



On the expected longest length probe sequence for hashing with separate chaining

Pedro Reviriego^{a,*}, Lars Holst^b, Juan Antonio Maestro^a

^a Universidad Antonio de Nebrija, C/ Pirineos, 55, E-28040 Madrid, Spain

^b Royal Institute of Technology, SE-100 44 Stockholm, Sweden

ARTICLE INFO

Article history:

Received 2 November 2010

Received in revised form 14 March 2011

Accepted 4 April 2011

Available online 15 April 2011

Keywords:

Hashing

Table search

Separate chaining

Analysis of algorithms

ABSTRACT

Hashing is a widely used technique for data organization. Hash tables enable a fast search of the stored data and are used in a variety of applications ranging from software to network equipment and computer hardware. One of the main issues associated with hashing are collisions that cause an increase in the search time. A number of alternatives have been proposed to deal with collisions. One of them is separate chaining, in which for each hash value an independent list of the elements that have that value is stored. In this scenario, the worst case search time is given by the maximum length of that list across all hash values. This worst case is often referred to as Longest Length Probe Sequence (*llps*) in the literature. Approximations for the expected longest length probe sequence when the hash table is large have been proposed and an exact analytical solution has also been presented in terms of a set of recurring equations. In this paper, a novel analytical formulation of the expected longest length probe sequence is introduced. The new formulation does not require a recursive computation and can be easily implemented in a symbolic computation tool.

© 2011 Elsevier B.V. All rights reserved.

1. Introduction

Hashing has proven to be a very useful tool for data storage and retrieval in many environments [3]. It is currently used in network equipment to perform route lookup [5], in software applications [1] and in computer hardware [8] among other applications.

The main issue with hashing is collisions whereby two or more data elements are mapped onto the same hash value. A number of schemes have been proposed to deal with collisions [3]. One of them is separate chaining in which for each hash value a linked list is used to store the elements.

When separate chaining is used, the worst case access time is given by the longest length of the linked list across all possible hash values. In theory, if n elements are stored in a table of size m , the worst case would be a list of length n . However, the probability of such event happening is $(1/m)^{n-1}$ which is negligible. Therefore previous works have focused on the expected or average worst case for which approximations when the size of the hash table is large have been presented [2,4].

An exact solution was presented in [7] which makes use of a recurrent equation to compute the expected longest length probe. In this paper an alternative analytical expression for the expected longest length probe is introduced. The new expression is not recurrent and can be easily computed using a symbolic computational tool.

The rest of the paper is organized as follows: in Section 2 the previous work is summarized and the notation used through the paper is introduced; in Section 3 the new analytical formulation is presented; Section 4 compares the results

* Corresponding author.

E-mail address: previrie@nebrija.es (P. Reviriego).

obtained with the new analytical formulation with those obtained in simulation and reported in previous works; finally Section 5 summarizes the conclusions from this work.

2. Previous works

Let us consider a hash table of size m that stores n elements. Let us also assume that the hash function is such that each data element has a probability of $1/m$ of having a given hash value. When multiple elements are stored in the same hash value they are linked in a list such that in order to access the last element, the complete list has to be visited. In this case the longest length probe sequence is given by the length of the longest list across all hash values.

Following [7] let us define $P_b(n, m)$ as the probability that no hash value contains more than b elements, which can be computed using the following recurrence relation (see [7]):

$$P_b(n + 1, m) = P_b(n, m) - \binom{n}{b} \cdot P_b(n - b, m - 1) \cdot \frac{(m - 1)^{n-b}}{m^n}. \tag{1}$$

From $P_b(n, m)$ the expected length of the longest probe sequence ($llps$) can be written as:

$$E[llps] = 1 + \sum_{k=1}^n (1 - P_k(n, m)). \tag{2}$$

When m is large several approximations have been considered [2,4,6]. For example in [2] the following expression for the asymptotic behavior of $E[llps]$ was proposed

$$E[llps] \cong \Gamma^{-1}(m) \cdot \left(1 + O\left(\frac{1}{\ln(\Gamma^{-1}(m))}\right) \right) \tag{3}$$

which shows that the growth of the longest length probe sequence is small with the table size m .

