

HEURISTIC SAMPLING: A METHOD FOR PREDICTING THE PERFORMANCE OF TREE SEARCHING PROGRAMS*

PANG C. CHEN†

Abstract. Determining the feasibility of a particular search program is important in practical situations, especially when the computation involved can easily require days, or even years. To help make such predictions, a simple procedure based on a stratified sampling approach is presented. This new method, which is called heuristic sampling, is a generalization of Knuth's original algorithm for estimating the efficiency of backtrack programs. With the aid of simple heuristics, this method can produce significantly more accurate cost estimates for commonly used tree search algorithms such as depth-first, breadth-first, best-first, and iterative-deepening.

Key words. heuristics, stratified sampling, search tree, feasibility testing, cost estimation, Monte Carlo method, analysis of algorithms

AMS(MOS) subject classifications. 68Q25, 65C05

1. Introduction. Tree searching [8] is a general, easily implemented problem-solving technique. Unfortunately, the efficiency of tree searching programs is usually difficult to analyze, even at a rudimentary level. Without analytic cost information, the typical course is to let the computer run until it either finishes the job or exhausts our patience. Switching to a more computational sampling approach provides a less haphazard alternative: We gain accuracy in understanding particular search programs and thus design better ones.

The sampling method that we will discuss generalizes an algorithm of Knuth [5] for estimating properties of a backtrack tree. Starting at the root, Knuth's algorithm extends a partial path by expanding the end node and picking a child according to a uniform distribution. It then forms an unbiased estimate of the tree property using the branching degrees along the randomly selected path. According to Knuth, this simple estimation procedure worked consistently well in his experiments. But as a refinement, Knuth also suggested the technique of importance sampling [2] in which the child is selected according to a weighted distribution, with the weight of each child being an estimate of the property of the corresponding subtree.

Unsatisfied with the results of Knuth's algorithm in his own experiments, Purdom [9] later modified the algorithm in an attempt to reduce variance. The modified algorithm allows more than one child to be considered for further exploration, and it is therefore called partial backtracking. If only one child per parent is explored, then Purdom's algorithm reduces to Knuth's algorithm; but if every child is explored, then the estimate becomes exact due to a complete backtrack search. The algorithm offers a varying degree of efficiency that depends on the extent of exploration. However, it is difficult to reduce variance significantly without incurring high experimental cost.

Heuristic sampling generalizes Knuth's algorithm in a different direction. Instead of taking the importance sampling approach, the method adopts another statistical technique known as stratified sampling [2], based on a "heuristic function" to be supplied by the algorithm designer. This heuristic function, which we call a stratifier, should in principle reflect the broad characteristics of the nodes in the search tree. If two nodes have similar features, then they should be classified into the same stratum. By exploiting the tree structure as reflected through the stratifier, stratified sampling on the search tree

*Received by the editors April 16, 1990; accepted for publication May 4, 1991. The work was performed at Stanford University, where it was supported by a Hertz Fellowship, and at Sandia National Laboratories, where it was supported by U.S. Department of Energy contract DE-AC04-76DP00789.

†Sandia National Laboratories, Albuquerque, New Mexico 87185.

reduces variance relative to Knuth's algorithm, and limits the cost of prediction to the number of strata.

To present heuristic sampling, we first introduce the concept of a stratifier, and then discuss a sampling scheme that employs stratifiers to estimate tree properties. We examine the accuracy and cost of the sampling scheme both analytically and empirically. In particular, we demonstrate the effectiveness of heuristic sampling analytically with a random graph problem, and empirically with a chessboard recreation.

2. Problem formulation. Let T be a tree with nodes (or states) S , and let f be an arbitrary function on S . Then, abstractly, the goal of heuristic sampling is to produce estimates of the sum

$$(1) \quad \varphi \stackrel{\text{def}}{=} \sum_{s \in S} f(s)$$

without traversing the entire tree. The value of φ is a property of T , as captured by the definition of f . For instance, if $f(s)$ is the cost of processing node s , then φ becomes the total cost of traversing T ; if $f(s)$ is the number of children of s , then φ becomes the total number of edges in T ; and if f is an indicator function for some property $\mathcal{X}$, then φ becomes the number of nodes satisfying $\mathcal{X}$.

The performance of some alternative heuristic search algorithms can be studied by estimating φ , using an appropriate choice of f . For example, suppose that F is a monotone heuristic cost function [8], and that $\mathcal{X}$ represents the property $F(s) \leq F_0$. Then estimating $\varphi(F_0)$ predicts the number of nodes that a best-first search will have to examine before it can process any other node with cost $> F_0$. Likewise, estimating the sum $\sum_{F_1 \leq F_0} \varphi(F_1)$ predicts the respective complexity for iterative deepening [6].

3. Heuristic sampling. Our sampling technique requires a stratifier: a heuristic function $h : S \rightarrow \mathcal{P}$ that maps each state s to a stratum $h(s)$ in a partially ordered set $\mathcal{P}$, with h strictly decreasing along each edge of T . (Notice that we can always construct a stratifier by concatenating the depth of the node along with any other state information into a vector representing the stratum, and using a lexicographic ordering to compare these strata.) Given that T is stratified under h , our idea is to sample each stratum α for a representative node s_α and simultaneously to obtain an estimate $w_\alpha = w(s_\alpha)$ for the number of nodes in that stratum. From the sample set of nodes, an unbiased estimate of φ can then be obtained by computing

$$(2) \quad \hat{\varphi} \stackrel{\text{def}}{=} \sum_{\alpha} w_{\alpha} f(s_{\alpha}).$$

Figure 1 gives an algorithm (HS) for finding the sample nodes and weights needed to compute $\hat{\varphi}$. The algorithm explores at most one node for each stratum and pretends that all other nodes in the stratum have the same subtree structure. It initializes the weight of the root to 1, and inserts the node-weight pair into a queue for further exploration. The queue is maintained with at most one representative from each stratum. The procedure is to remove and expand the node in the front of the queue (the one in a maximal stratum) until the queue becomes empty. To prevent multiple representatives, each child is given a chance to replace the current representative in the queue. The random replacement is implemented in such a way that a child will be expanded with probability proportional to the weight of its parent.

Both the accuracy and the time complexity of the sampling scheme depend on the stratifier. The variance of the estimator depends on the homogeneity of each stratum,

Algorithm HS

input: root of a tree and a stratifier h ;
output: set of sample nodes S and corresponding weights W ;
data structure: a queue Q with elements (s, w) indexed by $h(s)$;
begin
 $Q \leftarrow \{(\text{root}, 1)\};$
while (Q not empty) **do**
 output an element (s, w) of Q with maximal $h(s)$;
 for each child t of s **do**
 $\alpha \leftarrow h(t);$
 if Q contains an element (s_α, w_α) in stratum α **then do**
 $w_\alpha \leftarrow w_\alpha + w;$
 with probability w/w_α **do**
 $s_\alpha \leftarrow t;$
 else
 insert a new element (t, w) into Q ;
end.

FIG. 1. *Heuristic sampling.*

which we will discuss in the analysis section. As for the sampling time, notice that we can guarantee an output of at most one representative node per stratum because we choose to process from the higher stratum down. As a result, the number of nodes that need to be expanded by the sampling scheme is bounded by the number of strata.

