Flood Search under the California Split Rule

Y. Baryshnikov[1], E. Coffman[2,*,†], P. Jelenković[2,*], P. Momčilović[2,‡] and D. Rubenstein[2,*]

1. Bell Labs
Lucent Technologies
Murray Hill, NJ 07974
ymb@research.bell-labs.com

2. Department of Electrical Engineering
Columbia University
New York, NY 10027
{egc, predrag, petar, danr}@ee.columbia.edu

November 2002, revised August 2003

Abstract

We consider flood search on a line and show that no algorithm can achieve an average-case competitive ratio of less than 4 when compared to the optimal off-line algorithm. We also demonstrate that the optimal scanning sequences are described by simple recursive relationships that yield surprisingly complex behavior related to Hamiltonian chaos.

Keywords: Flood search, peer-to-peer systems, expanding ring, average case competitive analysis

1 Introduction

Since a major portion of traffic in the Internet is attributed to file swapping, overlay peer-to-peer (P2P) networks are one of the most popular applications on the Web. Since such networks are fully distributed, the efficiency of their operation highly depends on one's ability to locate information in them. Two types of P2P networks have received considerable attention: structured and unstructured. Structured networks form an overlay and configure information in a specific way that allows clients to search efficiently (see e.g., [3, 11, 12]). For example, file `xyz` is always stored in a part of the overlay having files with similar characteristics like the values of a distributed hash function. Thus, when a client searches for `xyz` it only needs to examine a small number of specific "places". Such an approach seems very reasonable when most of the clients belong to the same entity and are willing to cooperate, such as would be the case when a client is required to store `xyz` even if it has no interest in it. However, most of the operational P2P networks (e.g., Gnutella, KaZaA) are formed by clients that have little or no incentive to perform social functions at their own expense. Hence, unstructured P2P networks, the focus of this paper, are likely to remain popular.

Efficient search in unstructured P2P networks is a challenging problem. With no specific knowledge as to where the object might be located, or even whether it exists, the only option that a client has is to query other clients for the object until it is found. Usually, when designing a search algorithm, two parameters need to be balanced: (i) the time it takes to locate an object, assuming it exists, and (ii) the overhead associated with the search, e.g., the total number of query messages. Two methods developed in the literature are the expanding ring algorithm (used in Gnutella) and the random-walkers approach [10]. Flood search, i.e., the expanding ring algorithm, is a natural solution when one would like to complete the search in a short period of time. Briefly, this algorithm can be described as follows. In the first round, the client queries its neighbors, i.e., nodes that are one hop away in the overlay. This is achieved by setting the time-to-live (TTL) field in the query to one. If the object is not located, the client sets a

[*] Supported by NSF Grant ANI-0117738.

[†] Corresponding author

[‡] Supported in part by an IBM PhD Fellowship.

new value of TTL (e.g., TTL=2) and forwards the requests to its neighbors. The neighbors decrement the value of TTL by one and forward the request to their neighbors. This continues until the object has been found or the TTL has been decremented to 0. In the latter case, a new round of flooding with a larger TTL begins; the process continues until the object is found or it is decided that the object is not in the graph. The total overhead is created by duplicate messages within one round of flooding and by the duplication caused by an inappropriate choice of TTL. When searches on trees are considered, no duplicate queries are generated within a round since there are no cycles.

This type of search problem has appeared in other contexts, e.g. see [2, 8, 1, 5, 6, 7]. Indeed, a result similar to our Theorem 1 has used the same proof technique [2], viz., reduction to a second order difference equation. However, our specific problem is new and leads to a different competitive ratio (the competitive setting is formally described in the next section). Moreover, our average-case analysis of the continuous relaxation brings out complex behavior that was not observed before. We note that our problem bears some similarity to the search problem described in [9], where the authors examine the algorithmic issues in congestion (finding the optimal transmission rate based on partial knowledge). Although both in [9] and here, "search" is the key word, the problem setups are different.