3. Proposed formulation

In this section an alternative formulation for the expected value of the longest length probe sequence is presented. For the derivation each hash value is considered independently and items are supposed to arrive at each value following a Poisson process with rate λ . Then the random variable T_b is defined as the time to have more than b items in one of the hash values. Additionally the random variables W_1, \dots, W_m are defined as the time to have more than b items in the hash value $1, \dots, m$.

Then as arrivals follow a Poisson process we have:

$$P(W_k > t) = \sum_{j=0}^b \frac{(\lambda \cdot t)^j}{j!} \cdot e^{-\lambda \cdot t}. \tag{4}$$

From this equation and the previous definitions it is obvious that

$$P(T_b > t) = P(\min(W_1, \dots, W_m) > t) = P(W_1 > t) \cdot \dots \cdot P(W_m > t) = \left(\sum_{j=0}^b \frac{(\lambda \cdot t)^j}{j!} \cdot e^{-\lambda \cdot t} \right)^m. \tag{5}$$

As the arrivals are assumed to be Poisson, the times between the arrival of two consecutive items are exponentially distributed. Let us denote by Z_k an exponential random variable with mean one. Then the interarrival times between items for the entire hash table are distributed by $\frac{Z_k}{\lambda \cdot m}$. Let us now define the random variable N_b as the number of events to have one hash value with more than b items. Then the random variable T_b can be expressed in terms of Z_k and N_b as follows:

$$T_b = \frac{\sum_{k=1}^{N_b} Z_k}{\lambda \cdot m}. \tag{6}$$

From (6) the following equation can be obtained by multiplying both sides by s and taking the exponential:

$$e^{-s \cdot m \cdot T_b} = e^{-s \cdot (\sum_{k=1}^{N_b} Z_k / \lambda)}. \tag{7}$$

Now taking the expectation on both sides of (7)

$$E(e^{-s \cdot m \cdot T_b}) = E(e^{-s \cdot (\sum_{k=1}^{N_b} Z_k / \lambda)}). \tag{8}$$

The right side of (8) depends on random variables Z_k and N_b , which are independent and can be solved first for the random variables Z_k . To do so, let us now consider an exponential random variable with mean one, Z . Then

$$E(e^{-s \cdot t}) = \int_0^\infty e^{-s \cdot t} \cdot P(Z = t) \cdot dt = \int_0^\infty e^{-s \cdot t} \cdot e^{-t} \cdot dt = \frac{1}{1 + s}. \tag{9}$$

Now using (9) the right side of (8) can be expressed as

$$E(e^{-s \cdot (\sum_{k=1}^{N_b} Z_k / \lambda)}) = E\left(\left(\frac{1}{1 + s/\lambda}\right)^{N_b}\right). \tag{10}$$

And for the left size of (8), taking into account that

$$\frac{\partial(e^{-s \cdot m \cdot t} \cdot P(T_b > t))}{\partial t} = -s \cdot m \cdot e^{-s \cdot m \cdot t} \cdot P(T_b > t) + e^{-s \cdot m \cdot t} \cdot \frac{\partial P(T_b > t)}{\partial t} \tag{11}$$

the following relation can be obtained:

$$\begin{aligned} E(e^{-s \cdot m \cdot T_b}) &= \int_0^\infty e^{-s \cdot m \cdot t} \cdot \frac{\partial P(T_b < t)}{\partial t} \cdot dt = \int_0^\infty e^{-s \cdot m \cdot t} \cdot -\frac{\partial P(T_b > t)}{\partial t} \cdot dt \\ &= 1 - \int_0^\infty s \cdot m \cdot e^{-s \cdot m \cdot t} \cdot P(T_b > t) \cdot dt. \end{aligned} \tag{12}$$