3.1. An example. Consider the simple problem of determining the number of paths from (m, n) to $(0, 0)$ on the Cartesian plane, where the only legal moves are to the left or downward in steps of unit length. This problem is well known and the exact answer is $\binom{m+n}{m}$. Nevertheless, suppose that we do not know how to solve the problem analytically. Then a naïve approach is to enumerate a tree with each node corresponding to a partial legal path starting (m, n) , and each edge corresponding to a legal move. In this tree, the number of leaves is the desired answer. To estimate, apply heuristic sampling with f being 1 on each leaf, and zero everywhere else. If we stratify according to

$$(3) \quad h_0(s) = -\text{depth}(s),$$

then the sampling procedure becomes Knuth's algorithm. Observe that in the absence of additional knowledge, we can always try this depth stratification. In fact, we can also try

$$(4) \quad g_0(s) = \langle -\text{depth}(s), \text{children}(s) \rangle,$$

which can perhaps give us a better stratification. Still, for our problem, there is a much better stratifier that exploits additional knowledge; we can stratify according to

$$(5) \quad h(s) = \langle i, j \rangle$$

with $\langle i, j \rangle$ being the end point of the partial path corresponding to s . Under this stratification, the estimator $\hat{\varphi}$ is exact (unbiased with zero variance), because for each stratum the number of leaves in each subtree is the same. Therefore, the tree size $\binom{m+n}{m}$ is estimated exactly by expanding only $(m+1)(n+1)$ nodes, even though $\binom{m+n}{m}$ is exponentially large.

3.2. Adding importance sampling. We will show that $\hat{\varphi}$ remains unbiased when we alter the child selection probability by replacing the clause

$$\begin{aligned} w_\alpha &\leftarrow w_\alpha + w; \\ \textbf{with probability } w/w_\alpha &\textbf{ do} \\ s_\alpha &\leftarrow t; \end{aligned}$$

in heuristic sampling (HS) with

$$\begin{aligned} w_\alpha &\leftarrow w_\alpha/(1-p); \\ \textbf{with probability } p &\textbf{ do} \\ s_\alpha &\leftarrow t; \\ w_\alpha &\leftarrow w/p; \end{aligned}$$

using an arbitrary importance probability $0 < p < 1$ in each iteration. We call the resulting generic sampling algorithm GS. As with Knuth's algorithm [5], there exists a sequence of p 's that will force $\hat{\varphi}$ to be exactly equal to φ , provided that f is nonnegative. In practice, however, this sequence may be difficult to find.

4. Analysis. We will now examine the behavior of HS, GS (which incorporates importance sampling), and simple sampling (SS) of Knuth [5] that uses depth as the stratifier. Because GS is a generalization of HS, which is in turn a generalization of SS, anything we prove for GS also holds for HS, and anything that is true for HS is also true for SS.

4.1. Preliminaries. Let $\alpha_0 = h(\text{root})$ be the maximum stratum and denote the parent of s by $\hat{s}$. From the definition of a stratifier, we know that the set of expanded nodes S in the output can have at most one node per stratum; hence we can write

$$(6) \quad S = \bigcup_{\alpha \in \mathcal{P}} \{s_\alpha\}$$

with s_α being the unique representative node for stratum α , if it exists. Let W be the set of weights produced along with S , and let $w(s_\alpha) = w_\alpha$. To prevent any ambiguity in selecting a maximal element of Q , we assume that the strata are processed according to a total ordering $\succ$, consistent with $\mathcal{P}$. Later we show that the output of HS is statistically invariant under any particular total ordering of $\mathcal{P}$. Thus, define

$$(7) \quad S_\alpha \stackrel{\text{def}}{=} \bigcup_{\beta \succ \alpha} \{s_\beta\}$$

to be the intermediate output before processing stratum α , and denote the substratum of α (the one immediately below) by $\underline{\alpha}$.

4.2. Sample trees. According to GS, every intermediate output S_α must form a subtree T_α because for any nonroot node to be inserted into Q , its parent must have already been expanded. In particular, the final output S must also form a subtree T , which we call a sample tree.

From the sampling procedure, we know that a sample tree T must contain the parent of every nonroot node of T and a stratum representative for every child of every nonleaf

node of T . Conversely, any subtree T of $\mathbf{T}$ satisfying the condition above must also have a positive probability of being generated by GS. The output probability $\mathbf{P}(T)$ associated with each instance of T is related directly to the weights of the nodes produced.

THEOREM 1. *The probability that GS generates sample tree T with weights W is*

$$(8) \quad \mathbf{P}(T) = \prod_{(s,t) \in T} \frac{w(s)}{w(t)}.$$

Under HS, these weights satisfy the recurrence

$$(9) \quad w_{\alpha_0} = 1; \quad w_\alpha = \sum_{\beta \succ \alpha} w_\beta c_\alpha(s_\beta) \quad \text{for } \alpha \prec \alpha_0,$$

where $c_\alpha(s)$ denotes the number of children of s in stratum α .

Proof. Given T_α , the probability of leaving stratum α with $s_\alpha = t$ is equal to $p \prod_i q_i$, where p is the probability of choosing t to replace the current queue element, and q_i is the probability of not making the i th replacement of t with t being in the queue. These probabilities are just ratios of weights by construction, and their product telescopes to $w(\hat{t})/w(t)$. On the other hand, if s_α does not exist, then $T_\alpha = T_\alpha$ and $\mathbf{P}(s_\alpha | T) = 1$ vacuously. Hence, unfolding the recurrence

$$(10) \quad \mathbf{P}(T_\alpha) = \mathbf{P}(T_\alpha) \mathbf{P}(s_\alpha | T_\alpha)$$

yields the desired product for $\mathbf{P}(T)$.

Under HS, the weight w_α for $\alpha \prec \alpha_0$ is simply equal to the sum of $w(\hat{t})$ for all t with $h(t) = \alpha$ and $\hat{t}$ in T , a sum which is clearly equivalent to the desired formula. $\square$

COROLLARY 2. *The output probability of a sample tree of HS is independent of the total ordering imposed on $\mathcal{P}$. Consequently, which maximal element we choose to expand at each stage has no effect on the distribution of $\hat{\varphi}$.*

Proof. The recurrence governing the weights is invariant under different total orderings of $\mathcal{P}$. $\square$

For the SS case, every sample tree is just a (sample) path $\rho(s)$ from the root to a leaf s .

COROLLARY 3. *Under SS, the probability of producing a sample path $T = \rho(s)$ with length k is*

$$(11) \quad \mathbf{P}(\rho(s)) = 1/w(s) = 1 / \prod_{0 < j \leq k} c(s^j),$$

where $c(s)$ denotes the number of children of s , and s^j denotes the j th ancestor of s , with $s^1 = \hat{s}$.

4.3. Unbiasedness of $\hat{\varphi}$. The unbiasedness of $\hat{\varphi}$ as produced from GS can be proved quite easily with concepts from martingale theory [4]. During the expansion of s_α , define $Q_{\alpha,k}$ to be the set of representatives in the queue before considering the k th child of s_α , and define $C_{\alpha,k}$ to be those children of s_α yet to be considered. Also define $\mathbf{T}(s)$ to be the induced subtree of s in $\mathbf{T}$, and define

$$(12) \quad \Phi(s) \stackrel{\text{def}}{=} \sum_{t \in \mathbf{T}(s)} f(s).$$

THEOREM 4. *The estimator $\hat{\varphi}$ under GS is unbiased in that*

$$(13) \quad \mathbf{E}\hat{\varphi} = \Phi(\text{root}) = \varphi.$$

Proof. The appropriate statistic to study at each iteration of GS is the successive approximation

$$(14) \quad \varphi^{(\alpha,k)} \stackrel{\text{def}}{=} \sum_{u \in Q_{\alpha,k}} w(u)\Phi(u) + \sum_{t \in C_{\alpha,k}} w_{\alpha}\Phi(t) + \sum_{\beta \succ \alpha} w_{\beta}f(s_{\beta}),$$

which is completely determined when given the data

$$(15) \quad Z_{\alpha,k} \stackrel{\text{def}}{=} S_{\alpha} \cup Q_{\alpha,k} \cup C_{\alpha,k}.$$

If $k \leq c(s_{\alpha})$ and the k th child t of s_{α} is supposed to replace the current node u in the queue with probability $p = 1 - q$, then the expected value of the successive approximation remains the same after the random replacement, for we have

$$(16) \quad \mathbf{E}(\varphi^{(\alpha,k+1)} - \varphi^{(\alpha,k)} \mid Z_{\alpha,k}) = p \frac{w_{\alpha}}{p} \Phi(t) + q \frac{w(u)}{q} \Phi(u) - w_{\alpha}\Phi(t) - w(u)\Phi(u) = 0.$$