2 Model

An infinite, ordered list L of items is to be searched for a given item I whose position in L is not known in advance. Searches are composed of scans, each beginning at the first position and proceeding sequentially to some given position number. The cost of a scan to position x is simply x. *Scan costs must be paid in advance*, so finite search costs require that an algorithm put a limit to each scan, and perform larger and larger scans until I is found. In particular, we consider algorithms that are defined by integer sequences $0 < x_1 < x_2 < \cdots$ and that operate by scanning the first x_1 positions of L, then the first x_2 positions of L, and so on, until I is found; they are called *expanding scan algorithms.* If I is in position k, then the total search cost is defined to be $\sum_{1 \le i \le j} x_i$, where $x_{j-1} < k \le x_j$. Note that the scan of positions 1 through x_j repeats, with no added benefit, the scan of positions 1 through x_{j-1}. Also, the cost of the last scan, say to x_j, is defined to be x_j even though I may have been found before the scan was complete.

Let $\mathcal{E}$ be the expected search cost of the algorithm corresponding to a given sequence $\{x_i\}$ under the distribution $\{p_i\}$ of I's position in L. An off-line algorithm knows I's position in advance, so its expected search cost is simply the expected value $\mathcal{E}_o := \sum_{i \ge 1} i p_i$ which clearly satisfies $\mathcal{E}_o \le \mathcal{E}$. For any given algorithm, i.e., for any fixed sequence $\{x_i\}$, an adversary chooses a distribution $\{p_i\}$ in order to maximize the ratio $\mathcal{E}/\mathcal{E}_o$. We prove that the adversary can always make the worst-case ratio arbitrarily close to 4. We also show that the constant 4 in this bound is optimal in that there exists an algorithm whose worst-case expected search cost is at most $4\mathcal{E}_o$. We do this by verifying that the bound applies to the "California Split" algorithm; in this case, $x_i = 2^{i-1},\ i \ge 1$ (in gambling terms, a gambler doubles the bet at every loss).

In Section 4, we turn our attention to the average case behavior of flood searches. It will be shown that the scanning sequences that minimize the expected search cost are described by simple recursive relationships that yield very complex behavior. More importantly, the examples will reveal that the expected performance of the California Split algorithm is close to optimal.

Next, we briefly outline the two points that relate searches on a line to the corresponding ones on graphs: (i) non-linear overhead and (ii) limits on how many nodes can be visited in each step.

First, with no cycles in the graph, the search cost is linear, i.e., the number of messages is equal to the number of distinct nodes visited. Our calculations indicate that in large random (Erdös-Rényi) graphs the total number of messages, as a function of the number x of distinct nodes visited, behaves roughly as $x + cx^2$, where c is a constant that depends on the parameters of the graph. Hence, potentially one could replace the linear cost of the one-dimensional problem with a quadratic cost to examine search performance in random graphs.

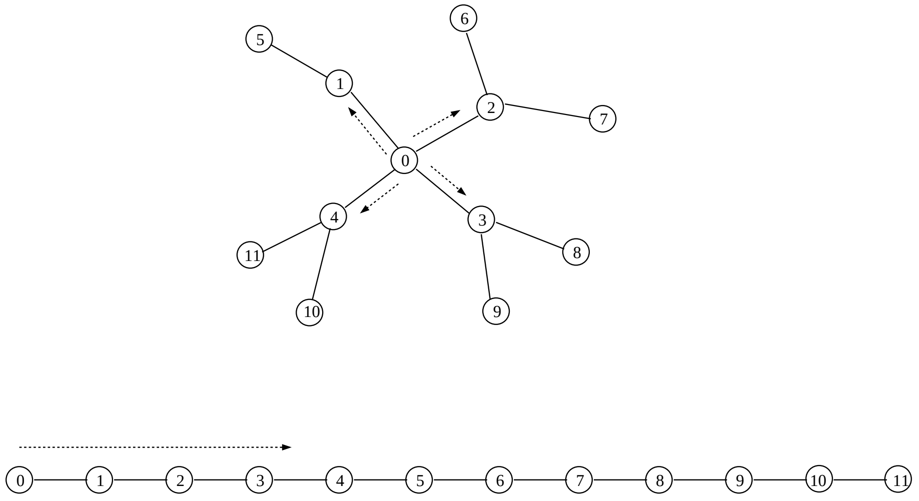

Figure 1: An example of mapping searches on a tree (top) to searches on a line (bottom). Dashed arrows indicate the flood direction. Search is initiated from the node 0. When considering the line, scan points are restricted to 4 and 11.

