(T) - P_k(T)] = \sum_{k=1}^{\infty} [1 - P_k(T)], \end{aligned} \tag{3.11}$$

where we have used (3.7) and the fact that $\lim_{k\to\infty} P_k(T) = 1$.

The expression in (3.10) should be compared with that in (2.18), which corresponds to the M/M/∞ queue. We note that $P_1(T)=0$. Also, from (3.7), we have

$$P_k(0)=\sum_{m+n<k} p_{m,n}=\sum_{j=0}^{k-1}\sum_{m+n=j} p_{m,n}. \tag{3.12}$$

Now $p_{0,0}=0$ and for $j\geqq 1$, from (3.5), we obtain

$$\begin{aligned}\sum_{m+n=j} p_{m,n} &= \frac{ja^{j-1}e^{-a}}{\Gamma(a+j+1)}\sum_{m=0}^{j}\left[\frac{\Gamma(a+m+1)}{m!}-\frac{m\Gamma(a+m)}{m!}\right] \\ &= \frac{a^{j-1}}{(j-1)!}e^{-a}=p_{j-1},\end{aligned} \tag{3.13}$$

from (2.1). Hence, from (2.2) and (3.12), for $k\geqq 2$,

$$P_k(0)=\sum_{j=1}^{k-1} p_{j-1}=\sum_{j=0}^{k-2} p_j=\Pi_{k-1}(0). \tag{3.14}$$

4. Application. In this section we show explicitly the queue parameters that were implicit in the last two sections. That is, we write $N(t,\lambda,\mu)$ to denote $N(t)$ of § 2 and $W(t,\lambda,\mu)$ to denote $W(t)$ of § 3. We also define the following:

$$\Pi_k(T,\lambda,\mu)\equiv \Pr\left\{\max_{0\leqq t\leqq T} N(t,\lambda,\mu)<k \,\middle|\, \text{equilibrium at } t=0\right\}, \tag{4.1}$$

$$\Lambda(T,\lambda,\mu)=E\left\{\max_{0\leqq t\leqq T} N(t,\lambda,\mu) \,\middle|\, \text{equilibrium at } t=0\right\}, \tag{4.2}$$

$$P_k(T,\lambda,\mu)\equiv \Pr\left\{\max_{0\leqq t\leqq T}[N(t,\lambda,\mu)+W(t,\lambda,\mu)]<k \,\middle|\, \text{equilibrium at } t=0\right\} \tag{4.3}$$

and

$$\Omega(T,\lambda,\mu)=E\left\{\max_{0\leqq t\leqq T}[N(t,\lambda,\mu)+W(t,\lambda,\mu)] \,\middle|\, \text{equilibrium at } t=0\right\}. \tag{4.4}$$

To apply the results of §§ 2 and 3 to the case $H>1$, we treat the processing in each bucket i as an independent M/M/∞ queue with Poisson arrival rate λ/H and mean service time $1/\mu$; for bucket i, we denote the number of items in service at time t by $N_i(t,\lambda/H,\mu)$ and the number of items waiting to be deleted at time t by $W_i(t,\lambda/H,\mu)$.

Under these assumptions, the space complexity of hashing with lazy deletion using H buckets is

$$U_H(t,\lambda,\mu)\equiv\sum_{i=1}^{H}[N_i(t,\lambda/H,\mu)+W_i(t,\lambda/H,\mu)]. \tag{4.5}$$

Let

$$\Phi_{H,k}(T,\lambda,\mu)\equiv \Pr\left\{\max_{0\leqq t\leqq T} U_H(t,\lambda,\mu)<k \,\middle|\, \text{equilibrium at } t=0\right\}, \tag{4.6}$$

and

$$\Psi_H(T,\lambda,\mu)\equiv E\left\{\max_{0\leqq t\leqq T} U_H(t,\lambda,\mu) \,\middle|\, \text{equilibrium at } t=0\right\}. \tag{4.7}$$

The value of $\Psi_H(T,\lambda,\mu)$ can be bounded from below and from above [1].

