

Dynamical Systems Theory and Hashing

Chaouki T. Abdallah[†] Gregory L. Heileman[†] Bernard M. E. Moret[‡] Bradley J. Smith[†]
chaouki@ecece.unm.edu heileman@ecece.unm.edu moret@cs.unm.edu bsmith@ecece.unm.edu

Department of Electrical and Computer Engineering[†]
Department of Computer Science[‡]
University of New Mexico
Albuquerque, NM 87131

Summary

Number theory and statistics have been used to justify the use of various probing strategies for hashing with open addressing. We show how measures from nonlinear dynamical systems theory can be used to analyze the behavior of different probing strategies. The usefulness of this approach is demonstrated on a number of widely used hash functions. An important connection we make between hashing and nonlinear dynamics is that many of the properties of a good probe sequence are precisely those that define a chaotic dynamical system. As a specific example, we show that double hashing can be described as chaotic discrete-time dynamical systems and use nonlinear dynamical systems theory to quantify how well double hashing can be expected to perform.

The dynamical systems approach to the analysis of hash functions has also proven useful as a design tool. We derive a new family of hash functions, called exponential hashing, from this perspective and show that it improves on conventional double hashing. Theoretical results are supported by experimentation which demonstrates that, for uniform data distributions, exponential hashing matches the performance of double hashing, but that, for nonuniform data distributions (the more likely to be encountered in practice), exponential hashing suffers none of the performance peaks and valleys that affect double hashing.

Finally, we briefly describe how this same approach could be used to derive random number generators with very fast mixing properties.

1 Introduction

We consider the analysis of hash functions that are used to implement dynamic dictionaries. In a *dynamic dictionary* we are required to maintain a set of data elements S (where each $x \in S$ is indexed by a key k_x drawn from a totally ordered universe U) that can be accessed according to the following operations:

- *Find*(k, S). Returns the $x \in S$ such that $k_x = k$, if such an element exists.
- *Insert*(x, S). Adds element x to S .
- *Delete*(k, S). Removes the element $x \in S$ that satisfies $k_x = k$, if such an element exists.

When implementing the above operations using a hash table of size m , a table index is computed from the key value using a hash function h , that performs the mapping $h : U \rightarrow \mathbb{Z}_m$, where \mathbb{Z}_m denotes the integers modulo m .

A *collision* is said to occur if given two keys k_i and k_j , with $i \neq j$, $h(k_i) = h(k_j)$. One method of resolving collisions, termed *open addressing* by Peterson [7], involves computing a sequence of hash slots that are successively probed until an empty hash table slot is found in the case of an *Insert*, or the desired item is found in the case of a *Find* or *Delete*. In open addressing, a hash function is modified so that it uses both a key, as well as a probe number when computing a hash value. Thus, hashing with open addressing uses the mapping $h : U \times \mathbb{Z} \rightarrow \mathbb{Z}_m$ and produces the *probe sequence* $\langle h_0(k), h_1(k), h_2(k), \dots \rangle$.

In this paper we express hash functions with open addressing as discrete-time dynamical systems and demonstrate that tools from nonlinear dynamical systems theory can be used to analyze the capabilities of these hash functions.

Ideally, such hashing strategies are not biased towards any particular probe sequence, and each probe sequence examines hash table locations in an essentially random fashion. This is also characteristic feature of chaotic iterators. Thus, we begin in Section 2 by discussing the characteristics of chaotic iterators, as well as an important property of these systems that will be used in subsequent sections. In Section 3 we introduce and subsequently analyze a number of important collision resolution strategies used in open addressing. We demonstrate how each of these strategies can be expressed as iterators, and then analyze them from the point of view of nonlinear dynamical systems. Specifically, the Lyapunov exponent of each strategy is computed, allowing us make some judgments about how well each can be expected to perform.