Second, tree searches can be transformed to searches on a line by mapping the tree into a breadth-first ordering of the nodes (see Fig. 1). However, the limitation of this case is that one can not query an arbitrary number of nodes in a given round. In this regard, consider as an example a search on a binary tree starting from the root. Here, the number x_i of nodes that can be visited in the ith round must be $x_i = 2^{\mathrm{TTL}+1} - 1$. Thus, taking the preceding into account, we can conclude that the case examined in this paper gives in fact the best possible search cost performance over all classes of connected graphs. In other words, flood search on graphs will exhibit worse performance than that obtained for the one-dimensional case.

3 Expanding-Scan Bound

Our main result states that no expanding-scan algorithm has a performance ratio $\mathcal{E}/\mathcal{E}_o$ that is less than 4 under a worst-case choice for the distribution $\{p_i\}$. This bound is achievable since there exists an expanding-scan algorithm whose upper bound is as small as 4 for all distributions $\{p_i\}$ with a finite first moment.

Theorem 1. *The average-case competitive ratio satisfies*

$$\inf_{\{x_i\}} \sup_{\{p_i\}:\, \mathcal{E}_o < \infty} \mathcal{E}/\mathcal{E}_o = 4. \tag{1}$$

Remark It is worth noting that the upper bound, which says that no algorithm achieves a ratio $\mathcal{E}/\mathcal{E}_o$ less than 4, holds for arbitrary cost functions.

Proof. When considering supremums on the performance ratio for fixed $\{x_i\}$, it is useful to transform $\{p_i\}$ into the distribution $\{\hat{p}_i\}$, where $\hat{p}_i = 0$ everywhere except at the positions following the strategy boundaries, where

$$\hat{p}_{x_j+1} = \sum_{i=x_j+1}^{x_{j+1}} p_i.$$

This transformation concentrates all of the probability mass of the items in positions x_j+1 through x_{j+1} into the first item. It is easy to see that this can only decrease the expected cost $\mathcal{E}_o$, and that it has no effect on $\mathcal{E}$, since the search costs of all items in positions x_j+1 through x_{j+1} are the same. Thus, the transformation only increases the ratio $\mathcal{E}/\mathcal{E}_o$.

The tails of $\{p_i\}$ and $\{\hat{p}_j\}$ evaluated at x_j are the same and denoted by $f_j := \sum_{i>x_j} p_i$. The tail of $\{\hat{p}_j\}$ remains constant throughout an interval $[x_j+1, x_{j+1}]$ and thus we derive

$$\begin{aligned} \mathcal{E} &= \sum_{i\geq 0} f_i x_{i+1} && (2) \\ \mathcal{E}_o &\geq f_0 + \sum_{i\geq 1} f_i(x_i - x_{i-1}) && (3) \end{aligned}$$

where we define $x_0 = 0$, which implies $f_0 = 1$.

We first verify that $\sup \mathcal{E}^* \leq 4\mathcal{E}_o$, where $\mathcal{E}^*$ is the expected search cost under the California split rule: $x_i = 2^{i-1}$, $i \geq 1$. Indeed, (2) can be rendered in this case as

$$\mathcal{E}^* = \sum_{i\geq 0} f_i 2^i \qquad (4)$$

and (3) as

$$\mathcal{E}_o^* \geq f_0 + f_1 + \sum_{i\geq 2} f_i 2^{i-2}$$

which together imply $\mathcal{E}^* \leq 4\mathcal{E}_o^* - 3f_0 - 2f_1 < 4\mathcal{E}_o^*$.

It remains to prove that $\sup_{\{f_i\}} \mathcal{E}/\mathcal{E}_o \geq 4$ for any given strategy $\{x_i\}$. To this end, we let an adversary try to find a sequence $f_0 = 1 \geq f_1 \geq f_2 \geq \cdots$ that maximizes $\mathcal{E}/\mathcal{E}_o$. Assume that, for some positive constants $C < 4$ and A, and for some sequence $\{x_i\}$, we have $\mathcal{E} \leq C\mathcal{E}_o + A$, no matter what f_i's are chosen by the adversary. This would imply in particular that this inequality is true for the singular probability measure concentrated at point $x_k + 1$, that is, for the tail-probability sequence $\{f_i(k)\}$ with $f_i(k) := \mathbf{1}\{i \leq k\},\ i \geq 0$. Substituting the $f_i(k)$ into (2) and (3), this implies that, for any $k \in \mathbf{Z}_+$,