Which using (5) can be reduced to

$$E(e^{-s \cdot m \cdot T_b}) = 1 - \int_0^\infty s \cdot m \cdot e^{-s \cdot m \cdot t} \cdot \left(\sum_{j=0}^b \frac{(\lambda \cdot t)^j}{j!} \cdot e^{-\lambda \cdot t}\right)^m \cdot dt. \tag{13}$$

Therefore

$$E\left(\left(\frac{1}{1 + s/\lambda}\right)^{N_b}\right) = 1 - \int_0^\infty s \cdot m \cdot e^{-s \cdot m \cdot t} \cdot \left(\sum_{j=0}^b \frac{(\lambda \cdot t)^j}{j!} \cdot e^{-\lambda \cdot t}\right)^m \cdot dt. \tag{14}$$

And making $u = \lambda m t$

$$E\left(\left(\frac{1}{1 + s/\lambda}\right)^{N_b}\right) = 1 - \frac{1}{\lambda} \cdot \int_0^\infty s \cdot e^{-(1 + \frac{s}{\lambda})u} \cdot \left(\sum_{j=0}^b \frac{(u/m)^j}{j!}\right)^m \cdot du. \tag{15}$$

Finally changing to $z = 1/(1 + s/\lambda)$ and $v = u/z$

$$E(z^{N_b}) = 1 - (1 - z) \cdot \int_0^\infty e^{-v} \cdot \left(\sum_{j=0}^b \frac{(z/m)^j \cdot v^j}{j!}\right)^m \cdot dv. \tag{16}$$

But

$$E(z^{N_b}) = \sum_{k=1}^{N_b} z^k \cdot P(N_b = k). \tag{17}$$

Which is the generating function of N_b from which its probability density function is extracted.

To simplify the numerical evaluation of (16) the polynomial in the right side can be expanded in terms of the coefficients for each term on $\frac{z \cdot v}{m}$. To that end the coefficients c_i are defined by the following equation:

$$\left(\sum_{j=0}^b \frac{(z/m)^j \cdot v^j}{j!}\right)^m = \sum_{i=0}^{b \cdot m} c_i \cdot \left(\frac{z \cdot v}{m}\right)^i. \tag{18}$$

And then (16) can be written as

$$E(z^{N_b}) = 1 - (1 - z) \cdot \sum_{i=0}^{b \cdot m} c_i \cdot i! \cdot \left(\frac{z}{m}\right)^i. \tag{19}$$

Table 1
Comparison of the expected *llps* for $m = 24$.

n	12	24	48
$E[llps]$ from [2]	2.2360	3.3347	5.1688
$E[llps]$ from [7]	2.2637	3.3515	5.1755
$E[llps]$	2.2636	3.3515	5.1755

Table 2
Comparison of the expected *llps* analytical vs simulation.

m	$n = m/2$	$n = m$	$n = 2m$
32	2.3924	3.5334	5.4148
	(2.3924)	(3.5334)	(5.4143)
64	2.7218	3.9577	5.9673
	(2.7220)	(3.9575)	(5.9678)
128	3.0784	4.3787	6.4926
	(3.0784)	(4.3786)	(6.4930)
256	3.3852	4.7666	6.9926
	(3.3852)	(4.7665)	(6.9930)
512	3.6836	5.1586	7.4760
	(3.6836)	(5.1586)	(7.4761)

And finally

$$P(N_b = k) = c_k \cdot \left(\frac{k!}{m^k}\right) - c_{k-1} \cdot \left(\frac{(k-1)!}{m^{k-1}}\right). \tag{20}$$

Which can be easily computed using a symbolic tool and then $P_b(n, m)$ can be computed as

$$P_b(n, m) = 1 - \sum_{k=b+1}^n P(N_b = k). \tag{21}$$

Now using (2)

$$E[llps] = 1 + \sum_{k=1}^n (1 - P_k(n, m)) = 1 + \sum_{k=1}^n \sum_{i=k+1}^n P(N_k = i). \tag{22}$$

Therefore from the probability density functions of the random variables N_b it is straightforward to compute $E[llps]$. As mentioned before, the probability density functions are obtained through the generating functions given by (16). The computation of (16) can be easily done in a symbolic computation tool using (20).

