J. Appl. Prob. **43,** 899–904 (2006)
Printed in Israel

SEARCHING FOR SEARCHERS

COLIN L. MALLOWS * ** AND
JEAN MELOCHE,* *** *Avaya Labs*

Abstract

We describe a search problem that has arisen in the context of network monitoring. Abstractly, a known (very large) region may contain one or more 'agents'. Starting with just one agent, we search until another agent is found; this new agent can assist in the remaining search, and so on recursively.

Keywords: Search; exploration; functional equation

2000 Mathematics Subject Classification: Primary 68P10; 60J85
Secondary 90B40; 39B99

1. Monitoring a network

The reader interested only in the idealized problem discussed in this paper should go directly to Section 2.

To monitor the quality of service of an IP (Internet protocol) network, engineers place communication end-points at strategic physical locations in that network. End-points require an IP address in order to communicate. IP addresses are assigned by the dynamic host configuration protocol. Once an end-point has been installed, it can inject traffic to other end-points to measure quality of service characteristics. But before this can be done, all IP addresses must be discovered, i.e. made known to some originating end-point. While the engineer knows what subnet (i.e. what range of IP addresses) each end-point is on, there is no direct way to find out which IP address in the range an end-point has been allocated. There may be more than one end-point in a subnet.

The problem for the engineer is thus to discover the IP addresses of the installed end-points. For that purpose, the end-points are programmed with the ability to discover their peers by probing IP addresses. If a functioning end-point receives a probe, it replies with a specific response that prevents false positives. Probing an IP address at which there is no end-point will result in either the wrong response or no reply at all. This discovery process starts from one end-point with the task of searching through the list of all IP addresses in all the ranges where end-points may be found.

Realistic numbers to work with are one end-point per 256 addresses, 1 million addresses to probe, and 100 probes per second. As the search progresses, newly discovered end-points can join in the search to speed up the discovery process.

The method is related to the statistical 'snowball sampling' method, in which 'interesting' people found in an initial sample are interrogated to suggest where more 'interesting' people might be found. Our analysis is also related to the theory of branching processes.

Received 18 February 2004; revision received 11 May 2006.
* Postal address: Avaya Labs, 233 Mt. Airy Road, Basking Ridge, NJ 07920, USA.
** Email address: colinm@research.avayalabs.com
*** Email address: jmeloche@research.avayalabs.com

2. A probability problem

The problem outlined in Section 1 suggests the following abstract probability problem. There is a region to be searched, consisting of a number of 'cells'. There may be some topology on the cells. We start with a single agent, which can search cells one by one. Each cell may or may not contain another agent. When a new agent is found, it can be co-opted to help with the search of the remaining cells. The question is, how long will it take to search all the cells? There is a considerable literature on search and exploration problems (see, for example, Knuth (1998)), but as far as the authors know this problem is a new one. We will usually assume that each cell contains an agent with the same probability p (this may not be realistic), and that each search takes the same length of time.

There are several possible search strategies. First, we may work hierarchically, i.e. when the first agent finds a new agent, it splits the remaining region into two equal parts (or differing by 1) and each agent works on its own region. Recursively, each new agent is assigned half of the remaining region of its 'parent'. This strategy may result in one agent working through a large region with no help while other agents have found several additional agents and are now idle, having finished searching their regions.

A more efficient (time-saving) strategy is that when an agent becomes idle, it is offered for adoption by its parent to help with part of that parent's region. When a parent with offspring becomes idle, it and all its offspring are offered to the grandparent, etc. This will result in efficient searching even when agents are not distributed uniformly throughout the region. This strategy can be modelled by simply assuming that all active agents search the whole remaining region efficiently. (However, this ignores the additional bookkeeping that is needed.) In a limiting case, we assume that agents are situated along the positive real line according to a stationary Poisson process. The search starts at the origin, moving to the right at unit speed; every time a new agent is found, the speed increases by 1.

3. Efficient searching

Consider searching an unlimited number of cells, where for each cell the probability that it contains an agent is $p = 1 - q$, independently for all cells. If the time to complete searching n cells is T_n and the number of cells searched in time t is N_t, then we have

$$\{T_n \le t\} = \{N_t \ge n\}.$$

For convenience, we can work with N_t (for fixed t) rather than T_n (for fixed n).