$$\sum_{i=1}^{k+1} x_i \leq C\left(1 + \sum_{i=1}^{k}(x_i - x_{i-1})\right) + A = Cx_k + B$$

with B being some positive constant. Introducing $\tilde{y}_k := \sum_{i=0}^{k} x_i$ into this inequality, we obtain the second order difference inequality $\tilde{y}_{k+1} - C\tilde{y}_k + C\tilde{y}_{k-1} \leq B$ for any $k \in \mathbf{Z}_+$. Clearly, the sequence $\tilde{y}$ increases without bound, and so for some $N \geq 0$ one has $\tilde{y}_{N+1} > B$. Define the new, positive and increasing sequence $y_k := \tilde{y}_{N+k} - B$ which apparently satisfies the difference inequality

$$y_{k+1} - Cy_k + Cy_{k-1} \leq 0. \qquad (5)$$

We show that this inequality can not hold if $C < 4$. Accordingly, define a sequence $\ldots, \xi_{-1}, \xi_0 = 0, \xi_1 = 1, \xi_2, \ldots$ satisfying the recurrence $\xi_{\ell-1} - C\xi_\ell + C\xi_{\ell+1} = 0$ (which is adjoint to the corresponding operator on the y_k's). This sequence is uniquely determined by its values $\xi_0 = 0,\ \xi_1 = 1$. Further, $C \geq 1$ must hold ($\mathcal{E} \geq \mathcal{E}_o$), and the roots of the characteristic equation $1 - C\lambda + C\lambda^2 = 0$ are complex conjugates, since the condition $C < 4$ implies $C^2 - 4C < 0$. Thus, the general solution has an oscillatory component yielding, for some $M \geq 1$,

$$\xi_i > 0, \quad 0 < i \leq M, \quad \xi_{M+1} \leq 0 \qquad (6)$$

Also, $\xi_{-1} = C\xi_0 - C\xi_1 = -C < 0$. Next, (5) and (6) directly imply that the following sum is nonpositive

$$\sum_{i=1}^{M}(y_{i+2} - Cy_{i+1} + Cy_i)\xi_i \leq 0. \tag{7}$$

Next, using elementary algebra, this sum can be rendered as

$$\sum_{i=1}^{M+1} y_i(\xi_{i-2} - C\xi_{i-1} + C\xi_i) + [-Cy_{M+1}\xi_{M+1} - y_1\xi_{-1} + y_{M+2}\xi_M].$$

However, the preceding expression can be shown to be positive since the first sum vanishes by the definition of ξ_i and the second term is positive by $y_i > 0$, $i > 0$, $\xi_{M+1} \leq 0$, $\xi_{-1} < 0$ and $\xi_M > 0$. Hence, (7) is contradicted, proving that $C < 4$ is impossible. □

4 Average-Case Performance

In what follows we examine the average-case behavior of the search algorithms. Recall that p_i is the probability of the item being in position i. With the distribution $\{p_i\}$ fixed, we would like to determine the flood strategy that minimizes the expected search cost, as given by (2). The optimal solution in full generality appears to be out of reach, so we resort to a continuous relaxation, i.e., the position of the item and the scanning points take values on the positive reals. The examples we discuss reveal that, even when considering only the continuous relaxation, the solution has an intriguing and extremely complex structure. Throughout this section, we denote by X the position of the object.

First, we consider the easiest case when the item's position is uniformly distributed on $(0, 1]$. It is straightforward to verify that any strategy that scans the whole interval in no more than two steps ($0 < x_1 \leq 1$, $x_2 = 1$) achieves the search-cost ratio $\mathcal{E}/\mathcal{E}_o = 2$:

$$\frac{\mathcal{E}}{\mathcal{E}_o} = \frac{x_1\mathbb{P}[X \leq x_1] + (x_1 + 1)\mathbb{P}[X > x_1]}{1/2} = 2.$$

Furthermore, using (2) it can be shown that strategies employing more than 2 scans result in a competitive ratio greater than 2. Thus an optimal strategy must have at most two scans. Observe that the California Split algorithm achieves the same minimum for the choices $x_1 = 0.5$ and $x_1 = 1$.