The proposed analytical formulation does not require a recursive computation as in the original solution presented in [7].

4. Numerical examples

In the previous section an alternative analytical formulation to calculate $E[llps]$ was presented. In this section a few numerical examples are given to compare the results with those obtained in previous works.

In [7] some results are presented in Table 1 for $m = 24$ and three values of n . The results are compared with those from the approximation presented in [2]. Using the same parameters, the new formulation has been applied and the results are shown in Table 1. It can be seen that the results obtained almost exactly match those presented in [7].

In Table 2 additional results for other sets of parameters are presented. In this case the results have been checked by simulation. A script was written to randomly insert n items in a hash table and measure the *llps*. The script is run 10 millions times to get an accurate estimate of the expected value of the *llps*. The simulation results are given between parenthesis. It can be observed that the simulation results match the analytical model almost perfectly.

Finally, from the analytical model additional information can be easily extracted. For example, the distribution of the *llps* is given by

$$P(llps = b) = \sum_{k=b+1}^n P(N_b = k) - \sum_{k=b+1}^n P(N_{b+1} = k). \tag{23}$$

This distribution can be used to determine for a given value b the probability that the *llps* exceeds it. As an example, in Figs. 1–4 the distribution is shown for different values of m and n . It can be observed that the distribution tends to zero quickly once the expected *llps* is exceeded.

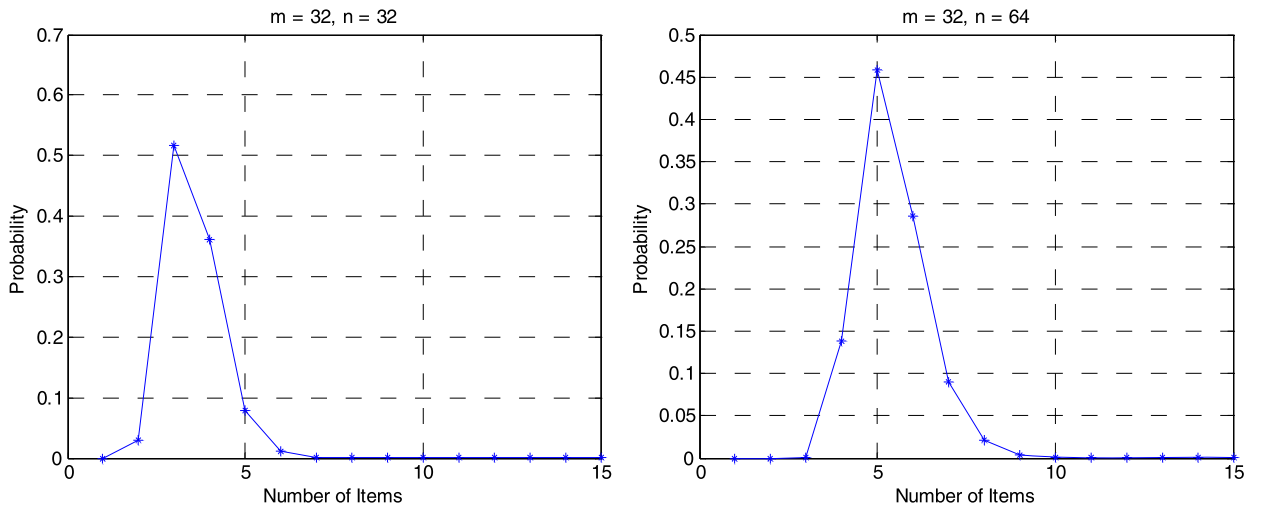


Fig. 1. Distribution of the $llps$ for $m = 32$.

