Searching for an Optimal Path in a Tree with Random Costs*

Richard M. Karp
Computer Science Division, University of California, Berkeley, CA, U.S.A.

Judea Pearl
Cognitive Systems Laboratory, School of Engineering and Applied Science, University of California, Los Angeles, CA 90024, U.S.A.

ABSTRACT

We consider the problem of finding an optimal path leading from the root of a tree to any of its leaves. The tree is known to be uniform, binary, and of height N, and each branch independently may have a cost of 1 or 0 with probability p and 1 − p, respectively.

We show that for $p < \frac{1}{2}$ the uniform cost algorithm can find a cheapest path in linear expected time. By contrast, when $p > \frac{1}{2}$, every algorithm which guarantees finding an exact cheapest path, or even a path within a fixed cost ratio of the cheapest, must run in exponential average time. If, however, we are willing to accept a near optimal solution almost always, then a pruning algorithm exists which finds such a solution in linear expected time. The algorithm employs a depth-first strategy which stops at regular intervals to appraise its progress and, if the progress does not meet a criterion based on domain-specific knowledge, the current node is irrevocably pruned.

1. Introduction

It is often said that heuristic methods are unpredictable; they work wonders *most* of the time, but may fail miserably *some* of the time. Indeed, some heuristics greatly reduce search effort but occasionally fail to find an optimal or even a near-optimal solution. Others work efficiently on most problems in a given domain but a rare combination of conditions and data may cause the search to continue forever.

Since the ultimate test for the success of heuristic methods is that they work

*This work was supported in part by The National Science Foundation, Grant MCS 8114209.

Artificial Intelligence **21** (1983) 99–116
0004-3702/83/0000–0000/$03.00

well 'most of the time', and since probability theory is our principal formalism for quantifying concepts such as 'most of the time', it is only natural that probabilistic models should provide a formal ground for evaluating the performance of heuristic methods quantitatively. Moreover, it is equally natural that probabilistic models of the problem domain should participate in the process of devising heuristic methods, or in selecting the parameters which govern these methods so as to guarantee that only a small fraction of problem instances will escape an adequate treatment.

In many practical problems we are interested in optimizing some performance measure involving a combination of solution quality and search effort *averaged* over all problems likely to be encountered. The average value of this performance measure, however, is seldom actually computed, primarily because it is difficult to define and analyze a probability distribution which adequately represents the set of problems to be encountered.

Nevertheless, in order to understand what attributes make a candidate search method suitable or unsuitable for a given class of problems some analysis must be conducted. Toward this goal one normally starts with a simplified synthetic model, containing the smallest number of parameters required for representing the salient features of the problem domain, and then analyzes the performance of various search algorithms on this synthetic model as a function of the model's parameters. If, as a result of such an exercise, it becomes apparent that the value of a certain parameter has a dramatic influence over the relative performances of the candidate search methods, then that parameter becomes a focus of attention in empirical efforts to characterize the problem environment. Alternatively, if a given search method appears to exhibit superior performance over a wide range of model parameters, that method then becomes the first candidate to be tried on actual problem instances.

This paper presents a tractable simplified synthetic model for path optimization problems and analyzes the performances of two search methods within this model.

Problem. Find the cheapest path leading from the root of a tree to any of its leaves. The tree is known to be uniform, binary, and of height N. Each branch independently may have a cost of 1 or 0 with probability p and $1-p$, respectively.

The first search method involves a blind, uniform-cost algorithm [3] which, of course, makes no use of heuristics regarding the unexplored portion of the search tree. The second method is a version of staged-search [3] which periodically uses heuristic information to prune away those 'bad' nodes which do not perform in accordance with expectations.

Our analysis shows that the uniform-cost algorithm can be remarkably efficient when the branch costs are more likely to assume the value zero. By contrast, when the branch costs are more likely to have a unity cost, the uniform-cost algorithm requires exponential time while the pruning algorithm runs in linear average time and almost always finds a near optimal solution.

2. Notation and Preliminaries

The problem statement defines a task to be executed over an ensemble of problem instances. In each problem instance we have to find the cheapest root–leaf path in a tree T drawn from the sample space $\boldsymbol{T}(N, p)$ containing uniform binary trees of height N with 0–1 edge costs. The probability that a tree T with a specified assignment of edge costs is drawn from $\boldsymbol{T}(N, p)$ is given by $p^{N_1}(1-p)^{N_0}$ where N_1 and N_0 are respectively the number of edges in T with cost 1 and 0 ($N_1 + N_0 = 2^{N+1} - 2$).

We shall say that a node J in the tree has cost c if c is the sum of the costs of the edges along the path from the root to J. Of particular interest to us are the costs of leaves and especially the cost of the cheapest leaf, which is our target of pursuit. Let the random variable $C(N, p)$ be the minimum number of 1's on a root–leaf path in a tree T drawn from $\boldsymbol{T}(N, p)$. We shall see later that the distribution of $C(N, p)$ will vary significantly with p and will assume a different character in the following three regions: $p < \frac{1}{2}$, $p = \frac{1}{2}$, and $p > \frac{1}{2}$. Moreover, the nature of this distribution will dictate which algorithm is most suitable for search and so, we will discuss these three regions separately.

We shall say that a node J' is a (α, L)-*son* of node J iff the following two conditions hold:

(1) J' is situated L levels below J, and

(2) the cost of the path connecting J and J' does not exceed αL.

A family line of t successive (α, L)-sons represents a path in T consisting of t consecutive segments each with cost at most αL. We shall call such a path a (α, L)-*regular* path, and families generated by this process (α, L)-regular families. In order to keep our notation simple we shall still call a path terminating at a leaf node (α, L)-regular even when its last segment contains fewer than L nodes as long as the cost of that last segment is at most αL. Clearly, any root-leaf path which is (α, L)-regular may not cost more than $\alpha(N + L)$.