Next, we let the position of the item be exponentially distributed with parameter 1, i.e., $\mathbb{P}[X > x] = e^{-x}$. Then, the optimal sequence of x_i's satisfies (by differentiating $\mathcal{E}$)

$$e^{-x_{i-1}} - x_{i+1}e^{-x_i} = 0$$

and therefore

$$x_i = e^{x_{i-1} - x_{i-2}}, \quad i > 2$$

with $x_2 = e^{x_1}$. Hence, the optimal sequence is defined by the choice of x_1. However, not all choices of x_1 lead to increasing sequences. In particular, empirical evidence suggests that the sequence is not increasing for x_1 in the interval (0.20..., 0.74...). Thus, it is of interest to examine the feasible values of x_1. To determine a set of such values, we define $\Delta_n \equiv \Delta_n(x_1) := x_{n+1} - x_n$ for $n \geq 1$. Then a given x_1 generates a monotonic sequence only if $\Delta_n > 0$ for all $n \geq 1$. In Fig. 2 we plot $\Delta_n(x_1)$ for $n = 4, \ldots, 10$; the first three Δ_n are positive for all $x_1 > 0$. As can be seen from the figure, the interval of non-monotonicity appears not to contain isolated points.

The expected search cost assuming an increasing sequence can be represented as

$$\mathcal{E} = 1 + x_1 + \sum_{i=1}^{\infty} e^{-x_i},$$

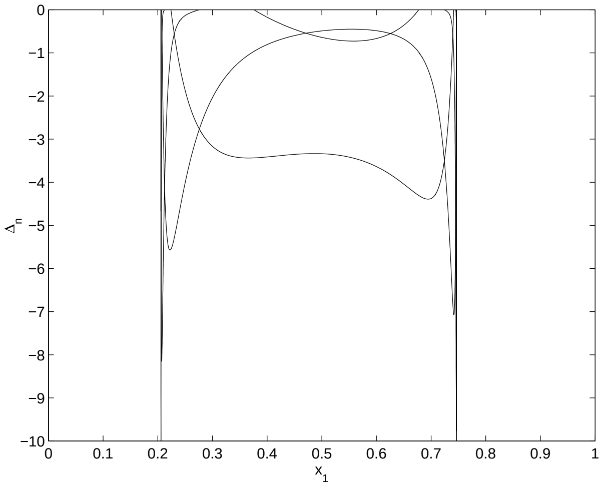

Figure 2: Values of Δ_n for $n = 4, 5...10$ as a function of the initial scanning point x_1. The interval in which at least one $\Delta_n < 0$ has no isolated points. The position of the object is exponentially distributed with mean 1.

which is plotted in Fig. 3 for different values of x_1. We observe that the optimal choice of x_1 is at the boundary of the region in which the sequence is increasing. It is interesting to see that by approaching points 0.20... and 0.74... (see the figure) from the left and right, respectively, the sequence of scans $\{x_i\}$ increases less and less rapidly. In the same figure, we plot with a dashed line the expected cost under the California Split rule ($x_i = 2^{i-1}x_1$). The two algorithms differ by less than 3% for the optimal choices of x_1.

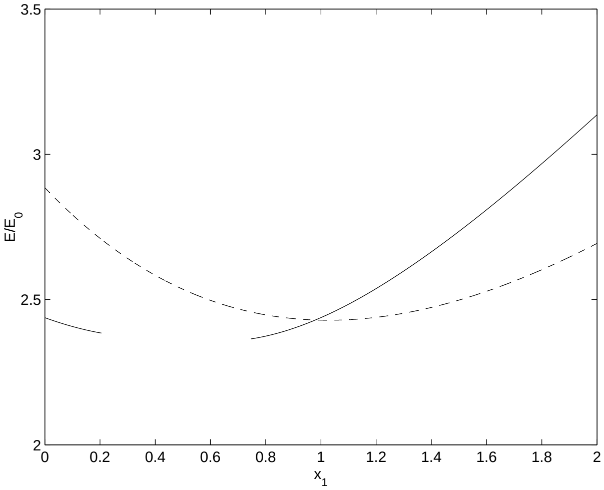

Figure 3: The search cost as a function of the first scanning point for the optimal (solid line) and California Split (dashed line) strategies. The position of the object is exponentially distributed with mean 1.