We start (at time 0) with a single agent. Suppose that at time t we have k active agents and have searched n cells. Let the probability of this (given t) be $\mathrm{P}(t, k, n)$. Initially we have $t = 0$, $k = 1$, and $n = 0$. At time $t + 1$ another k cells will have been searched and some random number X_t of new agents will have been found, so that now there are $K_t = k + X_t$ agents. The distribution of X_t is binomial(k, p). We obtain the generating function

$$F(u, x, y) = \sum u^t x^k y^n \,\mathrm{P}(t, k, n),$$

where all sums, on t, k, and n, are from 0 to ∞. Considering what will happen in the next time interval, we have

$$F(u, x, y) = x + uF(u, qxy + px^2 y, y); \tag{1}$$

whence, by differentiation and setting x and y to 1, we can derive moments of K_t and N_t.

https://doi.org/10.1239/jap/1158784957 Published online by Cambridge University Press

Putting $s = 1 + p$, we find that

$$\mathrm{E}(K_t) = s^t, \qquad \mathrm{var}(K_t) = q(s^{2t-1} - s^{t-1}),$$
$$\mathrm{E}(N_t) = \frac{s^t - 1}{p}, \qquad \mathrm{var}(N_t) = \frac{(s^{2t-1} - (2t-1)ps^{t-1} - 1)q}{p^2}.$$

The authors have not seen how to solve (1) explicitly.

4. Asymptotics

The moments suggest that for t large the distribution of N_t is approximately the same as that of K_t/p, and that, as $t \to \infty$, the distributions of $K_t' = K_t/s^t$ and $N_t' = N_t/s^t$ may approach limits. Assuming this, we find that, for each of $X = K'$ and $X = pN'$, the Laplace transform $f(u) = \mathrm{E}(\exp(-uX))$ must satisfy

$$f((1+p)u) = qf(u) + pf(u)^2, \tag{2}$$

with $f'(0) = 1$. It is not hard to show that all moments exist, and that these moments do determine the distribution of X. We find that

$$m_1 = 1, \qquad m_2 = \frac{2}{s}, \qquad m_3 = \frac{12}{s^2(s+1)}, \qquad m_4 = \frac{24(5+s)}{s^3(s+1)(s^2+s+1)},$$

where $s = 1 + p$. The distribution of X approaches a standard exponential distribution as $s \to 1$ (i.e. $p \to 0$) and approaches a point mass at $X = 1$ as $s \to 2$ (i.e. $p \to 1$).

After much effort, the authors have not been able to solve (2) explicitly. It appears that $f = f(p, u)$ has a Taylor expansion about $p = 0$, the first two terms of which are

$$f(p,u) = \frac{1}{1+u} - \frac{pu\log(1+u)}{(1+u)^2} + \cdots,$$

but we have not found the next term.

Another asymptotic approximation can be obtained as follows. Suppose that new agents appear according to a stationary Poisson process on $(0, \infty)$ with rate λ. The gaps $g_1, g_2, \ldots$ between successive new agents are independent exponential variables with mean $1/\lambda$. If we fix k then the position of the kth new agent is $X_k = g_1 + \cdots + g_k$, and the time it takes to search up to this agent is

$$T_k = g_1 + \frac{g_2}{2} + \frac{g_3}{3} + \cdots + \frac{g_k}{k},$$

so that (for k fixed)

$$\mathrm{E}(X_k) = k, \qquad \mathrm{var}(X_k) = k, \qquad \mathrm{E}(T_k) \approx \gamma + \log(k), \qquad \mathrm{var}(T_k) \approx \frac{\pi^2}{6},$$

where $\gamma = 0.5772$ is Euler's constant. The Laplace transform of the distribution of T_k is

$$\prod_{j=1}^{k} \frac{1}{1 + s/j}.$$

As k increases, the distribution of $T_k - \log k - \gamma$ approaches a limit, which is the double-exponential or extreme-value distribution

$$\mathrm{P}(T_k - \log k - \gamma < y) = \exp(-\exp(-y)),$$

https://doi.org/10.1239/jap/1158784957 Published online by Cambridge University Press