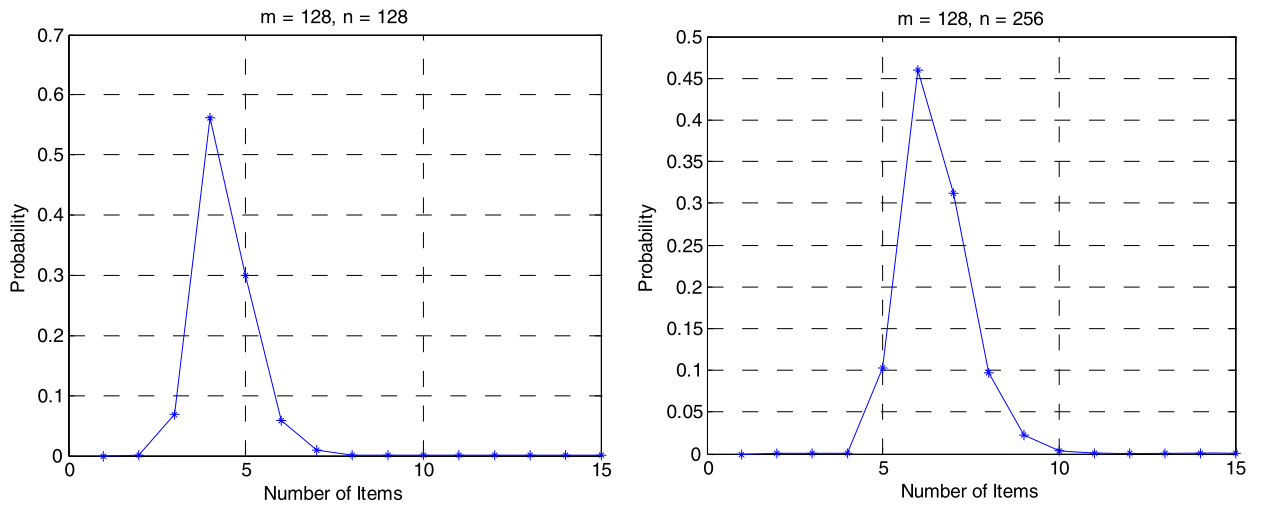


Fig. 2. Distribution of the $llps$ for $m = 128$.

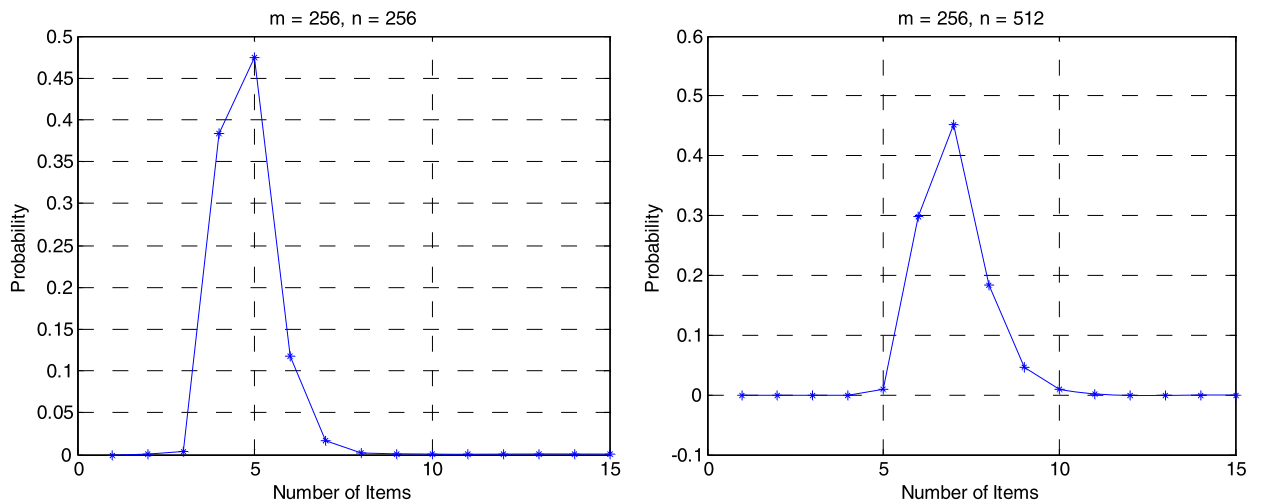


Fig. 3. Distribution of the $llps$ for $m = 256$.