Under Zipf's law, $\mathbb{P}[X > x] = (x + 1)^{-\alpha}$, $x \geq 0$, $\alpha > 0$, it easily follows that

$$-\alpha x_{i+1}(x_i + 1)^{-\alpha-1} + (x_{i-1} + 1)^{-\alpha} = 0$$

or equivalently

$$x_i = \frac{1}{\alpha}\left(\frac{x_{i-1}+1}{x_{i-2}+1}\right)^{\alpha}(x_{i-1} + 1)$$

with $x_2 = \alpha^{-1}(x_1+1)^{\alpha+1}$. We assume that $\alpha > 1$ in order to ensure a finite mean distance to the object. Next, we note that

$$f_i x_{i+1} = \frac{x_{i+1}}{(x_i+1)^\alpha} = \frac{1}{\alpha} f_{i-1} x_i + \frac{1}{\alpha} \frac{1}{(x_{i-1}+1)^\alpha},$$

which leads to

$$(\alpha - 1)\mathcal{E} = 1 + \alpha x_1 + \sum_{i \geq 1} (1+x_i)^{-\alpha}.$$

On the other hand $\mathcal{E}_o = (\alpha-1)^{-1}$, and so the ratio of the search cost to the optimal cost is given by $(\alpha-1)\mathcal{E}$. Numerical evaluation for $\alpha = 2$ yields a structure surprisingly similar to the one in the case of the exponential distribution. For x_1 in a finite interval, the sequence $\{x_i\}$ is not monotonic and the increase in the competitive ratio for the California Split strategy is less than 4%.

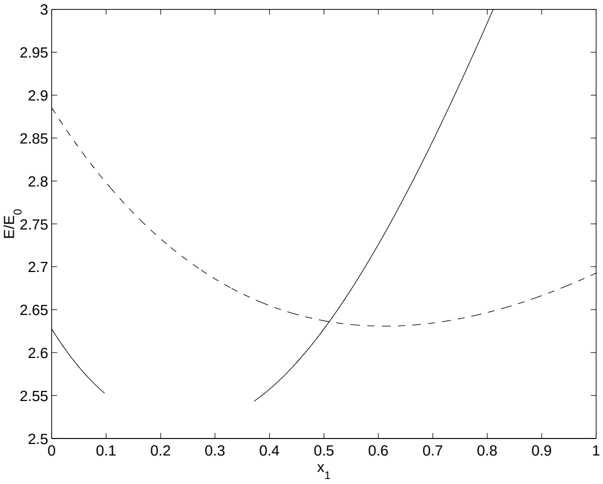

Figure 4: The ratio of search costs as a function of the first point for the optimal (solid line) and California Split (dashed line) strategies. The position of the object has Zipf's law with $\alpha = 2$.

4.1 Why gaps?

In this subsection we make an informal attempt to explain the appearance of gaps between the best starting points for scans as shown in Figures 3 and 4. It turns out that these gaps are by no means incidental, but rather reflect some general features pertaining to the extremes of the infinite sums

$$F(x_0, x_1, \ldots) = \sum_i g(x_i, x_{i+1}).$$

A variational principle states that the extremal point (or rather sequence) $x_0, x_1, \ldots$ is characterized by vanishing partial derivatives $\partial F/\partial x_i = 0$, $i = 0, 1, \ldots$. This defines a mapping on the plane

$$(x, y) \mapsto (y, z), \tag{8}$$

where $z = z(x,y)$ solves $g_2(x,y) = -g_1(y,z)$ (here, g_i is the partial derivative of g with respect to i-th argument). The mapping in (8) preserves the area form $g_{12}(x,y)dx \wedge dy$ (g_{12} is the mixed second derivative of g) and is therefore *Hamiltonian*. Generically, one expects a 2-dimensional Hamiltonian map to exhibit the so called *Hamiltonian chaos*: a complex structure composed of fixed points, closed orbits and chaotic regions between them, where points wander with dense orbits; the so-called *strange attractors* are typically present as well. Instead of giving a detailed description of these complex phenomena, we