Since the operations in each hash bucket are independent, the input to the whole algorithm is the same as that of an M/M/∞ queue with Poisson arrival rate λ and mean service time $1/\mu$. Hashing with lazy deletion using H buckets inserts each item at the same time as hashing with lazy deletion using only one bucket, but the H-bucket algorithm cannot delete any item sooner than the single-bucket algorithm. Therefore U_1 is a lower bound on the space complexity of hashing with lazy deletion:

$$U_1(t, \lambda, \mu) \leqq U_H(t, \lambda, \mu). \tag{4.8}$$

It follows from (3.7), (4.6) and (4.8) that

$$P_k(T, \lambda, \mu) \geqq \Phi_{H,k}(T, \lambda, \mu); \tag{4.9}$$

and (3.11), (4.7) and (4.9) imply that

$$\Omega(T, \lambda, \mu) \leqq \Psi_H(T, \lambda, \mu). \tag{4.10}$$

The maximum space used by the algorithm is no larger than the sum over the buckets of the maximum space used in each:

$$\max_{0 \leqq t \leqq T} U_H(t, \lambda, \mu) \leqq \sum_{i=1}^{H} \max_{0 \leqq t \leqq T} [N_i(t, \lambda/H, \mu) + W_i(t, \lambda/H, \mu)]. \tag{4.11}$$

It follows from (4.4), (4.7) and (4.11) that

$$\Psi_H(T, \lambda, \mu) \leqq H\Omega(T, \lambda/H, \mu). \tag{4.12}$$

To summarize, (4.10) and (4.12) give the following upper and lower bounds on the space complexity of hashing with lazy deletion using H buckets:

$$\Omega(T, \lambda, \mu) \leqq \Psi_H(T, \lambda, \mu) \leqq H\Omega(T, \lambda/H, \mu). \tag{4.13}$$

5. Numerical results. We evaluated numerically formulas (2.18), (2.21), (3.10) and (3.11) for several choices of λ and μ. This required computing $\Pi_k(T)$ and $P_k(T)$ for $k = 0, 1, \cdots$ until they were close enough to one that they made no discernible contribution to $\Lambda(T)$ or $\Omega(T)$. Since the evaluation of $\Pi_k(T)$ or $P_k(T)$ requires finding the zeros of the kth degree polynomial (2.11), we chose λ and μ so that both $1 - \Pi_{20}(T) \ll 1$ and $1 - P_{20}(T) \ll 1$ for all relevant values of T. We used subroutine IMTQL2 [8] to determine the zeros of (2.11) by finding the eigenvalues of the appropriate symmetric tridiagonal matrix.

In Figs. 1 and 2 we have taken $\lambda = 10$, $\mu = 2$. Figure 1 shows $\Pi_k(T, 10, 2)$ and $P_k(T, 10, 2)$ for selected values of T. The curve for $T = 0$ shows that $P_k(0, 10, 2) = \Pi_{k-1}(0, 10, 2)$, illustrating (3.14). The graph shows that the difference between $P_k(T, 10, 2)$ and $\Pi_k(T, 10, 2)$ is already quite small at $T = 1$, and has almost vanished by $T = 10$.

Figure 2 shows $\Lambda(T, 10, 2)$ and $\Omega(T, 10, 2)$. Since the Markov chains studied in §§ 2 and 3 are irreducible, we know that both $\Lambda(T, \lambda, \mu)$ and $\Omega(T, \lambda, \mu)$ grow without bound as $T \to \infty$, for any choice of λ and μ; this figure shows that the growth is slow. As one would expect from Fig. 1, $\Lambda(T, 10, 2)$ and $\Omega(T, 10, 2)$ start out differing by one, but quickly become much closer. The data structure interpretation of this is that hashing with lazy deletion using one bucket has almost optimum space complexity.

Figure 3 shows $H\Omega(T, 10/H, 2)$ for several values of H. The curve for $H = 1$, $\Omega(T, \lambda, \mu)$, is a lower bound on $\Psi_H(T, \lambda, \mu)$ by (4.10). The upper bound given by