We shall analyze the performance of two search algorithms which we shall call A_1 and A_2. A_1 is a *uniform-cost* algorithm and will be analyzed for the cases: $p < \frac{1}{2}$ and $p = \frac{1}{2}$. A_2 is a hybrid of local and global *depth-first* search strategies and will be analyzed for $p > \frac{1}{2}$.

Both algorithms start with a subtree containing the root of T and, at a general step, expand some node on the frontier of the subtree. Expanding a node J means creating its two children and the edges from J to these children.

Algorithm A_1. At each step, expand the leftmost node among those frontier nodes of minimum cost. The algorithm halts when it tries to expand a leaf of T. That leaf then represents an optimal solution.

Algorithm A_2. A_2 conducts a depth-first-search to find a (α, L)-regular path from a level d node to a leaf where d, α, and L are three externally chosen parameters. In other words, A_2 is a depth-first strategy which stops at regular intervals of L levels to appraise its progress. If the cost increase from the last appraisal is at most αL, the search continues; if that cost increase is above αL, the current node is irrevocably pruned, the program backtracks to a higher level, and the search resumes. If it succeeds in finding a (α, L)-regular path, A_2 returns that path as a solution; its cost is at most $d + \alpha(N - d + L)$. If it fails, the search is repeated from another d-level node. If all the 2^d nodes at level d fail to root a (α, L)-regular path to a leaf, A_2 terminates with failure. The probability of such an event, however, will be made vanishingly small by an appropriate choice of the parameters d, α, and L.

This search is shown schematically in Fig. 1. It consists of two components: (1) local depth-first search (with depth-bound L) to find (α, L)-sons, and (2) global depth-first search on members of the (α, L)-family, seeking a family line extending to level N.

The rationale for Algorithm A_2 is as follows: probabilistic analysis reveals (Theorem 3.3) that for $p > \frac{1}{2}$ and large N the optimal cost $C(N, p)$ lies very near $\alpha^* N$ where α^* is a constant dictated by p. If we settle for finding a near-optimal solution, we may stop the search as soon as we find any leaf node with cost below αN $(\alpha > \alpha^*)$ where α is chosen sufficiently close to α^*. For-

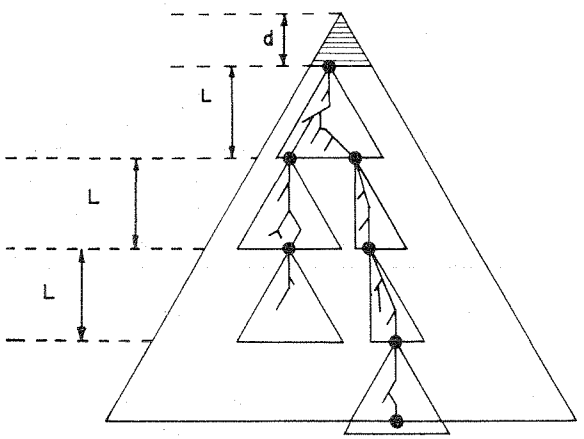

FIG. 1. Schematic representation of the search algorithm A_2. The triangles represent local depth-first searches for (α, L)-sons (solid circles).

tunately there are many leaves with costs between $\alpha^* N$ and αN and so, we are at liberty to confine our search to a special subset of these leaves which are easy search targets. A convenient choice would be the set of leaves reachable by (α, L)-regular paths, along which the cost increases by not more than a fixed increment αL in every successive path segment of length L. Such paths are convenient search targets because violations of the requirement for a gradual cost increase (unlike violations of requirements posed on the overall cost of a path) are detectable by local analysis. Indeed, we shall see later (Theorem 3.6) that A_2 runs in linear expected time and, if the parameters d and L are chosen appropriately, A_2 is guaranteed to find a near optimal solution with probability approaching one.

3. Summary of Results

The results of our analysis are summarized in the following six theorems. The first three theorems determine the asymptotic distribution of the optimal cost $C(N, p)$ while the later three quantify the expected complexities of algorithms A_1 and A_2 in the three regions of p.

Theorem 3.1. *If* $p < \frac{1}{2}$, *then*

$$P[C(N, p) > k] \leq (2p)^{2^{k+1}-2}, \quad k = -1, 0, 1, \ldots$$

Thus, the optimal cost is almost certain to remain bounded as $N \to \infty$.

Theorem 3.2. *If* $p = \frac{1}{2}$, *then, for every unbounded monotone function* H,

$$P[|C(N, \tfrac{1}{2}) - \log_2 \log_2 N| > H(N)] \to 0 .$$

This implies that the optimal cost is very likely to be near $\log_2 \log_2 N$.

Theorem 3.3. *Let* $p > \frac{1}{2}$ *and*

$$G(\alpha, p) = \left(\frac{p}{\alpha}\right)^{\alpha} \left(\frac{1-p}{1-\alpha}\right)^{1-\alpha} .$$

Define α^* *by the equation* $G(\alpha^*, p) = \frac{1}{2}$.

For $\alpha < \alpha^*$

$$P[C(N, p) \leq \alpha N] = O(c_\alpha^{-N}), \quad c_\alpha > 1 .$$

For $\alpha > \alpha^*$

$$P[C(N, p) > \alpha N] = O(d_\alpha^{-N}), \quad d_\alpha > 1 .$$

This theorem states that the optimal cost is proportional to N and that the proportionality factor is very likely to be near α^*.