We should also mention that the idea of non-expansive hashing was recently introduced by Linial and Sasson [5]. With these hash functions, similar keys are stored in table locations that are close to one another. Furthermore, Indyk et. al [3] have derived a family of non-expansive hashing for multidimensional spaces. This is actually the antithesis of what we are trying to measure—chaotic behavior—however, a dynamical systems analysis of non-expansive hashing may still make sense. We suspect that such an analysis would seek to discover contractive behavior in the underlying system.

2 Preliminaries

Our goal is to express important families of hash functions as dynamical systems and use tools from nonlinear dynamical systems theory to study their performance. The general form for the dynamical systems we consider is given by the first order recurrence relation

$$x_{i+1} = f(x_i) \quad x_0 = c \tag{1}$$

where the constant c is the initial condition, and $f : \mathbb{R} \rightarrow \mathbb{R}$. We will refer to this system as an *iterator*. The *orbit* of x_0 under f is the set of points $\{x_0, x_1 = f(x_0), x_2 = f(x_1), \dots\}$, and in general, we will denote the i th iterate of the point x_0 as $f_i(x_0)$. A point x_0 in the range of f is called a *period- i fixed point* if $f_i(x_0) = x_0$, and the orbit itself is said to be *periodic* with period i . We will refer to a period-1 fixed point as simply a *fixed point*.

The notion of the stability of fixed points is important to our study of open address hash function behavior. Qualitatively, a point x in the range of f is unstable with respect to a nearby fixed point p if the orbits that result from applying f to both p and x move apart from one another. More precisely, let us define the ϵ -neighborhood $N_\epsilon(p)$ of a fixed point p in the range of f as the interval

$$N_\epsilon(p) = \{x \in \mathbb{R} : |x - p| < \epsilon\}.$$

If there exists an $\epsilon > 0$ such that for all $x \in N_\epsilon(p)$, $\lim_{i \rightarrow \infty} f_i(x) = p$, then p is a stable fixed point called a *sink*; however, if each $x \in N_\epsilon(p)$, except p , eventually map outside of $N_\epsilon(p)$ under f , then

p is an unstable fixed point called a *source*. If f is a smooth map on \mathbb{R} , it can be shown that a fixed point p is a sink if $|f'(p)| < 1$, and a source if $|f'(p)| > 1$. Likewise, consider the periodic orbit $\{p_0, p_1, \dots, p_{k-1}, p_0\}$, and define

$$A = |f'(p_0) \cdots f'(p_{k-1})|. \quad (2)$$

If $A < 1$, then the period orbit is a sink. This means that the distance between the orbit of p_0 and the orbit of any other nearby point $x \in N_\epsilon(p_0)$ (both under f) draws closer by a factor of A after k iterations. Similarly, if $A > 1$, the orbit of p_0 is a source, and the distance between this orbit and the orbit of any other nearby point $x \in N_\epsilon(p_0)$ separates by a factor of A after k iterations.

It is well-known that for some choices of even simple f , a system that exhibits extremely complex behavior can be obtained. Such behavior is often referred to as *chaos*. While a universally accepted definition of chaos does not exist, it is generally agreed that chaotic systems exhibit sensitive dependence on initial conditions, coupled with bounded behavior [1, 2, 6]. Qualitatively, an iterator is said to be sensitive to initial conditions if the orbits that result from two initial conditions, which are arbitrarily close, are distinctly different. The Lyapunov number $L(p_0)$ of a general orbit (not necessarily periodic) $\{p_0, p_1, \dots, p_n\}$ is given by

$$L(p_0) = \lim_{n \rightarrow \infty} (|f'(p_0) \cdots f'(p_n)|)^{1/n},$$

and the Lyapunov exponent of this orbit is defined as

$$\lambda(p_0) = \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n \ln |f'(p_{i-1})|. \quad (3)$$

The Lyapunov exponent quantifies the mean exponential rate of divergence or contraction between two nearby orbits. If a positive value is obtained then the distance between the two orbits will rapidly cover the whole set of admissible values of system (1). Furthermore, a Lyapunov exponent of 0 indicates that the system is periodic, and a negative Lyapunov exponent indicates convergence to an equilibrium point.