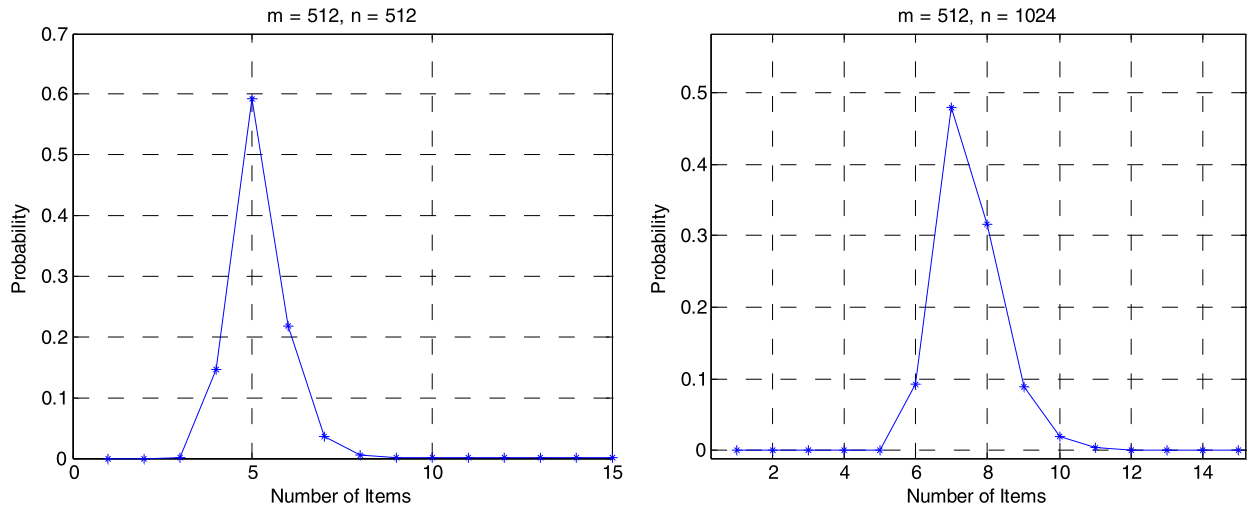


Fig. 4. Distribution of the *llps* for $m = 512$.

5. Conclusions

A new analytical solution for the expected longest length probe sequence in hashing with separate chaining has been presented. The new solution can be easily evaluated using a symbolic computation tool and does not need recursive calculations. Therefore, it can be useful to enable fast evaluation of hashing performance when the hash table size is small and approximations may not be accurate.

The analytical solution has then been used to study the distribution of the *llps* showing that their values are concentrated around the average value. This is an interesting result as it shows that the average value will in most cases be close to the observed *llps*.

References

- [1] E. Fox, L. Heath, Q. Chen, A. Daoud, Practical minimal perfect hash functions for large databases, *Communications of the ACM* 35 (1) (1992) 105–121.
- [2] G.H. Gonnet, Expected length of the longest probe sequence in hash code searching, *Journal of the ACM (JACM)* 28 (2) (1981) 289–304.
- [3] D.E. Knuth, *Art of Computer Programming*, vol. 3: Sorting and Searching, second ed., Addison-Wesley Professional, 1998.
- [4] P.A. Larson, Expected worst-case performance of hash files, *The Computer Journal* 25 (3) (1982) 347–352.
- [5] H. Lim, J.-H. Seo, Y.J. Jung, High speed IP address lookup architecture using hashing, *IEEE Communications Letters* 7 (10) (2003) 502–504.
- [6] M.V. Ramakrishna, Hashing practice: analysis of hashing and universal hashing, *ACM SIGMOD Record* 7 (3) (1988) 191–199.
- [7] M.V. Ramakrishna, An exact probability model for finite hash tables, in: *Proceedings of the Fourth International Conference on Data Engineering*, 1988, pp. 362–368.
- [8] H. Vandierendonck, K.D. Bosschere, XOR-based hash functions, *IEEE Transactions on Computers* 54 (7) (2005) 800–812.