Let the random variables $t_1(N, p)$ and $t_2(N, p)$ be the number of nodes expanded in the execution of algorithm A_1 and A_2, respectively, on a tree T drawn from $\boldsymbol{T}(N, p)$.

Theorem 3.4. *If* $p < \frac{1}{2}$, *then* $E[t_1(N, p)] = O(N)$, *i.e.*, A_1 *finds an optimal cost leaf in linear expected time.*

Theorem 3.5. *If* $p = \frac{1}{2}$, *then* $E[t_1(N, \frac{1}{2})] = O(N^2)$, *i.e.*, A_1 *finds an optimal cost leaf in quadratic expected time.*

Theorem 3.6. *If* $p > \frac{1}{2}$, *then every algorithm which guarantees finding a path cheaper than* $(1+\varepsilon)C(N,p)$ *must run in exponential average time. However, for every* $\varepsilon > 0$ *it is possible to choose the parameters d and L in such a way that algorithm* A_2 *will find a solution costing at most* $(1+\varepsilon)C(N, p)$ *with probability approaching* 1 *and, moreover,* $E[t_2(N, p)] = O(N)$.

4. Proofs of Theorems 3.1–3.6

The proofs of Theorems 3.1–3.6 depend on some results from the theory of branching processes [1]. The basic notation and properties of branching processes are summarized in Appendix A while additional results are contained in the next two lemmas.

Lemma 4.1. *Let* Z_n *stand for the size of the nth generation in a branching process for which* $m = E(Z_1) \neq 1$, *then*

$$E\Big[\sum_n Z_n \mid \text{extinction}\Big] = Q < \infty .$$

Lemma 4.1 states that the expected size of an extinct family is bounded for both $m < 1$ and $m > 1$ even though the generation for which we can certify that extinction has occurred may be arbitrarily deep. The proof is given in Appendix B.

Lemma 4.2. *Let* $\{p_k\}$ *determine a branching process such that only finitely many of the* p_k*'s are non-zero. Define the random variable* $D(N)$ *as follows: let* T_0 *be the family tree created in a run of this process, and let* $D(N)$ *be the number of nodes encountered in a depth-first search of* T_0 *which terminates as soon as a node at depth* N *is reached. Then* $E[D(N)] = O(N)$.

Proof. If $m \leq 1$, then for each k, $E(Z_k) = m^k \leq 1$ [1, Theorem 5.1, p. 6], so

$$E[D(N)] \leq \sum_{k=0}^{N} E(Z_k) = \sum_{k=0}^{N} m^k \leq N + 1 .$$

Suppose $m > 1$. From Lemma 4.1 we know that $\mathrm{E}[D(N) \mid T_0 \text{ finite}] \leq Q$, so it is only necessary to show that

$$I_N = \mathrm{E}[D(N) \mid T_0 \text{ infinite}] = \mathrm{O}(N) .$$

Call a node J in T_0 *immortal* if the subtree rooted at J is infinite; otherwise, call J *mortal*. Assume J is immortal, and that J' is leftmost among the immortal sons of J. Then in searching the subtree rooted at J, J' is the only immortal son of J which is encountered. In addition to J', the search may also encounter all the mortal sons of J situated to the left of J'. The expected number of nodes encountered in searching the subtree rooted at J' is I_{N-1}, while that encountered in searching each of its mortal siblings is at most Q. Consequently, letting M stand for the average number of mortal sons of an immortal father ($M < \infty$), we have:

$$I_N \leq 1 + I_{N-1} + MQ$$

from which it follows that

$$I_N \leq 1 + (N-1)MQ .$$

Proof of Theorem 3.1. Let

$$F_k^N = \mathrm{P}[C(N, p) > k] , \quad k = -1, 0, 1, \ldots, N-1 . \tag{1}$$

To establish a recurrence relation for F_k^N we condition the event $C(N, p) > k$ on the four possible events of the first generation, as shown in Table 1, and obtain the recursion

$$F_k^N = [(1-p)F_k^{N-1} + pF_{k-1}^{N-1}]^2 , \quad F_{-1}^N = 1 . \tag{2}$$

We now take the limit as $N \to \infty$

$$F_k = \lim_{N \to \infty} F_k^N = [(1-p)F_k + pF_{k-1}]^2 \tag{3}$$

which, for $p < \frac{1}{2}$, has the non-trivial solution

TABLE 1

cost of left arc	0	1	0	1
cost of right arc	0	0	1	1
probability	$(1-p)^2$	$p(1-p)$	$(1-p)p$	p^2
$P[C(N, p) > k]$	$(F_k^{N-1})^2$	$F_{k-1}^{N-1} \cdot F_k^{N-1}$	$F_k^{N-1} \cdot F_{k-1}^{N-1}$	$(F_{k-1}^{N-1})^2$

$$F_k = \frac{1}{2(1-p)^2}\left[1 - 2F_{k-1}p(1-p) - \sqrt{1 - 4F_{k-1}p(1-p)}\right]. \tag{4}$$

Using the inequality

$$\sqrt{1-x} \geq (1-x)(1+\tfrac{1}{2}x) \quad \text{for } 0 \leq x \leq 1 \tag{5}$$

we obtain a bound on F_k in (4)

$$F_k \leq 4p^2(F_{k-1})^2. \tag{6}$$

The boundary condition $F_{-1} = 1$, together with (6) leads to the desired inequality

$$F_k \leq \frac{1}{4p^2}(4p^2)^{2^k} = (2p)^{(2^{k+1}-2)}. \tag{7}$$

Note that for $p \geq \frac{1}{2}$, the only meaningful solution of (3) is $F_k = 1$ for all k, implying that as $N \to \infty$ the cost of the cheapest path is almost surely unbounded.