On the other hand, if $k > c(s_{\alpha})$ and node u in the queue is supposed to be removed and expanded next, then the successive approximation also remains the same after the expansion of u , for we have

$$(17) \quad \mathbf{E}(\varphi^{(\alpha,1)} - \varphi^{(\alpha,k)} \mid Z_{\alpha,k}) = w(u)f(u) + w(u) \sum_{(u,t) \in T} \Phi(t) - w(u)\Phi(u) = 0.$$

Consequently, the expected value of our successive approximation at the end of the sampling ($\hat{\varphi}$) is equal to its initial value ($\Phi(\text{root}) = \varphi$). $\square$

4.4. Conditions for minimum variance. If f is nonnegative, then in the preceding proof we can choose to replace the current node u with the k th child t of s with probability

$$(18) \quad p = w(s)\Phi(t)/(w(s)\Phi(t) + w(u)\Phi(u))$$

to ensure $\varphi^{(\alpha,k+1)} = \varphi^{(\alpha,k)}$. The resulting sequence of p 's is exactly the one we need to reduce variance of $\hat{\varphi}$ to zero. Of course, computing this perfect sequence is not feasible in practice. But if we model our heuristic stratifier by assuming that the subtrees corresponding to each stratum are drawn from the same distribution of trees, then we can show that the probability $p = w(s)/(w(s) + w(u))$ used in HS is indeed the one that minimizes the average variance of $\hat{\varphi}$.

THEOREM 5. *Suppose that f is nonnegative, and that we are running GS on a random tree T in which the subtrees corresponding to each stratum are drawn from the same distribution. Then, $\mathbf{EV}(\hat{\varphi} \mid T)$, the average variance of $\hat{\varphi}$, is minimized if at every random replacement stage, we choose to replace the current node u with a child of s with probability $p = w(s)/(w(s) + w(u))$.*

Proof. Consider the effect of varying p for one particular random replacement stage and leaving everything else fixed. We have

$$(19) \quad \hat{\varphi} = \begin{cases} Aw(s)/p + B & \text{with probability } p, \\ Aw(u)/(1-p) + B & \text{with probability } 1-p, \end{cases}$$

where $A \geq 0$ and B are independent of p . Hence, the variance of $\hat{\varphi}$ can be written as

$$(20) \quad A'(w^2(s)/p + w^2(u)/(1-p)) + B',$$

with $A' \geq 0$ and B' independent of p . Taking the average over all random trees and differentiating with respect to p yields the formula

$$(21) \quad -A''(w^2(s)/p^2 - w^2(u)/(1-p)^2)$$

with $A'' \geq 0$. Setting this derivative to zero, we find that p has to be proportional to $w(s)$, implying that $p = w(s)/(w(s) + w(u))$. The nonnegativity of the second derivative shows that this condition does indeed minimize the average variance. $\square$

4.5. Variance of $\hat{\varphi}$. To derive a formula for $\mathbf{V}\hat{\varphi}$, we use the superscript (α) on an operator to denote the conditional operator given S_α . We also use ∂S_α to denote the children of S_α in stratum $\preceq \alpha$, and use $R_\alpha = R(S_\alpha)$ to denote the nodes of ∂S_α in stratum α . Notice that R_α contains precisely the candidates for s_α , and depending on which $t \in R_\alpha$ is selected as s_α , the weight w_α is equal to $w(\hat{t})$ divided by $\mathbf{P}^{(\alpha)}(t)$, the probability of choosing t .

For $\alpha \prec \alpha_0$, consider the successive approximation

$$(22) \quad \varphi^{(\alpha)} \stackrel{\text{def}}{=} \sum_{\beta \succ \alpha} w_\beta f(s_\beta) + \sum_{s \in \partial S_\alpha} w(\hat{s}) \Phi(s),$$

which depends only on S_α . At the initial stage when α corresponds to $\underline{\alpha}_0$, we have $S_\alpha = \{\text{root}\}$ and $\varphi^{(\alpha)} = \varphi$; at the final stage when α corresponds to the lowest stratum, we have $S_\alpha = S$ and $\varphi^{(\alpha)} = \hat{\varphi}$. Because of this link, we can characterize the behavior of $\hat{\varphi}$ by studying the changes in $\varphi^{(\alpha)}$ as α moves through $\mathcal{P}$.

LEMMA 6.

$$(23) \quad \mathbf{E}^{(\alpha)} \varphi^{(\underline{\alpha})} = \varphi^{(\alpha)}.$$

Proof. The set $\partial S_{\underline{\alpha}}$ is obtained from ∂S_α by removing R_α and adding the children of s_α ; hence

$$(24) \quad \varphi^{(\underline{\alpha})} - \varphi^{(\alpha)} = w_\alpha f(s_\alpha) - \sum_{t \in R_\alpha} w(\hat{t}) \Phi(t) + \sum_{(s_\alpha, v) \in T} w(s_\alpha) \Phi(v)$$

$$(25) \quad = w_\alpha \Phi(s_\alpha) - \sum_{t \in R_\alpha} w(\hat{t}) \Phi(t).$$

But we also know that

$$(26) \quad \mathbf{E} w_\alpha \Phi(s_\alpha) = \sum_{t \in R_\alpha} \mathbf{P}^{(\alpha)}(t) \frac{w(\hat{t})}{\mathbf{P}^{(\alpha)}(t)} \Phi(t).$$

Therefore, we have $\mathbf{E}^{(\alpha)}(\varphi^{(\underline{\alpha})} - \varphi^{(\alpha)}) = 0$. $\square$

LEMMA 7.

$$(27) \quad \mathbf{V} \varphi^{(\underline{\alpha})} = \mathbf{V} \varphi^{(\alpha)} + \mathbf{E} \mathbf{V}^{(\alpha)} w_\alpha \Phi(s_\alpha).$$

Proof. Conditioning on S_α , we have

$$(28) \quad \mathbf{V}\varphi^{(\underline{\alpha})} = \mathbf{V}\mathbf{E}^{(\alpha)}\varphi^{(\underline{\alpha})} + \mathbf{E}\mathbf{V}^{(\alpha)}\varphi^{(\underline{\alpha})}$$

$$(29) \quad = \mathbf{V}\varphi^{(\alpha)} + \mathbf{E}\mathbf{V}^{(\alpha)}\varphi^{(\underline{\alpha})}$$

according to the previous lemma. The desired result then follows since

$$(30) \quad \mathbf{V}^{(\alpha)}\varphi^{(\underline{\alpha})} = \mathbf{V}^{(\alpha)}(\varphi^{(\underline{\alpha})} - \varphi^{(\alpha)})$$

$$(31) \quad = \mathbf{V}^{(\alpha)}w_\alpha\Phi(s_\alpha). \quad \square$$

THEOREM 8. *Under GS,*

$$(32) \quad \mathbf{V}\hat{\varphi} = \sum_{\alpha \prec \alpha_0} \mathbf{E}\mathbf{V}^{(\alpha)}w_\alpha\Phi(s_\alpha).$$

Proof. Simply unfold the recurrence of the lemma above, and identify $\varphi^{(\underline{a})}$ with $\hat{\varphi}$, assuming that a is the lowest stratum in $\mathcal{P}$. $\square$

For HS, the weight $w_\alpha = \sum_{t \in R_\alpha} w(\hat{t})$ is independent of the choice made on s_α , so we can move w_α out of $\mathbf{V}^{(\alpha)}$ and obtain the following theorem.

THEOREM 9. *Under HS,*

$$(33) \quad \mathbf{V}\hat{\varphi} = \sum_{\alpha \prec \alpha_0} \mathbf{E}w_\alpha^2\mathbf{V}^{(\alpha)}\Phi(s_\alpha).$$

For SS, the weight $w(s)$ is simply the product of branching degrees of the ancestors of s .