Because the derivative in equation (3) is often difficult to calculate, the Lyapunov exponent for discrete maps is approximated as

$$\lambda(p_0) = \lim_{n \rightarrow \infty} \left\{ \lim_{E_0 \rightarrow 0} \left(\frac{1}{n} \sum_{i=1}^n \ln \left| \frac{E_i}{E_{i-1}} \right| \right) \right\}. \quad (4)$$

where $E_i = f_i(p_0) - f_i(p_0 + \epsilon)$. Equation (4) is numerically calculated using

$$\lambda(p_0) = \lim_{n \rightarrow \infty} \left\{ \lim_{E_0 \rightarrow 0} \left(\frac{1}{n} \sum_{i=1}^n \ln \left| \frac{\tilde{E}_i}{\epsilon} \right| \right) \right\}. \quad (5)$$

where $\tilde{E}_i = f(p_i + \epsilon) - f(p_i)$.

3 Open Addressing

In Sections 3.1 and 3.2 we briefly review a number of important open addressing schemes that we analyze using nonlinear dynamical systems theory. Each of these strategies is discussed in more detail in [4]. In Section 3.3, a new family of exponential hash functions is derived, and subsequently analyzed from the dynamical systems perspective.

3.1 Linear and Quadratic Probing

Given any hash function $h : U \rightarrow \mathbb{Z}$, an open address hash function that uses linear probing is easily constructed using

$$h_i(k) = (h(k) + c_1 i) \bmod m \quad (6)$$

where $i = 0, 1, \dots$ is the probe number. Thus the argument of the modulus operator is linearly dependent on the probe number. In general, c_1 needs to be chosen so that it is relatively prime to m if all slots in the hash table are to be examined by the probe sequence.¹

In order to apply equation (5) to the analysis of linear probing, equation (6) must be rewritten as a recurrence relation so that it has explicit dependence on previous values of the iteration sequence. Since we have

$$h_{i+1}(k) = (h(k) + c_1 i + c_1) \bmod m,$$

and using the fact that for $a, b, m \in \mathbb{R}$, we have

$$(a + b) \bmod m = (a \bmod m + b \bmod m) \bmod m \quad (7)$$

we may rewrite equation (6) as the following linear time-invariant first-order iterator

$$\begin{aligned} h_{i+1} &= (h_i + (c_1 \bmod m)) \bmod m \\ h_0 &= h(k) \end{aligned} \quad (8)$$

Note that the dependence on k is specified in the initial condition h_0 .

Quadratic probing is a simple extension of linear probing that makes the probe sequence non-linearly dependent on the probe number. For any hash function h , quadratic probing uses the sequence

$$h_i(k) = (h(k) + c_1 i + c_2 i^2) \bmod m \quad (9)$$

where c_1 and c_2 are positive constants. Once again, the specific constants are critical to the performance of this method.

To obtain a recurrence relation solution to equation (9) we note that

$$h_{i+1}(k) = \left[(h(k) + c_1 i + c_2 i^2) + (c_1 + c_2(2i + 1)) \right] \bmod m,$$

which leads to the time-varying first-order recurrence relation

$$\begin{aligned} h_{i+1} &= (h_i + (c_1 + c_2(2i + 1)) \bmod m) \bmod m \\ h_0 &= h(k). \end{aligned} \quad (10)$$

The Lyapunov exponent for the linear and quadratic iterators, equations (8) and (10), were calculated using equation (5) yielding a value of 0 in both cases independent of the initial condition. This indicates the presence of periodic orbits, which verifies the easily observable characteristics of these hash functions.

¹This is actually a more general form of linear probing than most authors consider. Typically in linear probing it is assumed that $c_1 = 1$, and the probe sequence becomes $\langle h'(k), h'(k) + 1, \dots, m - 1, 0, 1, \dots, h'(k) - 2, h'(k) - 1 \rangle$.

3.2 The Double Hashing Family \mathcal{F}_D

It is well known that linear and quadratic probing strategies suffer from clustering, because both have sequence increments that are independent of the key. Double hashing remedies this problem by introducing a second hash function that is used in the computation of the increment.