Proof of Theorem 3.2. See Appendix C.

Proof of Theorem 3.3. Let $Z(\alpha, N)$ stand for the number of leaves in a tree of height N with costs not exceeding αN, $0 < \alpha < 1$. We first wish to show that the average number of such leaves undergoes a critical jump at $\alpha = \alpha^*$ as $N \to \infty$.

There are a total of 2^N root–leaf paths in the tree. The probability that the cost of any one path does not exceed αN is given by the binomial distribution

$$\mathrm{P}(\text{path cost} \leq \alpha N) = B_{p,N}(\alpha N) \stackrel{\Delta}{=} \sum_{i=0}^{\alpha N} \binom{N}{i} p^i (1-p)^{N-i} \tag{8}$$

and therefore

$$\mathrm{E}[Z(\alpha, N)] = 2^N B_{p,N}(\alpha N). \tag{9}$$

The binomial distribution can be bounded [2] between two entropy related functions, provided $\alpha < p$,

$$\frac{1}{\sqrt{8N\alpha(1-\alpha)}} G^N(\alpha, p) \leq B_{p,N}(\alpha N) \leq G^N(\alpha, p) \tag{10}$$

where

$$G(\alpha, p) = \left(\frac{p}{\alpha}\right)^{\alpha}\left(\frac{1-p}{1-\alpha}\right)^{1-\alpha} \tag{11}$$

or

$$\log G(\alpha, p) = (1-\alpha)\log(1-p) + \alpha \log p + H(\alpha) \tag{12}$$

where $H(\alpha)$ is the entropy function

$$H(\alpha) = -[\alpha \log \alpha + (1-\alpha)\log(1-\alpha)]\,. \tag{13}$$

Fig. 2 below depicts the dependence of $\log_2 G$ on α and p. For $p > \frac{1}{2}$, $\log_2 G$ crosses the level of -1 at $\alpha = \alpha^* < p$ and at this point $\mathrm{E}[Z(\alpha, N)]$ undergoes an abrupt jump:

$$\lim_{N\to\infty} \mathrm{E}[Z(\alpha, N)] = \lim_{N\to\infty} [2G(\alpha, p)]^N = \begin{cases} \infty, & \alpha > \alpha^*, \\ 0 & \alpha < \alpha^*. \end{cases} \tag{14}$$

The critical value α^* is defined by the equation:

$$G(\alpha^*, p) = \tfrac{1}{2} \tag{15}$$

and it varies with p in the manner shown in Fig. 3.

The fact that $\mathrm{E}[Z(\alpha, N)]$ vanishes for $\alpha < \alpha^*$ implies that the cheapest path has only vanishingly small probability of costing less than αN. In fact, from

$$\mathrm{E}[Z(\alpha, N)] \geqslant \mathrm{P}[C(N, p] \leqslant \alpha N]$$

together with (9) and (10), we have

$$\mathrm{P}[C(N, p) \leqslant \alpha N] \leqslant [2G(\alpha, p)]^N \tag{16}$$

and, since $2G(\alpha, p) < 1$ for $\alpha < \alpha^*$, the first part of Theorem 3.3 is established.

The behavior of $\mathrm{P}[C(N, p) \leqslant \alpha N]$ for $\alpha > \alpha^*$ is not implied by (14), but can be derived using a branching process argument on subtrees with depth L.

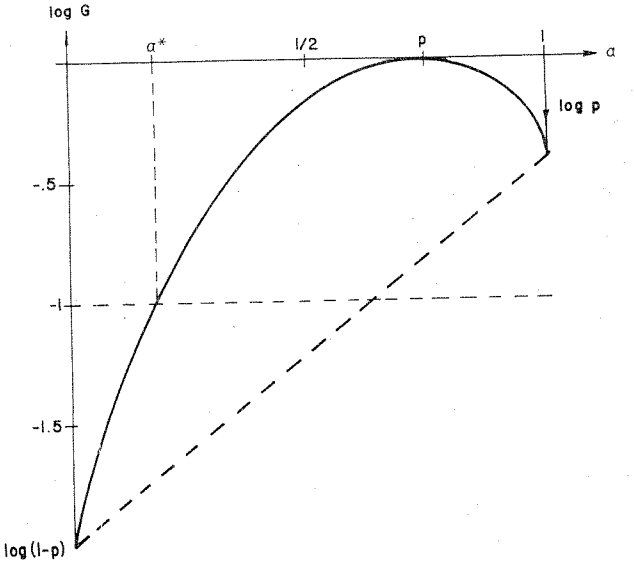

FIG. 2. The dependence of $\log_2 G$ on α and p.

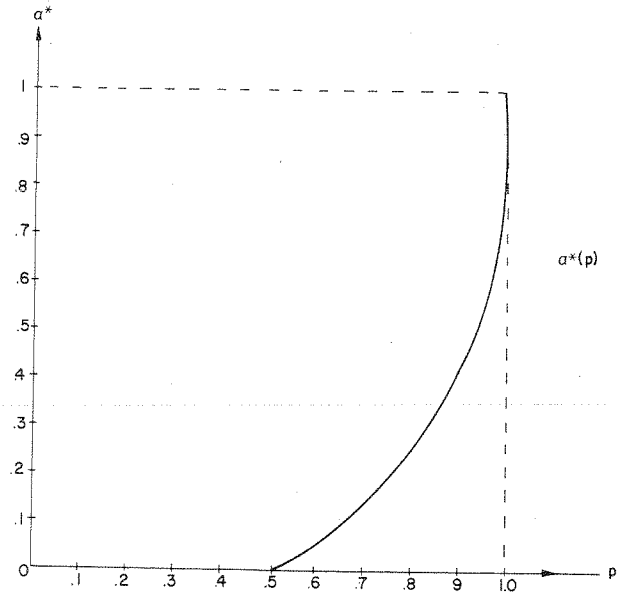

FIG. 3. The dependence of α^* on p. $\alpha^* N$ is the most likely cost of the cheapest path in the tree.