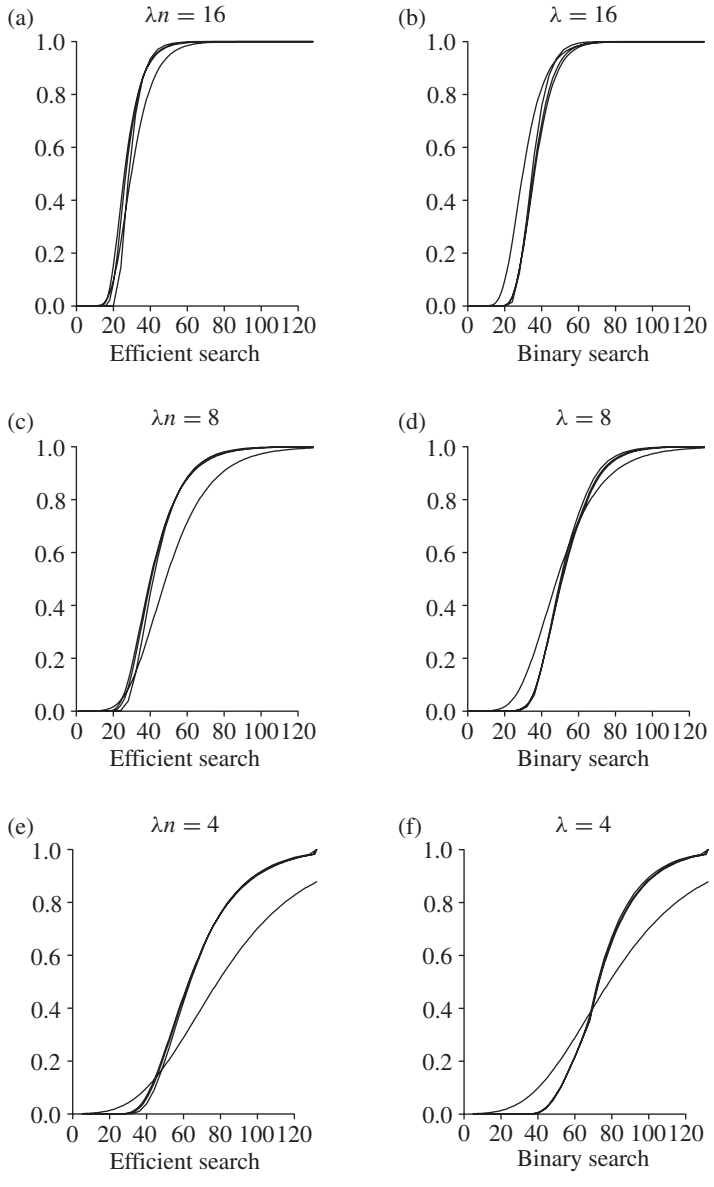

FIGURE 1: Exact and approximate distributions of T_n for nine cases; (a), (c), and (e) show an efficient search and (b), (d), and (f) show a binary search.

so that, as an approximation, we may take

$$\mathrm{P}(T_x < t) = \exp(-x \exp(\gamma - t)). \tag{3}$$

Equivalently, this approximation says that $\exp(-x \exp(\gamma - T_x))$ is uniform in (0, 1).

Figure 1 shows how this approximation performs. We calculated the exact distribution of T_n for three values of n, namely 32, 64, and 128, and three values of λn, namely 4, 8, and 16. Figures 1(a), 1(c), and 1(e) show the cumulative distributions of T_n for these nine cases, with

https://doi.org/10.1239/jap/1158784957 Published online by Cambridge University Press

the time scales adjusted to make the comparisons easier (for $n = 64$ we plot $\mathrm{P}(T < t)$ against $2t$, etc.) The curves for the three values of n (labelled 'Efficient search') lie on top of one another in each case. The approximation (3) is also shown. It appears that the approximation is poor for these values of λn, though it may be better for larger values. Figures 1(b), 1(d), and 1(f) show $\mathrm{P}(T < t)$ for the binary division strategy; see Section 6.