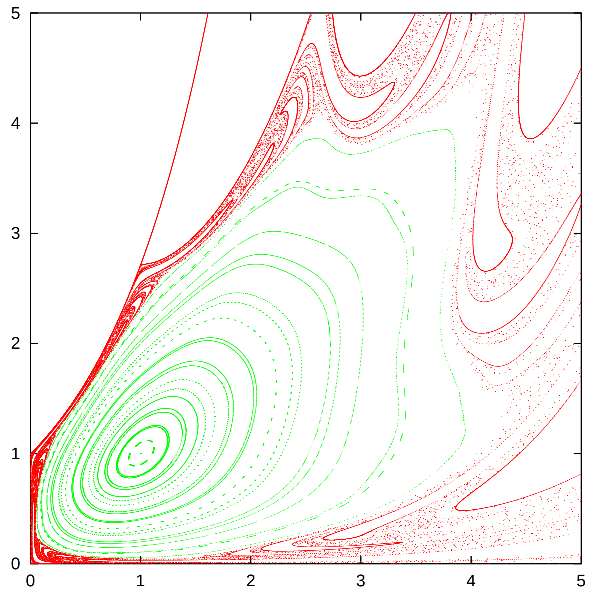

Figure 5: Several trajectories of the mapping $(x, y) \mapsto (y, e^{y-x})$. Fixed point $(1, 1)$ is the center of an integrable region surrounded by chaos.

refer the interested reader to any of many texts, e.g., [13]. Here, we just show a plot in Fig. 5 of several orbits of the map

$$(x, y) \mapsto (y, e^{y-x})$$

corresponding to $g(x, y) = ye^{-x}$, i.e., to the exponential distribution of the position of the object sought. One can see integrable regions near the fixed point $(1, 1)$, several periodic orbits and chaotic regions. Starting points of interest to us are those wandering off to infinity in a monotonically increasing fashion. They form an open set complemented by the gap.

5 Concluding remarks

We conclude the paper with brief discussions of two possible extensions of the model.

5.1 Infinite number of items

We focused on an infinite number of items since this assumption is more tractable and could approximate the case when the number of items is large. Consider a linear network of nodes enumerated by the nonnegative integers with searches being originated at the zero node. Sets of objects stored at different nodes are independent and identically distributed. This formulation of the problem is a one-dimensional version of the flood search problem in unstructured peer-to-peer networks. We use Y to denote the distance to an object and assume that searches follow the expanding ring algorithm. In addition, different objects may have different popularity. There are two sources of randomness in Y: (i) the object being searched for and (ii) the distance from the origin to that object. Let q_i be the relative popularity of item i, i.e., item i is being sought with probability q_i. Since the content of each node is independent of the contents of other nodes, the distance to the nearest object i is a geometric random variable with a parameter that depends on the replication strategy. Adopting the natural replication strategy, under which a node has item i with probability q_i, would not reveal much about flood-search performance. In that case the first moment of Y would be infinite

$$\mathcal{E}_o = \mathbb{E}Y = \sum_{i\geq 1} q_i \frac{1}{q_i} = \infty$$

as would the mean cost of any flood search algorithm. Instead, we assume that items are distributed according to the square-root rule, the optimality of which is shown in [4]. Under this rule a node stores item i with probability $\sqrt{q_i}$. Then, in analogy with (2), the expected search cost satisfies

$$\mathcal{E} = \sum_{i\geq 0} x_{i+1}\mathbb{P}[Y > x_i] := \sum_{i\geq 0} g_i x_{i+1}, \tag{9}$$

where

$$g_i \equiv g(x_i) := \sum_{j\geq 1} q_j (1-\sqrt{q_j})^{x_i}.$$

From (9) and (2) one readily concludes that there exists a mapping of the problem with multiple items to the case of a single object; the solution of the former is equivalent to the solution of the latter with the item being distributed according to $g(\cdot)$.

5.2 Adaptive California Split

In the model described in the previous subsection every item is geometrically distributed with a parameter that depends on its popularity. In general, the value of this parameter may be unknown in advance which makes the performance optimization of California Split impossible (the optimization of the first scanning point). Thus, it is of interest to have an adaptive version of the algorithm. This can be achieved by exploring the temporal locality in the request sequence. Namely, the popularity of two consecutive requests is assumed to be "close".