THEOREM 10. *Under SS,*

$$(34) \quad \mathbf{V}\hat{\varphi} = \sum_{s \in \mathcal{S}} c^2(s)w(s)V(s),$$

where $V(s)$ denotes the mean-square deviation of the Φ value of its children from the average Φ value of its children.

Proof. If $T_\alpha = \rho(s)$, then $\mathbf{P}(T_\alpha) = 1/w(s)$, and $w_\alpha = w(s)c(s)$, implying that

$$(35) \quad \mathbf{V}\hat{\varphi} = \sum_{s \in \mathcal{S}} \frac{1}{w(s)} w^2(s) c^2(s) V(s). \quad \square$$

4.6. Homogeneity of strata. The intuitive qualification of a good stratifier h is that it should provide some degree of homogeneity among nodes in the same stratum. We formalize this concept of homogeneity by introducing a measure on its counterpart. For each stratum α , define the parameter

$$(36) \quad \tilde{\nu}_\alpha \stackrel{\text{def}}{=} \max_{S_\alpha} \frac{\mathbf{V}^{(\alpha)}\Phi(s_\alpha)}{(\mathbf{E}^{(\alpha)}\Phi(s_\alpha))^2},$$

and call the parameters collectively as the degree of variation $\tilde{\nu}$ with respect to Φ under h . Notice that the sequence $\tilde{\nu}$ is always nonnegative, with the zero sequence indicating total homogeneity within each stratum.

The following theorem demonstrates a relationship between the degree of homogeneity and the accuracy of $\hat{\varphi}$.

THEOREM 11. *Suppose that f is nonnegative and that the tree $\mathbf{T}$ under stratifier h has degree of variation $\tilde{\nu}$ with respect to Φ . Then*

$$(37) \quad \mathbf{V}\hat{\varphi} \leq \left(\prod_{\alpha \prec \alpha_0} (1 + \tilde{\nu}_\alpha) - 1 \right) \varphi^2$$

under HS.

Proof. From the proof of Lemma 7, we know that

$$(38) \quad \mathbf{V}^{(\alpha)}\varphi(\underline{\alpha}) = w_\alpha^2 \mathbf{V}^{(\alpha)}\Phi(s_\alpha),$$

which is by definition less than or equal to $\tilde{\nu}_\alpha w_\alpha^2 (\mathbf{E}^{(\alpha)}\Phi(s_\alpha))^2$. On the other hand, we also know that

$$(39) \quad \mathbf{E}^{(\alpha)}w_\alpha\Phi(s_\alpha) = \sum_{t \in R_\alpha} w(t)\Phi(t) \leq \varphi^{(\alpha)},$$

because $R_\alpha \subseteq \partial S_\alpha$. Hence, we have

$$\begin{aligned} \mathbf{E}^{(\alpha)}(\varphi(\underline{\alpha}))^2 &= (\mathbf{E}^{(\alpha)}\varphi(\underline{\alpha}))^2 + \mathbf{V}^{(\alpha)}\varphi(\underline{\alpha}) \\ &\leq (\mathbf{E}^{(\alpha)}\varphi(\underline{\alpha}))^2 + \tilde{\nu}_\alpha(\varphi^{(\alpha)})^2 \\ &= (1 + \tilde{\nu}_\alpha)(\varphi^{(\alpha)})^2, \end{aligned}$$

since $\mathbf{E}^{(\alpha)}\varphi(\underline{\alpha}) = \varphi^{(\alpha)}$. Consequently, we have the inequality

$$(40) \quad \mathbf{E}(\varphi(\underline{\alpha}))^2 \leq (1 + \tilde{\nu}_\alpha)\mathbf{E}(\varphi^{(\alpha)})^2.$$

Unfolding this inequality and identifying $\varphi(\underline{\alpha_0})$ with φ then yields the theorem. $\square$

An immediate consequence of this theorem is that if the parameters $\tilde{\nu}_\alpha$ are all fairly small, say, less than some quantity ϵ , with $\epsilon \|h(\mathbf{S})\| = o(1)$, then the normalized variance $\mathbf{V}\hat{\varphi}/\varphi^2$ can be at most

$$(41) \quad \epsilon \|h(\mathbf{S})\| + O(\epsilon \|h(\mathbf{S})\|)^2.$$

However, if ϵ is not small enough, then the bound can grow exponentially in the number of strata. Fortunately, with more careful bookkeeping, we can obtain a sharper bound in which the growth is only exponential in the depth of the tree, rather than in the total number of strata.

To derive this bound, we introduce the concept of layered poset partitioning. A poset partition $\mathcal{P} = \bigcup_i A_i$ is said to be layered if the A_i 's are disjoint antichains with $A_i \prec A_j$ for all $j < i$. In other words, every pair of strata in A_i should be incomparable (in the sense that no member of one stratum is a descendent of another in the other stratum) and no stratum in A_i should be higher than any stratum in A_j for $j < i$. Think of A_j as the set of strata with depth j . With respect to our previous enforcement of a total ordering on $\mathcal{P}$, each A_i is simply a stratum, and each stratum are processed in the corresponding order. Under the more general layered partitioning of $\mathcal{P}$, each antichain is also processed in order. But because every pair of strata in an antichain is incomparable, the corresponding strata are processed independently. Exploiting this property yields the following generalization.

THEOREM 12. Suppose that f is nonnegative and that the state space $\mathcal{S}$ under the stratifier has degree of variation $\tilde{\nu}$ with respect to Φ . Then the variance of $\hat{\varphi}$ under HS is bounded above by

$$(42) \quad \left(\prod_{0 < i \leq r} \left(1 + \max_{\alpha \in A_i} \tilde{\nu}_\alpha \right) - 1 \right) \varphi^2,$$

where $\bigcup_{0 \leq i \leq r} A_i$ is an arbitrary layered partition of $\mathcal{P}$.

Proof. We take the same approach as in the previous theorem. However, instead of working with random variables conditioning on S_α , we now condition them on

$$(43) \quad S_i \stackrel{\text{def}}{=} \bigcup_{j < i} \bigcup_{\alpha \in A_j} \{s_\alpha\},$$

and use the superscript (i) on an operator to denote the conditional operator given S_i . Thus, consider the conditional random variable

$$(44) \quad \varphi_i \stackrel{\text{def}}{=} \mathbf{E}^{(i)} \hat{\varphi} = \sum_{s \in S_i} w(s) f(s) + \sum_{s \in \partial S_i} w(\hat{s}) \Phi(s),$$

where ∂S_i , analogous to ∂S_α , is defined to be the children of S_i in strata lower than those in S_i . Notice that as we extend the conditional information from S_i to S_{i+1} , the only part of $\hat{\varphi}$ that varies is the sum $\sum_{\alpha \in A_i} w_\alpha \Phi(s_\alpha)$. This result follows because for each α in A_i , the parents of the prospective nodes for s_α are all in S_i . Incidentally, this fact tells us that the weight $w(s_\alpha)$ depends solely on S_i , and not on $S_{i+1} \setminus S_i$. Hence, we have

$$(45) \quad \mathbf{V}^{(i)} \varphi_{i+1} = \mathbf{V}^{(i)} \sum_{\alpha \in A_i} w_\alpha \Phi(s_\alpha) = \sum_{\alpha \in A_i} w_\alpha^2 \mathbf{V}^{(i)} \Phi(s_\alpha),$$

since each s_α is selected independently. Applying the definition of $\tilde{\nu}$, we obtain the upper bound

$$(46) \quad \sum_{\alpha \in A_i} \tilde{\nu}_\alpha w_\alpha^2 (\mathbf{E}^{(i)} \Phi(s_\alpha))^2.$$