5. B-stability

Equation (2) is a special case of

$$f(cu) = g(f(u)), \tag{4}$$

where $f(u)$ is a Laplace transform and $g(z)$ is the probability generating function (PGF) of a random variable J, say, which takes values on the positive integers; thus, $g(z) = \mathrm{E}(z^J)$. The constant c is the mean of this distribution. In (2) we have $g(z) = qz + pz^2$. We recognise (4) as a generalization of the basic relation governing an explosive branching process. Suppose that a zeroth generation consists of a single individual, $X_0 = 1$. The PGF of this variable is simply $g_0(z) = z$. This individual has J descendants, so the PGF of the number X_1 of individuals in the first generation is $g_1(z) = g(z)$. Each of these has a random number of descendants, each distributed in the same way as J, and all mutually independent; so the PGF of X_2 is $g_2(z) = g(g(z))$. In general, we have $g_{n+1}(z) = g_n(g(z))$.

If $\mathrm{P}(J = 0) = 0$ and $c = \mathrm{E}(J)$ is finite, then the growth of X_n is exponentially explosive and $Y_n = X_n/c^n$ is a martingale. The distribution of Y_n converges, with the limiting Laplace transform satisfying (4). The relation (4) generalizes this by allowing $f(u)$ to be the transform of an arbitrary distribution. Mallows and Shepp (2005) studied the solutions of (4) in some generality.

6. Binary division

Suppose now that each agent is assigned a specific region to work on, and that when a new agent is found, it takes responsibility for half of the region assigned to its parent. This may be the simplest strategy to follow when the region to be searched has some topology, for example when it is a tree. The search ends when the slowest agent finishes. We have not seen how to tackle this case analytically, but a numerical approach is feasible. Let T_n be the time taken to search all of n cells, and denote by $\lfloor\cdot\rfloor$ and $\lceil\cdot\rceil$ the 'floor' and 'ceiling' functions, respectively. We have

$$\mathrm{P}(T_n \le t) = (1-p)\,\mathrm{P}(T_n - 1 \le t-1) + p\,\mathrm{P}(T_{\lfloor (n-1)/2\rfloor} \le t-1, T_{\lceil (n-1)/2\rceil} \le t-1), \tag{5}$$

so that, given p, values of $\mathrm{P}(T_n \le t)$ can be computed by induction on n. For n large, there is a Poisson process approximation

$$\mathrm{P}(T_x < t) = \int_0^x \mathrm{e}^{-w}\,\mathrm{P}^2(T_{(x-w)/2} < t-w)\,\mathrm{d}w,$$

which has proved to be intractable.

Numerical work with $p = 0.1$ shows that, for a binary search, T_{100} has a mean of about 71.2, and, for an efficient search, has a mean of about 62.1. The distribution of T_{100} for a binary search is close to that of T_{100} for an efficient search, translated by $+10$. For $p = 0.5$, T_{100} for a binary search has a mean of about 13.3 and is about 40% larger than T_{100} for an efficient

https://doi.org/10.1239/jap/1158784957 Published online by Cambridge University Press

search, which has a mean of about 9.7. Figures 1(b), 1(d), and 1(f) show the distributions of T_n for a binary search for the same nine cases as for the efficient case.

For a binary search, the distribution of T_n does not have a smooth density. This effect is negligible for large values of λn.

The numerical results suggest that the time needed to search n sites using a binary search is some 20% to 40% longer than for an efficient search. This penalty may not be large enough to warrant the additional organizational burden of an efficient search.

Several questions remain open for further work.

- Can (2) be solved explicitly?
- Is there a convenient correction term that will improve the approximation (3)?
- Can (5) be used to find the asymptotic form of the distribution of T_n for a binary search?

References

KNUTH, D. E. (1998). *The Art of Computer Programming*, Vol. 3, *Sorting and Searching*. Addison-Wesley, Reading, MA.

MALLOWS, C. AND SHEPP, L. (2005). B-stability. *J. Appl. Prob.* **42,** 581–586.

https://doi.org/10.1239/jap/1158784957 Published online by Cambridge University Press