We consider the following algorithm. Let X be the position of the item that was found during the latest search. Then the first scanning point of the California Split in the next search is chosen to be βX, $\beta > 0$. When the position of objects is exponentially distributed, $\mathbb{P}[X > x] = e^{-\lambda x}$, a continuous relaxation of the geometric distribution with unknown popularity parameter λ, (2) yields

$$\mathcal{E}/\mathcal{E}_o = \mathbb{E}\left[\beta X + \sum_{i\geq 1} 2^i \beta X e^{-2^i\beta\lambda X}\right] = \beta + \beta\sum_{i\geq 1}\frac{2^i}{(1+\beta 2^{i-1})^2}.$$

As seen in Fig. 6, the optimal choice of β is around 0.61 which achieves $\mathcal{E}/\mathcal{E}_o = 2.631$. On the other hand, with exact knowledge of λ we have $\mathcal{E}/\mathcal{E}_o = 2.429$ (see Fig. 3). Hence, at the expense of a small increase of $\mathcal{E}/\mathcal{E}_o$ the new algorithm is adaptive, i.e., one does not need to know λ in advance.

References

[1] L. Adamic, R. Lukose, A. Puniyani, and B. Huberman. Search in power-law networks. *Phys. Rev. E*, 64, 2001.

[2] R. Baeza-Yates, J. Culberson, and G. Rawlins. Searching in the plane. *Information and Computation*, 106:234–252, 1993.

[3] I. Clarke, O. Sandberg, B. Wiley, and T. Hong. Freenet: A distributed anonymous information storage and retrieval system. In *Proc. of the Workshop on Design Issues in Anonymity and Unobservability*, Berkeley, CA, 2000.

[4] E. Cohen and S. Shenker. Replication strategies in unstructured peer-to-peer networks. In *Proc. ACM Sigcomm*, Pittsburgh, PA, August 2002.

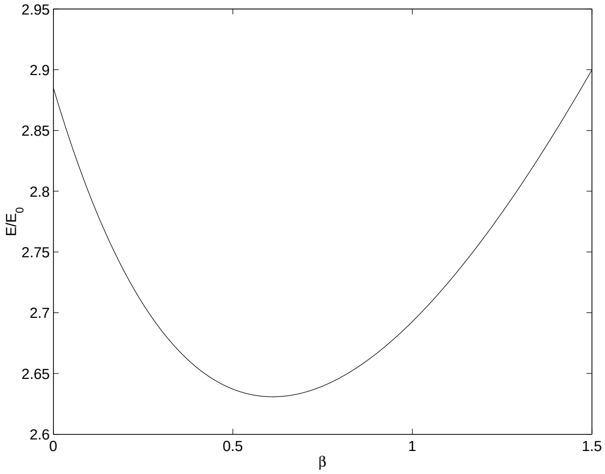

Figure 6: Ratio $\mathcal{E}/\mathcal{E}_o$ as a function of β for Adaptive California Split.

[5] S. Gal. *Search Games*. Academic Press, 1980.

[6] P. Jaillet and M. Stafford. Online searching. *Oper. Res.*, 49(4):501–515, 2001.

[7] P. Jaillet and M. Stafford. Note: Online searching / on the optimality of the geometric sequences for the m ray search. *Oper. Res.*, 50(4):744–745, 2002.

[8] M.-Y. Kao and J. Reif. Searching in an unknown environment: An optimal randomized algorithm for the cow-path problem. *Information and Computation*, 131:63–79, 1996.

[9] R. Karp, E. Koutsoupias, C. Papadimitriou, and S. Shenker. Algorithmic problems in congestion control. In *Proc. of FOCS*, Redondo Beach, CA, 2000.

[10] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker. Search and replication in unstructured peer-to-peer networks. In *Proc. of the 16th Annual ACM International Conference on Supercomputing*, New York, NY, June 2002.

[11] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content addressable network. In *Proc. of ACM Sigcomm*, 2001.

[12] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for Internet applications. In *Proc. of ACM Sigcomm*, 2001.

[13] M. Tabor. *Chaos and Integrability in Nonlinear Dynamics: An Introduction.* Wiley, New York, 1989.