But

$$(47) \quad w_\alpha^2 (\mathbf{E}^{(i)} \Phi(s_\alpha))^2 = \left(\sum_{t \in R_\alpha(S_i)} w(\hat{t}) \Phi(t) \right)^2,$$

with $R_\alpha(S_i)$ being those children in ∂S_i in stratum α . Again, because the strata in A_i are incomparable, the sets $R_\alpha(S_i)$ are disjoint between different α 's. This conclusion yields the bound

$$(48) \quad \sum_{\alpha \in A_i} w_\alpha^2 (\mathbf{E}^{(i)} \Phi(s_\alpha))^2 \leq \left(\sum_{t \in R_i} w(\hat{t}) \Phi(t) \right)^2,$$

where R_i is the union of $R_\alpha(S_i)$ over α . Finally, we also have

$$(49) \quad \varphi_i \geq \sum_{t \in R_i} w(\hat{t}) \Phi(t),$$

because $R_i \subseteq \partial S_i$. Combining all these facts, we get

$$(50) \quad \mathbf{V}^{(i)} \varphi_{i+1} \leq \max_{\alpha \in A_i} \tilde{\nu}_\alpha \varphi_i^2,$$

which implies that

$$(51) \quad \mathbf{E}^{(i)} \varphi_{i+1}^2 \leq \left(1 + \max_{\alpha \in A_i} \tilde{\nu}_\alpha\right) \varphi_i^2,$$

and consequently that

$$(52) \quad \mathbf{E} \varphi_{i+1}^2 \leq (1 + \max_{\alpha \in A_i} \tilde{\nu}_\alpha) \mathbf{E} \varphi_i^2.$$

The desired bound is now obtained by unfolding this inequality and identifying φ_1 with φ . $\square$

The preceding technique can also be applied to analyze the average variance of $\hat{\varphi}$ when the input tree $\mathbf{T}$ is random. Let the superscripts (i) and $\{i\}$ on an operator denote, respectively, the conditional operator given S_i , and the conditional operator given S_i and $\mathbf{T}$. Suppose that $\Phi(s)$ for each s in stratum α shares (but not necessarily independently) the same distribution Φ_α . Then instead of $\tilde{\nu}$, we can express a bound on $\mathbf{E}\mathbf{V}(\hat{\varphi} \mid \mathbf{T})$ in terms of the average variation $\bar{\nu}$, whose component corresponding stratum α is defined as

$$(53) \quad \bar{\nu}_\alpha \stackrel{\text{def}}{=} \mathbf{V}\Phi_\alpha / \mathbf{E}^2\Phi_\alpha.$$

LEMMA 13.

$$(54) \quad \mathbf{E}^{(i)} \mathbf{V}^{\{i\}} \Phi(s_\alpha) \leq \mathbf{V}\Phi_\alpha.$$

Proof. The conditional variable $\mathbf{V}^{\{i\}} \Phi(s_\alpha)$ can be put in the form

$$(55) \quad \sum_j p_j \Phi(s_j)^2 - \left(\sum_j p_j \Phi(s_j) \right)^2,$$

where p_j is the probability that we choose s_j to be s_α . Taking the conditional expectation $\mathbf{E}^{(i)}$ of the first sum, we get $\sum_j p_j \mathbf{E}\Phi_\alpha^2 = \mathbf{E}\Phi_\alpha^2$. Let X^2 be the second term. By the nonnegativity of the variance, we have $\mathbf{E}^{(i)} X^2 \geq (\mathbf{E}^{(i)} X)^2$. But $\mathbf{E}^{(i)} X = \sum_j p_j \mathbf{E}\Phi_\alpha = \mathbf{E}\Phi_\alpha$. Combining all of the facts, we have

$$(56) \quad \mathbf{E}^{(i)} \mathbf{V}^{\{i\}} \Phi(s_\alpha) \leq \mathbf{E}\Phi_\alpha^2 - (\mathbf{E}\Phi_\alpha)^2,$$

which implies the lemma. $\square$

THEOREM 14. *Suppose that f is nonnegative and that the random tree $\mathbf{T}$ under the stratifier has average degree of variation $\bar{\nu}$ with respect to Φ . Then, under HS, the average variance of $\hat{\varphi}$ over the random input trees is bounded above by*

$$(57) \quad \left(\prod_{0 \leq i \leq r} \left(1 + \max_{\alpha \in A_i} \bar{\nu}_\alpha \right) - 1 \right) \mathbf{E}^2 \varphi,$$

where $\bigcup_{0 \leq i \leq r} A_i$ is an arbitrary layered partition of $\mathcal{P}$.

Proof. Again, consider the conditional variable

$$(58) \quad \varphi_i \stackrel{\text{def}}{=} \mathbf{E}^{\{i\}} \hat{\varphi} = \sum_{s \in S_i} w(s) f(s) + \sum_{s \in \partial S_i} w(\hat{s}) \Phi(s).$$

We have

$$(59) \quad \mathbf{E}^{(i)} \mathbf{V}^{\{i\}} \varphi_{i+1} = \sum_{\alpha \in A_i} w_\alpha^2 \mathbf{E}^{(i)} \mathbf{V}^{\{i\}} \Phi(s_\alpha)$$

$$(60) \quad \leq \sum_{\alpha \in A_i} w_\alpha^2 \mathbf{V} \Phi_\alpha^2$$

$$(61) \quad \leq \sum_{\alpha \in A_i} \bar{\nu}_\alpha (\mathbf{E}^{(i)} w_\alpha \Phi_\alpha)^2$$

$$(62) \quad \leq \left(\max_{\alpha \in A_i} \bar{\nu}_\alpha \right) \sum_{\alpha \in A_i} (\mathbf{E}^{(i)} w_\alpha \Phi_\alpha)^2,$$

where the second step follows from the preceding lemma, and the third step follows from the definition of $\bar{\nu}$. To relate this inner sum with $\mathbf{E}^{(i)} \varphi_i$, notice that

$$(63) \quad \sum_{\alpha \in A_i} (\mathbf{E}^{(i)} w_\alpha \Phi_\alpha)^2 \leq \sum_{\alpha} \left(\sum_{t \in R_\alpha(S_i)} w(\hat{t}) \mathbf{E}^{(i)} \Phi(t) \right)^2$$

$$(64) \quad \leq \left(\sum_{t \in R_i} w(\hat{t}) \mathbf{E}^{(i)} \Phi(t) \right)^2$$

$$(65) \quad \leq (\mathbf{E}^{(i)} \varphi_i)^2$$

$$(66) \quad \leq \mathbf{E}^{(i)} \varphi_i^2.$$

Consequently, we have

$$(67) \quad \mathbf{E} \varphi_{i+1}^2 \leq \left(1 + \max_{\alpha \in A_i} \bar{\nu}_\alpha \right) \mathbf{E} \varphi_i^2.$$

Unfolding this inequality then yields

$$(68) \quad \mathbf{E} \mathbf{E}(\hat{\varphi}^2 | \mathbf{T}) \leq \prod_{0 < i \leq r} \left(1 + \max_{\alpha \in A_i} \bar{\nu}_\alpha \right) \mathbf{E}^2 \varphi,$$

which implies the theorem, since

$$(69) \quad \mathbf{E} \mathbf{V}(\hat{\varphi} | \mathbf{T}) = \mathbf{E} \mathbf{E}(\hat{\varphi}^2 | \mathbf{T}) - \mathbf{E} \mathbf{E}^2(\hat{\varphi} | \mathbf{T})$$

$$(70) \quad \leq \mathbf{E} \mathbf{E}(\hat{\varphi}^2 | \mathbf{T}) - \mathbf{E} \varphi^2$$

$$(71) \quad \leq \mathbf{E} \mathbf{E}(\hat{\varphi}^2 | \mathbf{T}) - \mathbf{E}^2 \varphi. \quad \square$$

Thus, we can deduce the performance of HS from the “goodness” of h , as quantified by the parameters $\tilde{\nu}$ and $\bar{\nu}$.