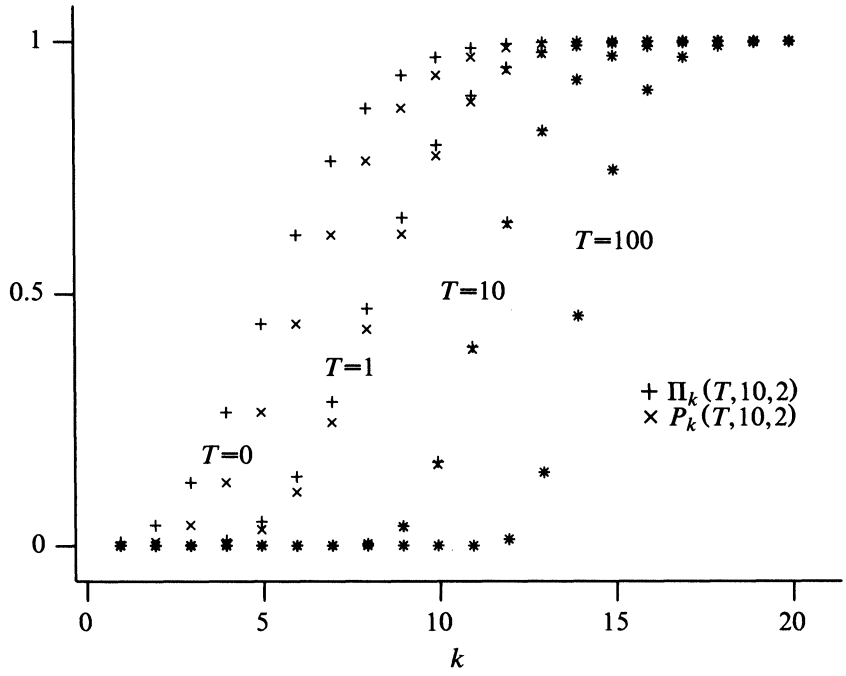

FIG. 1

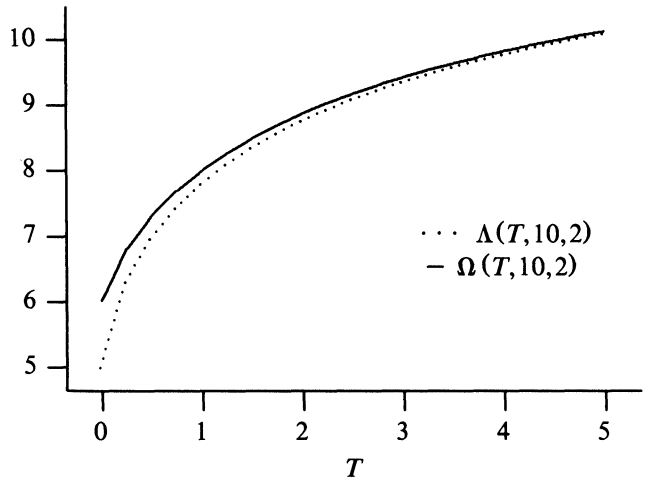

FIG. 2

(4.12) becomes progressively larger as H increases. It appears that $\Psi_H(T, \lambda, \mu)$ exceeds $\Omega(T, \lambda, \mu)$ by an amount that is at most $O(H)$. Since $\Omega(T, \lambda, \mu)$ is only a little larger than $\Lambda(T, \lambda, \mu)$, this suggests that $\Psi_H(T, \lambda, \mu) - \Lambda(T, \lambda, \mu) = O(H)$, which would imply that H can be chosen to be $O(\lambda/\mu)$ in order to give constant expected search time and almost optimum space complexity.

Observations about $\Lambda(T)$, $\Omega(T)$ and $\Psi_H(T)$ based on Figs. 1-3 can be applied to the case of constant λ/μ by an appropriate choice of scaling factors. Other models are appropriate for some applications. For example, to model the analysis of integrated