Given two hash functions g and h , the family \mathcal{F}_D of double hash functions is given by:

$$h_i(k) = (g(k) + ih(k)) \bmod m. \quad (11)$$

In this family of hash functions, the initial probe $h_0(k) = g(k)$, and successive probes are offset from previous probes by multiples of $h(k)$ modulo m . Thus the probe sequence depends on k through both g and h .

A recurrence relation for the family \mathcal{F}_D described by equation (11) is obtained by first noting that

$$h_{i+1}(k) = [(g(k) + ih(k)) + h(k)] \bmod m,$$

which leads to the iterator

$$\begin{aligned} h_{i+1} &= (h_i + h(k)) \bmod m \\ h_0 &= g(k), \end{aligned} \quad (12)$$

Notice that for a given key k , $h(k) \bmod m$ is constant over a single probe sequence, but may change from one probe sequence to the next. Therefore, a more complete state-space description of this system is given by the following set of coupled first-order difference equations:

$$x_{i+1} = x_i; \quad x_0 = h(k) \quad (13)$$

$$y_{i+1} = (y_i + x_i) \bmod m; \quad y_0 = g(k), \quad (14)$$

where $h_i = y_i$ is the output equation used to iterate over the table space.

Since the dynamical system specified in equations (13) and (14) is two dimensional, we must look at its rate of expansion and/or contraction in two dimensions. In other words, we will have to look at two Lyapunov exponents in order to determine whether the system is exhibiting sensitive dependence on initial conditions. These Lyapunov exponents are found from the eigenvalues of the Jacobian matrix of the mapping [2]. Fortunately, since we know that the first component x_i remains a constant, one can still use equation (5) to calculate the largest Lyapunov exponent in the y direction. The value of this Lyapunov exponent characterizes the performance of the iterator over the table space.

A widely used member of \mathcal{F}_D , proposed by Knuth [4], has

$$g(k) = k \bmod m \quad (15)$$

$$h(k) = k \bmod (m - 2), \quad (16)$$

where both m and $m - 2$ are prime. The largest Lyapunov exponent of this hash function was calculated using equation (5) with $m=109$, yielding a value of 0.69 for any initial condition. Indicating chaos may be present in this iterator. A graph of p_0 versus p_2 for this hash function is given in Figure 1. Notice that graph does not cross the 45 degree line, except at the point of singularity. This demonstrates that this iterator has no period 2 orbits. Similar behavior is obtained for plots of p_0 versus p_i , $i = 1, 2, \dots$ as long as m is prime.

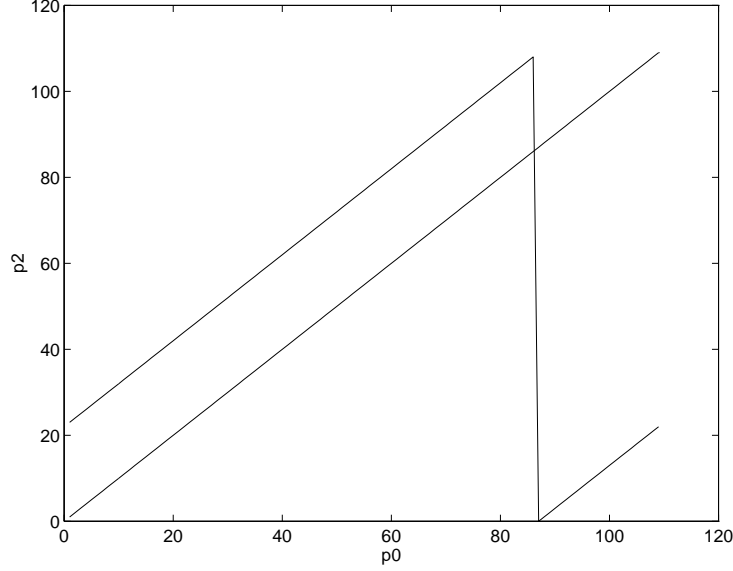


Figure 1: A plot of p_0 versus p_2 for the double hash function in equation (14) with $g(k) = k \bmod m$, $h(k) = k \bmod (m - 2)$, and $m = 109$.