From (9) and (10) we can write

$$\mathrm{E}[Z(\alpha, L)] \geqslant \frac{1}{\sqrt{8L\alpha(1-\alpha)}}[2G(\alpha, p)]^L \tag{17}$$

implying that if $\alpha > \alpha^*$, one can always find an L large enough (but independent of N) such that the expected number of (α, L)-sons of the root node is greater than unity. For each α, let us define the lowest possible value of L which satisfies this condition

$$L_\alpha = \min\{L \mid \mathrm{E}[Z(\alpha, L)] > 1\}, \tag{18}$$

i.e., L_α is the lowest value of L such that the expected number of leaf nodes costing at most αL in a tree drawn from $\boldsymbol{T}(L, p)$ exceeds unity.

The family of (α, L_α)-sons reproduces like a branching process for which $m > 1$. Such a family has a non-zero probability $1 - q_\alpha$ of lasting forever and, therefore, the probability of finding a (α, L_α)-regular path of length N (and cost at most αN) is at least equal to $1 - q_\alpha$.

We shall now show that the probability of finding a path costing at most αN approaches unity if $\alpha > \alpha^*$ and $N \to \infty$. Choose a value β between α^* and α, $\alpha^* < \beta < \alpha$, and a depth L_β such that the expected number of (β, L_β)-sons of each node be larger than unity. Any node at depth d of the tree has a probability of at least $1 - q_\beta$ of nucleating a (β, L_β)-regular path reaching level

N and costing at most $\beta(N-d+L_\beta)$. Since there are 2^d such nodes at depth d, the probability that at least one of them resides on top of a (β, L_β)-regular path to a leaf node is higher than $1-(q_\beta)^{2^d}$. The total cost of such a path including the portion from the root to depth d is at most $\beta(N+L_\beta)+d(1-\beta)$. Thus, we can write

$$\mathrm{P}[C(N,p)>\beta(N+L_\beta)+d(1-\beta)]\leq(q_\beta)^{2^d}. \tag{19}$$

Choosing d to vary slowly with N, e.g., $d=\lceil\log_2 N\rceil$, yields

$$\mathrm{P}[C(N,p)>\beta(N+L_\beta)+(1-\beta)(1+\log N)]\leq(q_\beta)^N.$$

Now, β was chosen strictly smaller than α and so, for sufficiently large N, we have

$$\mathrm{P}[C(N,p)>\alpha N]\leq(q_\beta)^N \tag{20}$$

which establishes the second part of Theorem 3.3.

The arguments used in this proof will dictate the appropriate choice of the parameters d, α, and L for the A_2 algorithm (see Theorem 3.6).

Proof of Theorem 3.4. The execution of Algorithm A_1 involves a series of *trials*, each of which consists of a depth-first search in the subtree of T rooted at some vertex J, and containing those vertices reachable from J at cost zero. The algorithm terminates as soon as some trial reaches a leaf of T.

Consider the branching process π in which the birth of a son corresponds to the discovery of a zero cost edge. For this process we have: $p_0=p^2$, $p_1=2p(1-p)$ and $p_2=(1-p)^2$. Since $p<\frac{1}{2}$, $m=2(1-p)>1$, and so the extinction probability q is less than 1. Each trial, starting at a node of depth d, corresponds to a depth-first search in a family tree associated with a run of the process π; the search terminates when a $(N-d)$th generation node is reached as in the conditions set for Lemma 4.2. The expected time for such a search is $D(N-d)\leq D(N)$ and the expected number of trials cannot exceed $1/(1-q)$, which is the expected number of trials until encountering the first immortal node. Hence, the expected number of nodes expanded by the algorithm is at most $D(N)/(1-q)$ and, using Lemma 4.2,

$$\mathrm{E}[t_1(N,p)]=\mathrm{O}(N).$$

Proof of Theorem 3.5. The proof is similar to that of Theorem 3.4. The expected time for each trial is still at most $D(N)$, and the probability that a trial succeeds is at least $\mathrm{P}[Z_N>0]$ for the process π. This process now has $m=1$ and so (see Appendix A)

$$\mathrm{P}[Z_N>0]\sim 2/N\sigma^2.$$

Hence, the expected number of trials is at most $\frac{1}{2}N\sigma^2 + O(N)$, and the expected number of nodes expanded satisfies

$$E[t_1(N, p)] \leq D(N)(\tfrac{1}{2}N\sigma^2 + o(N)) = O(N^2).$$

Proof of Theorem 3.6. If a tree $T \in \boldsymbol{T}(N, p)$ has a minimal cost $C(T)$, then every algorithm A which is guaranteed to return a path costing $C_R(T) \leq (1+\varepsilon)C(T)$ must examine every node at level $\lfloor C(T)/(1+\varepsilon')\rfloor$ of T, if $\varepsilon' > \varepsilon$. This is so because if A skips a node n at that level, it will also return the same solution, costing $C_R(T') = C_R(T) \geq C(T)$, on another tree T' which is identical to T in all respect but has an all zero's extension to n. This would violate the stated guarantee for A because the cheapest path of T' can cost at most $\lfloor C(T)/(1+\varepsilon')\rfloor < C(T)/(1+\varepsilon)$ while A would return $C_R(T') = C_R(T) \geq C(T) > (1+\varepsilon)C(T')$. At the same time Theorem 3.3 states that trees with $C(T) > (1-\delta)\alpha^* N$, $\delta > 0$, occur with probability $P_\delta > 0$, hence the expected number of nodes examined by A must be at least $P_\delta 2^{\lfloor((1-\delta)/(1+\varepsilon'))\alpha^* N\rfloor}$ which is exponential in N.