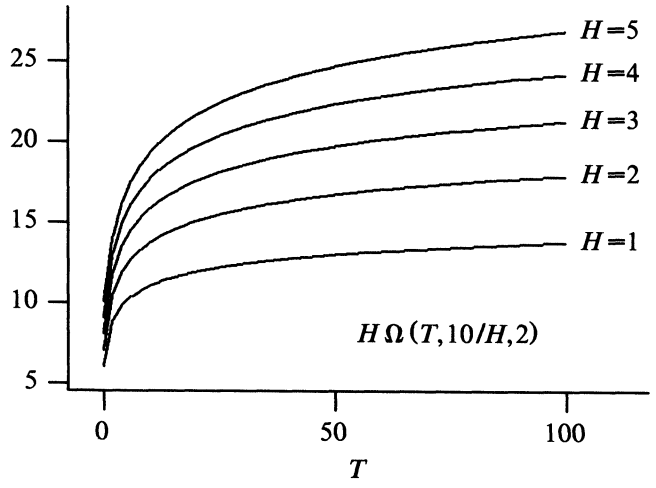

FIG. 3

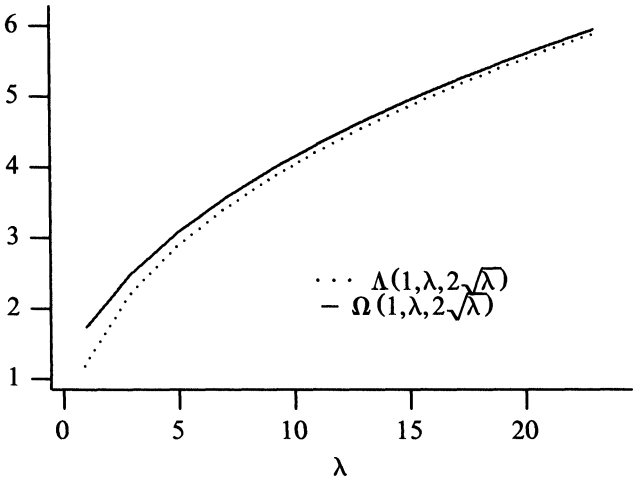

FIG. 4

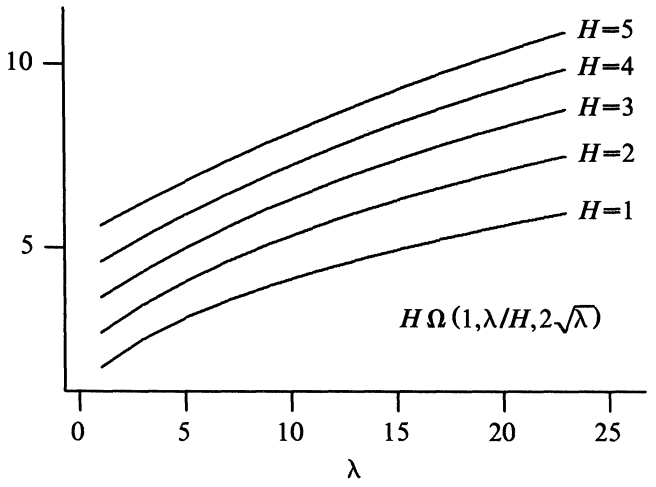

FIG. 5

circuits discussed in § 1, we take λ to be the chip size, $\mu = c\sqrt{\lambda}$, and study $\Lambda(1)$ and $\Omega(1)$ as λ increases [1]. Figures 4 and 5 show results for $c = 2$.

Acknowledgment. Thanks to Jeff Vitter for several thoughtful comments on the manuscript.

REFERENCES

[1] C. J. VAN WYK AND J. S. VITTER, *The complexity of hashing with lazy deletion*, Algorithmica, 1 (1986), pp. 17-29.

[2] T. G. SZYMANSKI AND C. J. VAN WYK, *Space efficient algorithms for VLSI artwork analysis*, Proc. 20th Design Automation Conference, 1983, pp. 734-739.

[3] L. KLEINROCK, *Queueing Systems, Vol.* I: *Theory*, John Wiley, New York, 1975.

[4] D. A. DARLING AND A. J. F. SIEGERT, *The first passage problem for a continuous Markov process*, Ann. Math. Statist., 24 (1953), pp. 624-639.

[5] A. J. F. SIEGERT, *On the first passage time probability problem*, Phys. Rev., 81 (1951), pp. 617-623.

[6] J. KEILSON, *Markov Chain Models—Rarity and Exponentiality*, Applied Mathematical Sciences, Vol. 28, Springer-Verlag, New York, 1979.

[7] J. A. MORRISON, *Analysis of some overflow problems with queueing*, Bell System Tech. J., 59 (1980), pp. 1427-1462.

[8] G. H. GOLUB AND J. H. WELSCH, *Calculation of Gaussian quadrature rules*, Math. Comp., 23 (1969), pp. 221-230.