4.7. A random graph problem. We illustrate the previous theorem with the following example. Consider the maximum independent set problem in which the object is to find the maximum number of independent vertices with each pair not connected by an edge. Suppose that we take the obvious approach of solving this problem by exhaustively enumerating every independent set in the graph [10]. Then it is useful to estimate the size (the number of nodes) of the search tree before running the actual brute-force algorithm. Thus, imagine applying HS with $f = 1$ on the backtrack tree induced by the input graph $G = (V, E)$ to the maximum independent set problem. In the backtrack tree, each node s corresponds to an independent set I as well as to the subgraph $G' = (V', E')$ of G with I and its neighborhood removed. Since our objective is to estimate the size of the tree, it is both natural and simple to use the cardinality of V' as a stratifier.

Now suppose that the input graph G is random in that every edge is independently likely to occur with probability $p = 1/2$. We can proceed to examine the average variance of the estimator produced by HS by obtaining some bound on $\bar{v}$. Thus, we need to study the distribution of Φ_n , with Φ_n being the size of the random tree induced by a random graph G_n of order n . Since Φ_n is equal to the number of independent sets in G_n , we can use standard techniques in random graph theory to deduce that

$$(72) \quad \mathbf{V}\Phi_n = O\left(\frac{\lg^4 n}{n^2}\right) \mathbf{E}^2\Phi_n,$$

where $\lg n$ denotes the binary logarithm of n . (A proof is provided in the appendix.) Therefore, we can immediately conclude the following.

THEOREM 15. *Suppose, as described in the beginning of this section, that we apply HS with the stratifier that uses the cardinality V' , on the backtrack tree induced by a random graph of order n , with edge probability $p = 1/2$. Then the variance of the estimator $\hat{\varphi}$, averaged over the possible random trees, is only on the order of the square of the average size of the tree. That is, we have*

$$(73) \quad \mathbf{E}(\mathbf{V}\hat{\varphi} \mid \mathbf{T}) = O(\mathbf{E}^2\varphi).$$

Proof. Combining Theorem 14 with (72), we have for some positive constant c that

$$\begin{aligned} \mathbf{E}(\mathbf{V}\hat{\varphi} \mid \mathbf{T}) &\leq \left(\prod_{1 \leq k < n} \left(1 + c \frac{\lg^4 k}{k^2} \right) - 1 \right) \mathbf{E}^2\varphi \\ &< \left(\prod_{k=1}^{\infty} \left(1 + c \frac{\lg^4 k}{k^2} \right) - 1 \right) \mathbf{E}^2\varphi, \end{aligned}$$

where the infinite product is convergent. $\square$

Thus, we have demonstrated one situation in which the average performance of HS using a natural stratifier is quite satisfactory, at least in a theoretical sense.

5. Computational experience. The practicality of HS has been tested in a variety of situations [3], so we will describe only one experiment here. Consider the chess-board recreation whose goal is to find the longest uncrossed knight's tour possible on the board (Fig. 2). Obviously, we can try to solve this problem by means of backtracking; however, it is not clear whether an exhaustive approach is feasible. In the original paper [5] where this example appears, Knuth answers the feasibility question by applying

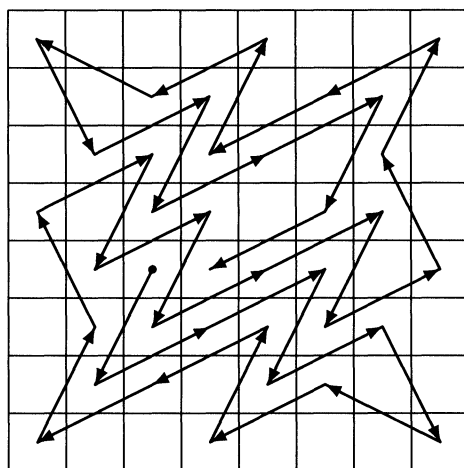


FIG. 2. A (longest) uncrossed knight's tour.

SS on the backtrack tree; using just the depth stratifier, he is able to pinpoint the number of nodes in the tree with surprising accuracy.

Although Knuth's experiment is successful in estimating the size of the tree, the estimation results can still be improved significantly by incorporating problem-specific heuristics into the random search algorithm. To illustrate this point, we identify each node in the backtrack tree with a corresponding uncrossed knight's tour, and consider the knight's tour problem as a directed-graph problem in which each vertex corresponds to a square on the chessboard, and each edge corresponds to a legal knight's move. Each edge of course precludes a number of other edges because we are only interested in tours that do not cross themselves. Now, among the knight's tours, we expect those tours that corner themselves to have less "mobility," and hence less possibility for larger subtrees than those that do not. We can gauge the mobility of a tour by counting the number of edges that have not yet been precluded by the tour and can still be reached with the edge preclusion condition relaxed. The resulting function, denoted by $\text{mobility}(s)$ with s being the node corresponding to the tour, is in fact a stratifier, as can be verified. With h_0 being the simple depth stratifier, and h_1 being the mobility function just defined, we can construct yet another stratifier h_2 by simply concatenating the values of the two former ones.

The respective performance of HS using these three stratifiers is shown empirically in Table 1. First, we repeated Knuth's experiment [5] by recording the mean number of nodes predicted for each level after running the h_0 algorithm 1000 times. (The data obtained do not coincide with the original ones because our pseudo-random number generators were different.) Next, we ran the h_1 algorithm 40 times, the h_2 algorithm 5 times, and recorded the corresponding estimates. For simplicity, we measured the total cost of each algorithm by the number of nodes examined. This accounting explains the numbers 40 and 5; for that was when the latter two algorithms had examined roughly the same number of nodes as the first one. To serve as a reference, the true profile of the tree is reproduced here. Incidentally, for a fairer comparison, we should really take into account the cost of evaluating $\text{mobility}(s)$. Because the h_0 algorithm need not compute this function, we could probably run it 6000 times to equal one trial of sampling with h_2 .

TABLE 1
Estimated tree profiles.

Level	h_0	h_1	h_2	Actual
0	1	1	1	1
1	10	10	10	10
2	43	42	34	42
3	257	196	204	251
4	1005	918	864	968
5	4267	3734	3604	4215
6	15684	14026	13724	15646
7	57443	46873	51326	56435
8	184017	126007	174680	182520
9	583197	445188	611950	574555
10	1608183	1613621	1832792	1606422
11	4371678	4679457	5106228	4376153
12	9920481	16793389	11767976	10396422
13	22507886	37750158	25507044	23978392
14	47054083	31161416	47574616	47667686
15	95846027	64365367	83428348	91377173
16	156085563	106268997	130208416	150084206
17	256650972	205655280	192462996	235901901
18	344261889	306859328	252526926	315123658
19	451703678	382647057	280613374	399772215
20	324806815	372852597	277536920	427209856
21	253992102	426672714	272541294	429189112
22	237855139	239446386	288235830	358868304
23	106973568	352833124	226878632	278831518
24	80018150	195532658	138579650	177916192
25	28304640	57001796	87525874	103894319
26	0	72444273	40045814	49302574
27	0	19732265	16732876	21049968
28	0	151248453	9984256	7153880
29	0	0	2985372	2129212
30	0	0	787150	522186
31	0	0	138532	109254
32	0	0	13210	18862
33	0	0	588	2710
34	0	0	588	346
35	0	0	0	50
36	0	0	0	8
total	2422806779	3046195329	2393871699	3137317290
std dev	10672087188	2331712470	878149488	
cost	11503	10875	11345	
trials	1000	40	5	

However, this comparison would depend more on the actual implementation.

It is apparent that the estimation power of HS increases upon the application of more sophisticated stratifiers. True, the three estimates for the actual size of the tree are all accurate enough to give us the right order of magnitude, but the estimates produced by HS using h_1 or h_2 are much more stable and hence give us more confidence than those produced by the algorithm using h_0 . This stability is indicated by the decreasing order of magnitude of the observed deviation as we go from h_0 to h_2 .