The second part of Theorem 3.6 is proven by the following construction. Given N, p, and ε, calculate α^* from (15) and let $d(N)$ be any unbounded monotone function of N such that $d(N) = o(N)$. For N sufficiently large it is possible to find an α between α^* and $(1+\varepsilon)\alpha^*$ satisfying

$$N\alpha + (1-\alpha)d(N) + \alpha L_\alpha \leq (1+\varepsilon)\alpha^* N.$$

Choose this α together with its associated L_α (18) and with $d(N)$ as the three input parameters for algorithm A_2.

From the proof of Theorem 3.3 we know that Algorithm A_2, when governed by these three parameters, will fail to reach a leaf of T with probability at most $(q_\alpha)^{2^{d(N)}}$. Moreover, when the search succeeds, it returns a path costing at most $N\alpha + (1-\alpha)d(N) + \alpha L_\alpha \leq (1+\varepsilon)\alpha^* N$ (see (19)). Thus, the probability that A_2 would fail to return a solution within a cost factor $(1+2\varepsilon)$ of the optimal is at most $P(A_2 \text{ fails to reach a leaf}) + P[C(N, p) < (1-\varepsilon)\alpha^* N]$ and, since both terms tend to zero, the second part of Theorem 3.6 is established.

To show that A_2 runs in linear expected time, we consider the depth-first search on an (α, L_α)-regular family. Let the existence of a (α, L_α)-son for some node J be regarded as a reproduction event in a branching process π', i.e., $p_k = P[Z(\alpha, L_\alpha) = k]$. Our choice of L_α (18) guarantees that for this process $m > 1$ and $q < 1$. We shall for simplicity assume that d is chosen so that $N - d$ is a multiple of L_α, which makes the $((N-d)/L_\alpha)$th generation of π' coincide with the leaf nodes of T. A *trial* in the execution of A_2, is a depth-first search through a family tree from π', terminating as soon as a node at depth $(N-d)/L_\alpha$ is reached. The expected number of nodes in a family tree of π' expanded in each trial is $D((N-d)/L_\alpha) = O(N)$. Each such node expansion involves creating at most $2^{L_\alpha} - 1$ nodes of T by the local search of A_2. The expected number of trials until Algorithm A_2 terminates is at most $1/(1-q)$, and so the expected number of nodes

expanded in Algorithm A_2 is

$$E[t_2(N, p)] \leq D\left(\frac{N-d}{L_\alpha}\right)(2^{L_\alpha}-1)\left(\frac{1}{1-q}\right) = O(N).$$

5. Conclusions

Finding a cheapest path in a tree, or even a path within a fixed cost ratio of the cheapest, requires exponential time in the worst case. It is remarkable, therefore, that subject to our probabilistic assumptions, both algorithms are executed in linear or quadratic expected times.

The blind search uniform-cost algorithm (A_1) derives its efficiency from the smallness of the optimal cost in the range $0 \leq p \leq \frac{1}{2}$. The solution path will normally contain only a few 1's. As a result a leaf node will be reached by A_1 after 'peeling' only a few cost layers. Moreover, backtracking from off-track excursions is extremely cheap; Lemma 4.1 asserts that the average effort spent in failing to find a zero-cost extension to any node is bounded (for $p < \frac{1}{2}$) or $O(N)$ (for $p = \frac{1}{2}$).

In the range $p > \frac{1}{2}$, A_1 becomes exponential. The optimal cost now grows linearly with N, and so disappointing excursions will only be abandoned after searching through a large number of cost layers. It is, in fact, possible to show that in the range $p > \frac{1}{2}$ any admissible algorithm, i.e., one which is guaranteed to find an exact optimal solution, must run in exponential average time. Therefore, the transition from $p \leq \frac{1}{2}$ to $p > \frac{1}{2}$ should serve as an indicator to abandon admissible search strategies altogether.

The success of the pruning Algorithm A_2 can be attributed to two factors: (1) our willingness to accept a near optimal solution *almost* always, and (2) the availability of probabilistic, domain-specific knowledge for determining the appropriate pruning criterion (i.e., determining α and L_α).

A_2 is another example in the class of 'bounded look-ahead plus partial backtrack' search strategies which have been proposed for the determination of near optimal solutions (almost everywhere) to certain NP-hard problems [3]. The common idea in this class of strategies is to examine nodes to determine their likelihood of yielding a near optimal solution and to prune away (irrevocably) unpromising nodes. Probabilistic knowledge is required to fine-tune the pruning criterion in such a way that *the number of nodes generated grows linearly instead of exponentially with problem size and, at the same time, near optimal solutions are obtained almost everywhere.*

Although it is hard to circumscribe exactly the type of problems which lend themselves to such a delicate tuning of the pruning criterion, the model analyzed in this paper highlights three conditions which essentially guarantee the success of this pruned depth-first method.

(1) The density of the optimal cost becomes highly concentrated about some predetermined function $K(N)$ of the problem size N.

(2) There exists a pruning criterion based on local information (i.e., each test can be completed in constant time) that will detect and prune away every solution which lies outside a specified neighborhood of $K(N)$.

(3) In almost every problem instance, the neighborhood about $K(N)$ should contain at least one solution which will survive the pruning axe.