3.3 The Exponential Hashing Family \mathcal{F}_E

A new family of hash functions \mathcal{F}_E , which we call exponential hashing, makes use of two hash functions g and h to compute a probe sequence according to

$$h_i(k) = (g(k) + h(k)^i) \bmod m. \quad (17)$$

The exponentiation does not have to explicitly implemented, but can be computed via successive multiplications during the probing process. For this reason, the number of mathematical operations needed to implement equation (17) is identical to the number needed to implement equation (11). Intuitively, this open addressing scheme may improve on standard double hashing because the sequence increment depends both on the key and on the step number.

To obtain an iterator for the family \mathcal{F}_E described in equation (17), we first write

$$\begin{aligned} h_{i+1} &= (h(k)h(k)^i + g(k)) \bmod m \\ &= \left[\left((h(k)^i + g(k))h(k) \right) \bmod m + ((1 - h(k))g(k)) \bmod m \right] \bmod m. \end{aligned} \quad (18)$$

Letting $n = m/h(k)$, we can rewrite the first term in equation (18) as

$$\begin{aligned} \left((h(k)^i + g(k))h(k) \right) \bmod m &= \left((h(k)^i + g(k))h(k) \right) \bmod (n \cdot h(k)) \\ &= h(k) \left((h(k)^i + g(k)) \bmod n \right) \\ &= h(k) \left\{ \left((h(k)^i + g(k)) \bmod m \right) \bmod n \right\}. \end{aligned}$$

Substituting this final result into equation (18) yields the following iterator for \mathcal{F}_E :

$$h_{i+1} = [h(k)(h_i \bmod n) + (g(k)(1 - h(k))) \bmod m] \bmod m \quad (19)$$

$$h_0 = g(k). \quad (20)$$

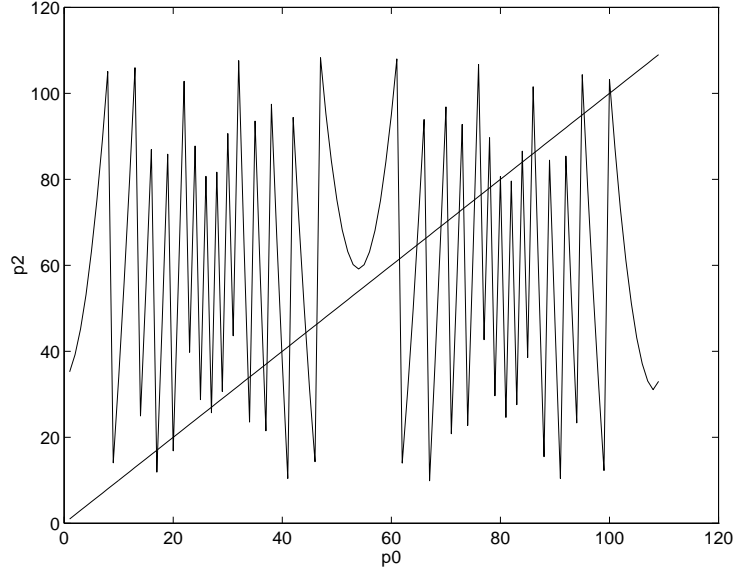


Figure 2: A plot of p_0 versus p_2 for the exponential hash function in equation (19) with $g(k) = k \bmod m$, $h(k) = k \bmod (m - 2)$, and $m = 109$.

A more complete state-space description of this system is given by the following triple of coupled first-order difference equations:

$$x_{i+1} = x_i; \quad x_0 = h(k), \quad (21)$$

$$y_{i+1} = y_i; \quad y_0 = g(k), \quad (22)$$

$$z_{i+1} = [x_i(z_i \bmod n) + (1 - x_i)y_i] \bmod m; \quad z_0 = g(k) \quad (23)$$

where $h_i = z_i$ is the output equation used to iterate over the table space.