Besides estimating the size of the tree, we can also use HS to estimate the entire profile of the tree. By computing the product of the branching degrees of path associated with one of the eight 35-move solution, we reckon that it would take about a billion trial

runs for SS to come up with a nonzero estimate for the number of nodes at the bottom level. Fortunately, as Table 1 clearly shows, the sampling technique can probe deeper into the tree with the aid of simple heuristics. Take, for example, the stratifier h_2 ; in just five trial runs, it has enabled HS to find a 33-move knight's tour.

In addition to estimating the cost of brute-force backtracking, we can also predict the performance of other heuristic search algorithms by profiling the backtrack tree according to the heuristic used. For example, consider the heuristic $F(s)$, which when given a state s with a partial uncrossed knight's tour, computes the total number of vertices whose set includes those already on the tour and those still reachable with the edge-preclusion condition relaxed. Because F is monotonically nonincreasing, we can invoke a best-first search strategy and be certain of an optimum solution when a leaf node is reached. However, a best-first strategy may require too much space; in this case, an iterative-deepening approach may be more appropriate. Using HS, we can predict whether these heuristic search algorithms are indeed better than simple backtracking by first estimating

$$(74) \quad \varphi_1(F_0) = \sum_s [F(s) = F_0],$$

where $[\cdot]$ denotes the indicator function: 1 if the predicate is satisfied, and zero otherwise. Then, by calculating

$$(75) \quad \varphi_2(F_0) = \sum_{F_1 \geq F_0} \varphi_1(F_1),$$

we can predict the number of nodes that a best-first search has to expand before it can process any s with $F(s) < F_0$. Likewise, by further summing

$$(76) \quad \varphi_3(F_0) = \sum_{F_1 \geq F_0} \varphi_2(F_1),$$

we can predict the number of nodes that an iterative-deepening search (with ∞ as the initial bound) has to expand before it can process any s with $F(s) < F_0$.

Shown in Table 2 is the result of applying HS 10 times using stratifier h_2 . For simplicity, we list only the average of the estimates for $F_0 \geq 30$, and compare only those with $F_0 \geq 36$ with the actual values that were computed later. The estimates are all quite accurate in determining the order of magnitude of the actual values, even though the total number of nodes examined by HS (22750) is only about 1/10000 of that required by an exhaustive search. In our implementation, the sampling procedure took about 10 minutes on a Sun, while the exact computation took about 3 days of work distributed among 4 Suns.

Incidentally, if we know that an optimum tour has length around 35, then with the estimates we would be able to predict the infeasibility of a best-first search and the inferiority of an iterative-deepening approach with respect to simple backtracking. A best-first search would require us to maintain a queue of about 10^8 nodes, even though it needs only to examine about 1/15 of the nodes in the entire backtrack tree. On the other hand, an iterative-deepening approach with ∞ as the initial bound would require an examination of about 1/3 of the backtrack tree, a factor which is not small enough to offset the cost of evaluating F . However, if we know that the optimum length is definitely equal to 35, then an iterative-deepening approach with 36 as the initial bound on F might now be better than simple backtracking, since it needs to examine only about 1/15 of the tree.

TABLE 2
Tree profile under a monotone heuristic.

F_0	$\widehat{\varphi}_1(F_0)$	$\varphi_1(F_0)$	$\widehat{\varphi}_2(F_0)$	$\varphi_2(F_0)$	$\widehat{\varphi}_3(F_0)$	$\varphi_3(F_0)$
64	39815	39379	39815	39379	39815	39379
63	111703	86641	151518	126020	191333	165399
62	144451	151599	295969	277619	487302	443018
61	226524	228636	522493	506255	1009795	949273
60	433814	309611	956307	815866	1966102	1765139
59	427256	403924	1383563	1219790	3349665	2984929
58	530568	512112	1914131	1731902	5263796	4716831
57	845676	614624	2759807	2346526	8023603	7063357
56	825869	756939	3585676	3103465	11609279	10166822
55	1329818	923511	4915494	4026976	16524773	14193798
54	1228600	1139180	6144094	5166156	22668867	19359954
53	991948	1439727	7136042	6605883	29804909	25965837
52	2659514	1793783	9795556	8399666	39600465	34365503
51	2669262	2211864	12464818	10611530	52065283	44977033
50	2499762	2741692	14964580	13353222	67029863	58330255
49	4332987	3379471	19297567	16732693	86327430	75062948
48	3465842	4178690	22763409	20911383	109090839	95974331
47	5277864	5144058	28041273	26055441	137132112	122029772
46	5980861	6322974	34022134	32378415	171154246	154408187
45	7613390	7753753	41635524	40132168	212789770	194540355
44	9427006	9486538	51062530	49618706	263852300	244159061
43	10839121	11483764	61901651	61102470	325753951	305261531
42	15375928	13812502	77277579	74914972	403031530	380176503
41	13808601	16432504	91086180	91347476	494117710	471523979
40	15494192	19418400	106580372	110765876	600698082	582289855
39	24607447	22642483	131187819	133408359	731885901	715698214
38	21976276	26231844	153164095	159640203	885049996	875338417
37	29037331	29968602	182201426	189608805	1067251422	1064947222
36	31250938	33920456	213452364	223529261	1280703786	1288476483
35	29411721		242864085		1523567871	
34	35719965		278584050		1802151921	
33	37585184		316169234		2118321155	
32	40970294		357139528		2475460683	
31	34998800		392138328		2867599011	
30	49462557		441600885		3309199896	

6. Conclusions. Heuristic sampling is a simple, easy to apply, and easy to implement technique designed to predict the performance of other tree searching programs. The flexibility and practicality of this technique stem from our accessibility to the input stratifier, through which natural heuristics can be introduced to guide and control the estimation process. In evaluating its effectiveness, we have shown the unbiasedness of the estimator and investigated the conditions for minimum variance. We have also demonstrated a relationship between the quality of the stratifier and the accuracy of the estimator by formalizing the concept of stratum homogeneity. In supporting our general results, we have also conducted some specific case analysis, one involving theoretical computation, and the other, experimental simulation. The results obtained are all very encouraging. Overall, we expect heuristic sampling to be an invaluable asset in aiding the design of tree searching programs as we use the technique to compare the merits of various search algorithms.

Appendix. Let G_n be a random graph with n vertices and independent edge probability p , and let Φ_n be the number of independent sets in G_n . To study the expectation

and the variance of Φ_n , we use related techniques and results derived in §4 of [10], §XI.1 of [1], and §5.3 of [7]. Thus, imagine an enumeration of all 2^n subgraphs H_i of G_n . Let X_i be the 0-1 random variable indicating the independence of H_i ; we have $\Phi_n = \sum_i X_i$. The probability that $X_i = 1$, or the probability that the subgraph H_i is independent, is equal to $q^{\binom{k}{2}}$, where k is the order of this subgraph and $q = 1 - p$. Since there are $\binom{n}{k}$ subgraphs of order k , we have

$$(77) \quad \mathbf{E}\Phi_n = \sum_k \binom{n}{k} q^{\binom{k}{2}},$$

and according to Wilf, for $p = q = 1/2$, this sum has most of its contributions coming from terms with k around $\lg n$.