In our model (and for $p > \frac{1}{2}$) condition (1) is satisfied by virtue of $C(N, p)$ concentrating about $\alpha^* N$ (Theorem 3.3). Condition (2) holds because by discarding all but (α, L)-sons we guarantee that for any arbitrary L each surviving path would cost at most $\alpha(N + L)$. Hence, choosing $\alpha \leq (1 + \varepsilon)\alpha^*$ assures the $\alpha^* N$-neighborhood of all survivors. Finally, condition (3) is satisfied by our ability to always find an L_α such that $\mathrm{E}[Z(\alpha, L_\alpha)] > 1$, thus guaranteeing a non-zero probability of survival which is further amplified to almost unity by postponing the pruning until past depth $d(N)$.

We hope that these local pruning strategies will find an increased popularity in application areas such as speech processing and scene analysis.

Appendix A. Basic Properties of Branching Processes

Branching processes describe phenomena in which objects generate additional objects of the same kind and where different objects reproduce independently of one another. Search strategies are related to branching processes in that the operation of node expansion can be viewed as a reproduction process; the node expanded gives rise to a certain number of children nodes, a random number of which will eventually be expanded.

Notation. A branching process is characterized by a set $\{p_k\}$ of probabilities, where

$$p_k = \mathrm{P}[\text{a typical father has } k \text{ sons}]$$

or equivalently, by the associated generating function,

$$f(s) = \sum_k p_k s^k .$$

Two important parameters dictated by $\{p_k\}$ are

$$m = \text{mean number of sons (of a typical father)} = f'(1),$$
$$\sigma^2 = \text{variance of the number of sons} = f''(1) + m - m^2.$$

Let Z_n be the size of the nth generation ($Z_0 = 1$). We define the generating function $f_{(n)}(s)$ for Z_n by

$$f_{(n)}(s) = \sum_n \mathrm{P}[Z_n = k] s^k .$$

Basic facts [1]. The generating function for Z_n is given by the nth iterate of $f(s)$, i.e.,

$$f_{(n)}(s) = f_n(s)$$

where

$$f_n(s) = f[f_{n-1}(s)] \quad \text{and} \quad f_1(s) = f(s) .$$

If $0 < p_0 < 1$, then $f(s)$ is strictly convex on the unit interval and the sequence $\{f_n(0)\}$ is strictly increasing.

Each branching process is characterized by a parameter q called the *extinction probability* given by the least non-negative root of the equation $f(s) = s$. q stands for the probability that after some finite number of generations the family will cease to reproduce, i.e., $Z_n \to 0$. Naturally, $1-q$ is called the probability of *immortality*.

The influence of m on q is illustrated in Fig. 4.

It is shown that

(1) if $m < 1$, then $q = 1$ and $f'(q) < 1$;

(2) if $m = 1$ and $p_1 \neq 1$, then $q = 1$ and $f'(q) = 1$;

(3) if $m > 1$, then $q < 1$ and $f'(q) < 1$.

Moreover, it can be shown that if $m = 1$ and $\sigma^2 < \infty$, then $\mathrm{P}[Z_n > 0] \sim 2/\sigma^2 n$.

Appendix B. The Expected Size of an Extinct Family

We show that the expected size of an extinct family is finite iff $m \neq 1$. By extinction we mean the event that some generation is empty, i.e., $\bigcup_{i=1}^{\infty}(Z_i = 0)$.

Lemma B.1.

$$\mathrm{E}\left[\sum_{i=0}^{\infty} Z_i \;\middle|\; \bigcup_{i=1}^{\infty}(Z_i = 0)\right] = \frac{1}{1-f'(q)}.$$

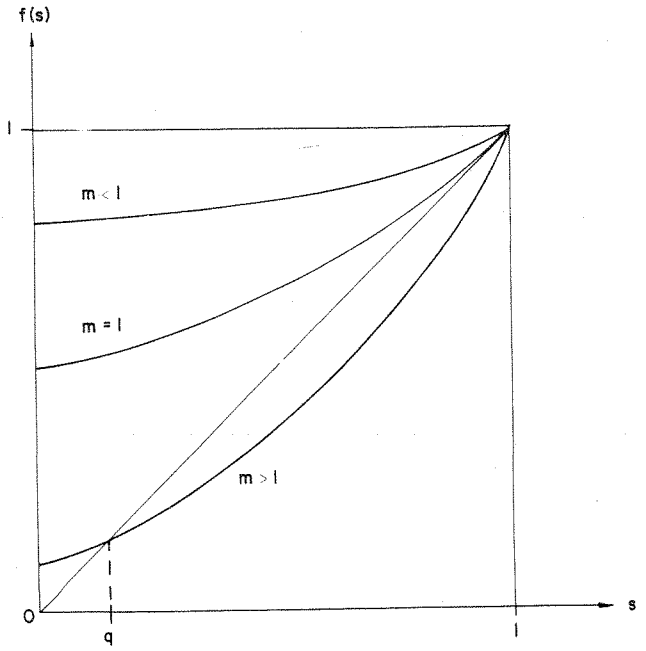

FIG. 4. The effect of the reproduction rate m on the shape of the generating function $f(s)$ and on the probability of extinction q.