Although the dynamical system given in equations (21)–(23) is three dimensional, the first two components remain constant, and equation (5) can once again be used to calculate the largest Lyapunov exponent in the z direction.

A particular member of \mathcal{F}_E , recently introduced by Smith [8], has $g(k) = k \bmod m$, and $h(k) = k \bmod (m - 2)$, where $m = 2p + 1$ is selected so that both m and p are prime. The largest Lyapunov exponent of this hash function was calculated using equation (5) with $m=109$, yielding values between 4 and 6 depending on the initial condition. This indicates higher sensitive dependence than the family of double hash functions. A graph of p_0 versus p_2 for this exponential hash function is given in Figure 2. Notice that the graph does cross the 45 degree line; which indicates the presence of period-2 orbits. However, since the slopes of these lines are greater than 1 in magnitude, these orbits are unstable. Period- i orbits can be seen in plots of p_0 versus p_i , $i = 1, 2, \dots$

Smith et. al [9] have experimentally examined the exponential iterator. Their analysis demonstrates that for uniform data distributions, the performances of double and exponential hashing are nearly identical; while for nonuniform data distributions, exponential hashing suffers none of the performance peaks and valleys that result from using double hashing.

4 Conclusions

The analysis employed here can be applied to any open address hashing strategy that can be expressed as an iterator. Furthermore, we are not aware of any such strategy that is not expressible

as an iterator. More importantly, it is possible to use this approach as a design tool in the construction of new hash functions, or to justify to choice of particular parameters in existing hash functions. We have done some preliminary work in this regard, and have found a number of very simple chaotic iterators that function well as hash functions. One other advantage of looking at hashing from this perspective is that it may allow other results obtained in the area of nonlinear dynamical systems theory to be applied to the design and analysis of hash functions.

Since the analysis base on iterators assumes a real domain, rather than the finite integer domain of a hash function, the results of the analysis can be viewed as characteristic of an idealized hash function. However, we can approach this ideal on very large tables or by the simple expedient of using real numbers in computing the iterator and discretizing the resulting values. Characterizing the effect of this discretization (or the difference between the idealized iterator on the reals and the has function on finite integers) remains an open problem.

Hash functions iterators are closely related to pseudo-random number generators; indeed, congruential generators are readily recognized as a special case of very simple iterators. Among the many combinatorial, number-theoretic, and statistical measures used to characterize pseudo-random number generators, none has proved satisfactory in and of itself; measures derived from nonlinear dynamical systems theory may shed a new light on the quality of such generators.

References

- [1] M. Barnsley. *Fractals Everywhere*. Academic Press, San Diego, CA, 1988.
- [2] R. L. Devaney. *An Introduction to Chaotic Dynamical Systems*. Addison-Wesley, Reading, MA, 2nd edition, 1989.
- [3] P. Indyk, R. Motwani, P. Raghavan, and S. Vempala. Locality-preserving hashing in multidimensional spaces. In *Proceedings of the 29th Annual ACM Symposium on the Theory of Computing*, 1997.
- [4] D. E. Knuth. *Searching and Sorting*, volume 3, *The Art of Computer Programming*. Addison-Wesley, Reading, MA, 1973.
- [5] N. Linial and O. Sasson. Non-expansive hashing. In *Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science*, 1996.
- [6] H.-O. Peitgen, H. Jürgens, and D. Saupe. *Chaos and Fractals: New Frontiers of Science*. Springer-Verlag, New York, 1992.
- [7] W. W. Peterson. Addressing for random access storage. *IBM Journal of Research and Development*, 1(2):130–146, 1957.
- [8] B. J. Smith. *An Exponential Open Address Hash Function Based on Dynamical Systems Theory*. PhD thesis, University of New Mexico, 1997.
- [9] B. J. Smith, G. L. Heileman, and C. T. Abdallah. The exponential hash function. *ACM Journal of Experimental Algorithmics*, 3(1), 1997 (to appear).