To compute $\mathbf{V}\Phi_n$, let us first consider

$$(78) \quad \mathbf{E}\Phi_n^2 = \sum_{i,j} \mathbf{E}X_i X_j.$$

Suppose that the subgraphs H_i and H_j are of orders k and l , respectively. Then the probability that $X_i X_j = 1$, or the probability that both H_i and H_j are independent, is equal to $q^{\binom{k}{2} + \binom{l}{2} - \binom{r}{2}}$, where r is the number of vertices H_i and H_j have in common. On the other hand, $\binom{n}{k-r, l-r, r}$ is the number of such pairs of subgraphs having r vertices in common, with the first of order k and the second of order l . Hence, we have

$$(79) \quad \mathbf{E}\Phi_n^2 = \sum_{k,l,r} \binom{n}{k-r, l-r, r} q^{\binom{k}{2} + \binom{l}{2} - \binom{r}{2}},$$

and under the same interpretation, we have

$$(80) \quad \mathbf{E}^2\Phi_n = \sum_{k,l,r} \binom{n}{k-r, l-r, r} q^{\binom{k}{2} + \binom{l}{2}},$$

which gives us

$$(81) \quad \mathbf{V}\Phi_n = \sum_{k,l} q^{\binom{k}{2} + \binom{l}{2}} \sum_{2 \leq r \leq \min(k,l)} \binom{n}{k-r, l-r, r} (q^{-\binom{r}{2}} - 1).$$

Thus, we define $b = 1/q$ and analyze the quantity

$$(82) \quad A_{k,l} \stackrel{\text{def}}{=} q^{\binom{k}{2} + \binom{l}{2}} \sum_{2 \leq r \leq k} \binom{n}{k-r, l-r, r} b^{\binom{r}{2}}$$

for different regions of k and l to bound $\mathbf{V}\Phi_n$.

LEMMA 16. If $k \leq l$ and $l \geq 1 + 4 \log_b n$, then

$$(83) \quad A_{k,l} \leq 1.$$

Proof.

$$(84) \quad A_{k,l} \leq q^{\binom{l}{2}} \sum_{r \leq k} \binom{n}{k-r, l-r, r}$$

$$(85) \quad = q^{\binom{l}{2}} \binom{n}{k} \binom{n}{l}$$

$$(86) \quad \leq \left(\frac{n^2}{b^{(l-1)/2}} \right)^l$$

$$(87) \quad \leq 1^l = 1. \quad \square$$

LEMMA 17. *Let*

$$(88) \quad B_r \stackrel{\text{def}}{=} \binom{n}{k-r, l-r, r} b^{\binom{r}{2}}.$$

If $l < 5 \log_b n$ and $3 \leq r \leq k \leq l$, then

$$(89) \quad B_r < B_3 + B_k$$

for sufficiently large n .

Proof. Suppose that $r \leq \frac{3}{2} \log_b n$. Then consider the ratio

$$(90) \quad a_r \stackrel{\text{def}}{=} B_r / B_3.$$

It is equal to

$$(91) \quad \left(\frac{3!}{r!} \right) \left(\frac{(k-3)!}{(k-r)!} \right) \left(\frac{(l-3)!}{(l-r)!} \right) \left(\frac{(n-k+3)!}{(n-k+r)!} \right) b^{\binom{r}{2}-3},$$

which is bounded above by

$$(92) \quad \left(\frac{(k-3)(l-3)}{(n-k+3)} \right)^{r-3} b^{r(r-1)/2}.$$

Hence, for sufficiently large n , we have

$$(93) \quad a_r = O(klb^{r/2}/n)^r$$

$$(94) \quad = O(kln^{3/4}/n)^r$$

$$(95) \quad = o(1).$$

Now suppose that $r \geq \frac{3}{2} \log_b n$. Then consider the ratio

$$(96) \quad b_r \stackrel{\text{def}}{=} B_r / B_k.$$

After expansion, it becomes

$$(97) \quad \left(\frac{0!}{(k-r)!} \right) \left(\frac{k!}{r!} \right) \left(\frac{(l-k)!}{(l-r)!} \right) \left(\frac{(n-l)!}{(n-l-(k-r))!} \right) b^{\binom{r}{2}-\binom{k}{2}},$$

which can be bounded above by

$$(98) \quad (kn)^{k-r} / b^{\binom{k}{2}-\binom{r}{2}}.$$

But notice that $k \geq r$ implies that

$$(99) \quad \binom{k}{2} - \binom{r}{2} \leq \frac{r(k-1)}{2} - \frac{r(r-1)}{2} = \frac{r(k-r)}{2}.$$

Hence, for sufficiently large n , we have

$$(100) \quad b_r = O(kn/b^{r/2})^{k-r}$$

$$(101) \quad \leq O(kn/n^{3/2})^{k-r}$$

$$(102) \quad = o(1).$$

□

LEMMA 18. If $k \leq l < 5 \log_b n$, then for sufficiently large n , we have

$$(103) \quad A_{k,j} \leq \left(\frac{bl^4}{n^2} + \frac{b^3 l^7}{n^3} \right) E_k E_l + l^{l+1} E_l,$$

where

$$(104) \quad E_k \stackrel{\text{def}}{=} \binom{n}{k} q^{\binom{k}{2}}.$$

Proof. According to the previous lemma, we can bound $A_{k,j}$ by

$$(105) \quad q^{\binom{k}{2} + \binom{l}{2}} (B_2 + (k-2)(B_3 + B_k)).$$

Next, observe that

$$(106) \quad \binom{n}{l-k, 0, k} = \binom{n}{l} \binom{l}{k} \leq \binom{n}{l} l^l$$

and

$$(107) \quad \binom{n}{k-r, l-r, r} \leq \frac{k^r l^r}{r! (n-r)^r} \binom{n}{k} \binom{n}{l}$$

$$(108) \quad \leq \frac{k^r l^r}{n^r} \binom{n}{k} \binom{n}{l}.$$

The lemma then follows immediately by substituting these bounds into the definition of B_r . □

LEMMA 19. For sufficiently large n , and $p = q = 1/2$, we have

$$(109) \quad \mathbf{V}\Phi_n = O\left(\frac{\lg^4 n}{n^2}\right) \mathbf{E}^2 \Phi_n.$$

Proof. From Lemma 16 comes the bound

$$(110) \quad \mathbf{V}\Phi_n \leq \sum_{k,l} 1 + \sum_{k,l < 5 \lg n} A_{k,l}.$$

Next, we apply Lemma 18 to bound $A_{k,l}$. By setting $p = q = 1/2$, we have

$$(111) \quad \mathbf{V}\Phi_n \leq n^2 + O\left(\frac{\lg^4 n}{n^2}\right) \sum_{k,l} E_k E_l + n^{O(\lg \lg n)} \sum_k E_k,$$

which implies the result since

$$(112) \quad \sum_r E_r = \mathbf{E}\Phi_n = \Omega(n^{\frac{1}{2} \lg n - \lg \lg n}).$$

□

Acknowledgment. This paper is based on the author's thesis supervised by Don Knuth. The author thanks Ernie Brickell, Carl Diegert, and David Strip for their encouragement and review of this publication.

REFERENCES

- [1] B. BOLLOBÁS, *Random Graphs*, Academic Press, New York, 1985.
- [2] P. BRATLEY, B. L. FOX, and L. E. SCHRAGE, *A Guide to Simulation*, Second Ed., Springer-Verlag, Berlin, New York, 1987.
- [3] P. CHEN, *Heuristic Sampling on Backtrack Trees*, Ph.D. thesis, Stanford University, Stanford, CA, 1989.
- [4] S. KARLIN AND H. M. TAYLOR, *A First Course in Stochastic Processes*, Second Ed., Academic Press, New York, 1975.
- [5] D. E. KNUTH, *Estimating the efficiency of backtrack programs*, Math. Comp., 29 (1975), pp. 121–136.
- [6] R. E. KORF, *Depth-first iterative-deepening: An optimal admissible tree search*, Artificial Intelligence, 27 (1985), pp. 97–109.
- [7] E. M. PALMER, *Graphical Evolution*, John Wiley, New York, 1985.
- [8] J. PEARL, *Heuristics: Intelligent Search Strategies for Computer Problem Solving*, Addison-Wesley, Boston, MA, 1984.
- [9] P. W. PURDOM, *Tree size by partial backtracking*, SIAM J. Comput., 7 (1978), pp. 481–491.
- [10] H. S. WILF, *Some examples of combinatorial averaging*, Amer. Math. Monthly, 92 (1985), pp. 250–261.