Proof (by Bayes theorem).

$$P\left[Z_1 = k \mid \bigcup_{i=1}^{\infty} (Z_i = 0)\right] = P[Z_1 = k] \frac{P[\bigcup_{i=1}^{\infty}(Z_i = 0) \mid Z_1 = k]}{P[\bigcup_{i=0}^{\infty}(Z_i = 0)]}$$

$$= p_k q^k / q = p_k q^{k-1} .$$

Thus,

$$E\left[Z_1 \mid \bigcup_{i=1}^{\infty} (Z_i = 0)\right] = \sum_{k=0}^{\infty} k p_k q^{k-1} = f'(q) .$$

Similarly,

$$E\left[Z_n \mid \bigcup_{i=1}^{\infty} (Z_i = 0)\right] = (f'(q))^n$$

and

$$E\left[\sum_{i=0}^{\infty} Z_i \mid \bigcup_{i=1}^{\infty} (Z_i = 0)\right] = \frac{1}{1 - f'(q)} .$$

The result now follows since

$$f'(q) < 1 \quad \text{iff} \quad m \neq 1 .$$

Obviously, Lemma B.1 does not hold for the case $m = 1$, where $f'(1) = 1$ and so, the expected size of an extinct family becomes infinite. It is possible to show, however, that in this case the expected family size is proportional to the depth of the shallowest generation for which extinction can be certified, i.e.,

$$E\left(\sum_{i=0}^{N-1} Z_i \mid Z_N = 0\right) \sim N/3 .$$

Appendix C. Proof of Theorem 3.2

Theorem 3.2 follows from a result of Bramson [5] concerning branching random walks. We present here a short proof, in the terminology of the present paper, by reformulating the theorem in terms of infinite trees. Consider an infinite uniform binary tree in which each edge has cost 0 with probability $\frac{1}{2}$ and cost 1 with probability $\frac{1}{2}$. Let $C(N, \frac{1}{2})$ be the minimum cost of a path from the root to a node at level N. Call a node an ith *generation frontier node* if its cost is i and its father's cost is $i - 1$, and let Z_i be the number of ith generation frontier nodes. Note that the level of each ith generation frontier node is at most Z_i.

We prove below that the number of ith generation frontier nodes tends to be approximately 2^{2^i}. More precisely,

$$P[Z_i + 1 \leq (2^{2^i})^x] \text{ approaches a limit } v(x) , \qquad (*)$$

where $v(x) \to 1$ as $x \to \infty$. Assuming this result we proceed as follows. Observe that the event $C(N, \frac{1}{2}) \leq i$ implies $Z_i \geq N$ because the former means that there exists an ith generation frontier node at level at least N. But, by the above claim

$$P[Z_{\log_2\log_2 N - H(N)} \geq N] \to 0 ,$$

and hence,

$$P[C(N, \tfrac{1}{2}) \leq \log_2\log_2 N - H(N)] \to 0 .$$

On the other hand, the existence of a zero-cost path of length N starting at an ith generation frontier node is sufficient to insure that $C(N, \frac{1}{2}) \leq i$. The probability that a particular ith generation frontier node is the origin of such a path is $2/\sigma^2 N + O(1/N)$, and thus, the probability that such a path fails to exist from any of the Z_i frontier nodes is

$$\left(1 - \frac{2}{\sigma^2 N} - O\left(\frac{1}{N}\right)\right)^{Z_i} .$$

In the case $i = \log_2\log_2 N + H(N)$, $P[Z_i > N^2] \to 1$ as $N \to \infty$ and hence the probability that such a path fails to exist goes to zero as $N \to \infty$. Hence,

$$P[C(N, \tfrac{1}{2}) > \log_2 \log N + H(N)] \to 0 \quad \text{as} \quad N \to \infty .$$

It remains to prove the claim (*). The random variables Z_i are determined by a branching process $\{p_k\}$ where $p_0 = p_1 = 0$, $p_2 = \frac{1}{4}$ and

$$p_k = \tfrac{1}{2} p_{k-1} + \tfrac{1}{4} \sum_{j=0}^{k} p_j p_{k-j}, \quad k \geq 3 .$$

The generating function $f(s)$ for this process satisfies $f(s) - \frac{1}{4}s^2 = \frac{1}{2}sf(s) + \frac{1}{4}f^2(s)$, whence $f(s) = 2 - s - 2\sqrt{1-s}$ and $m = \infty$.

We now invoke the following lemma.

Lemma C.1 [4]. *Consider a branching process with $m = \infty$ and $f(s) = \sum p_k s^k$. If $g(x)$, the inverse function of $1 - f(1-s)$, satisfies $g'(x) = ax^{b-1}(1 + O(x^\delta))$ as $x \to 0$ for some $a > 0$, $b > 1$, $\delta > 0$, then for every x $P[b^{-n}\log(Z_n + 1) \leq x]$ approaches a limit $v(x)$ as $n \to \infty$.*

In the present case $g(x) = 2 - x - 2\sqrt{1-x}$ and $g'(x) = \frac{1}{2}x(1 + O(x))$ as $x \to 0$. The conditions of Lemma C.1 are satisfied with $a = \frac{1}{2}$, $b = 2$, and the claim is proven.

ACKNOWLEDGMENT

The authors thank R. Arratia for reviewing the manuscript and pointing out the result of Bramson [5].

REFERENCES

1. Harris, T., *The Theory of Branching Processes* (Springer, Berlin, 1963).
2. Peterson, W.W., *Error Correcting Codes* (MIT Press, Cambridge, MA, and Wiley, New York, 1961) Appendix A.
3. Nilsson, N.J., *Problem-Solving Methods in Artificial Intelligence* (McGraw-Hill, New York, 1971).
4. Karp, R., The probabilistic analysis of some combinatorial search algorithms, in: J.F. Traub (Ed). *Algorithms and Complexity* (Academic Press, New York, 1976) 1–19.
5. Bramson, M.D., Minimal displacement of branching random walk, *Z. Wahrsch. Verw. Gebiete* **45** (1978) 89–108.
6. Darling, D.A., The Galton–Watson process with infinite mean, *J. Appl. Probab.* 1(7) (1970) 455–456.

Received July 1982; revised version received October 1982