# Which Search Problems Are Random?*

Tad Hogg

**Abstract**

The typical difficulty of various NP-hard problems varies with simple parameters describing their structure. This behavior is largely independent of the search algorithm, but depends on the choice of problem ensemble. A given problem instance belongs to many different ensembles, so applying these observations to individual problems requires identifying which ensemble is most appropriate for predicting its search behavior, e.g., cost or solubility. To address this issue, we introduce a readily computable measure of randomness for search problems called approximate entropy. This new measure is better suited to search than other approaches, such as algorithmic complexity and information entropy. Experiments with graph coloring and 3SAT show how this measure can be applied.

## 1 Introduction

Search is ubiquitous in AI. Hence the importance of developing rapid predictions of behavior, e.g., a problem's search cost or solubility. Such predictions could help select among alternate heuristics or problem formulations, decide whether to collect additional domain knowledge, or construct portfolios [12, 16]. A significant step toward this goal was the recognition of regularities, such as phase transitions, in many search ensembles, i.e., classes of problems with an associated probability for each to occur [5, 24, 8, 11, 14, 28]. These observations have also led to new heuristics [6, 10, 20]. Two key difficulties limit the application of this work to individual searches. First, the variance of ensemble behaviors is often very large, limiting the accuracy of predictions based on ensemble averages. Second, every problem is a member of many ensembles, with differing typical behaviors. So it is difficult to know which ensemble best applies to a particular problem.

This paper addresses the second difficulty by introducing a new measure of randomness particularly well-suited for search through its connection to the underlying problem structure [28]. This measure is then applied to distinguish typical instances of different search ensembles and to elucidate questions raised by a previous study of problem structure based on algorithmic complexity [27]. The results also give additional insight into the nature of the large variances.

## 2 Quantifying Randomness

When is a problem a typical member of an ensemble? This question's conceptual difficulty is seen with bit sequences [4, 17]: in the ensemble of random 10-bit sequences, 0101010101 and 0101110001 each appear with probability $2^{-10}$ but differ in their regularity. One way to quantify this difference is *algorithmic complexity*: irregular sequences are those that cannot be produced by a program substantially shorter than the sequence itself [4, 7]. However, this measure is not readily computable and applies only to asymptotically long sequences. Another approach is the *entropy* of the sequence probabilities [7, 26]. While readily computed, it applies to the process generating the sequences rather than to a resulting instance. Thus its usefulness is limited since different processes can produce the same instance. E.g., the first 10-bit sequence given above could also be produced by alternating 0 and 1 bits, a process with lower entropy than random generation.

A practical measure must be readily computable, meaningful even for fairly small instances, and defined by an instance itself rather than the, possibly unknown, process that created it. One measure with these


Appeared in *Proc. of AAAI98*, pp. 438-443, 1998.

properties is *approximate entropy* (ApEn) of sequences [21]. In this paper we apply ApEn to search problems. Specifically, this section defines ApEn for any discrete structure by comparing its component parts, generalizing a technique previously introduced for hierarchies [15]. Sequences and graphs are examples of such structures, whose parts are, respectively, subsequences and subgraphs. The next section describes how these ideas apply to combinatorial searches.

Let $C$ be a structure of size $n$ with $N$ parts, $\{c_1, \ldots, c_N\}$, of size $m \leq n$. Let $R(c, c')$ be an equivalence relation among the parts, identifying the set of $D$ *distinct* parts $\{d_1, \ldots, d_D\}$ of size $m$. Let $n_i$ be the number of times the $i^{th}$ distinct part $d_i$ appears in $C$, and $f_i = n_i/N$ its frequency, for $i = 1, \ldots, D$. These frequencies define

$$\Phi^{(m)} = -\sum_{i=1}^{D} f_i \ln f_i \tag{1}$$

The approximate entropy (ApEn) for size $m$ is

$$\mathrm{ApEn}(m) = \Phi^{(m+1)} - \Phi^{(m)} \tag{2}$$

and characterizes the log-likelihood two parts of size $m + 1$ are equivalent given that they each contain equivalent parts of size $m$ [21].

For sequences [21], $R$ tests for identical subsequences. Both sequences given above have 5 0's and 5 1's so $\Phi^{(1)} = \ln 2$. For $m = 2$, the regular sequence has 5 copies of 01 and 4 of 10, so $\Phi^{(2)} = -\frac{5}{9} \ln \frac{5}{9} - \frac{4}{9} \ln \frac{4}{9}$ or 0.69. The irregular sequence has 2 copies each of 00, 10 and 11, and 3 of 01, giving $\Phi^{(2)} = -3\frac{2}{9} \ln \frac{2}{9} - \frac{3}{9} \ln \frac{3}{9}$ or 1.37.

# 3 Randomness of Search Problems

To apply ApEn to search, we focus on constraint satisfaction problems (CSPs) [18] consisting of $n$ variables and the requirement to assign a value to each variable to satisfy given constraints. Searches examine various *assignments*, which give values to some of the variables. These assignments, and their consistency relationships, form the *deep structure* of the CSP, which is fully specified by the inconsistent assignments directly specified by the constraints, i.e., the *minimized nogoods* [28].

Using ApEn with CSPs requires specifying the parts to use and an equivalence relation among them. A natural choice relevant for search is to use subproblems defined by sets of $m$ variables. A subproblem's constraints and minimized nogoods are just those from the full problem involving only the subproblem's $m$ variables. Subproblems can be compared in various ways. A simple method labels their variables from 1 to $m$ in the same order as they appear in the full problem. Two subproblems are then considered equivalent when they have the same number of minimized nogoods (indicating the degree to which problems are constrained), and the same sets of variables involved together in these nogoods. That is, for each subset of $\{1, \ldots, m\}$, the corresponding variables in the two subproblems must appear together in at least one nogood of each subproblem, or not appear in any nogoods. Alternatively, each set of variables that appears together in at least one nogood can be viewed as a hyperedge in a hypergraph [1] whose nodes are the subproblem's variables. From this viewpoint, the equivalence relation requires both identical constraint hypergraphs and the same number of minimized nogoods.

A problem with $n$ variables has $N = \binom{n}{m}$ subproblems with $m$ variables. Because the number of subproblems grows rapidly with $m$, it is important that significant distinctions between problems are readily apparent even with small values of $m$. In fact, for studies of sequences [22] as well as the experiments on search problems reported here, values of 3 or 4 for $m$ suffice, even for significantly different problem sizes. This use of small $m$ makes approximate entropy tractably computable for search problems[1].

When ApEn varies over a small range, it is convenient to use

$$\mathrm{def}(m) = \mathrm{ApEn}_{\max} - \mathrm{ApEn}(m) \tag{3}$$

where $\mathrm{ApEn}_{\max}$ is the maximum value of $\mathrm{ApEn}(m)$ for a given *class* of problems, and is the same for *all* ensembles defined on that class of problems.

[1] Although not used in this paper, faster, but approximate, estimates of the $f_i$ values are possible by sampling only some of the $m$ variable subproblems.

# 4 Examples

Most empirical studies of search use random ensembles, with constraints selected randomly from among all possibilities. Ensembles emphasizing more realistic structures have quantitatively different behaviors [13]. In many cases the constraints arise from physical interactions whose strength decrease with distance. In others, some variables are more critical and hence more tightly constrained than others. These cases have clustered constraints: some sets of variables are more constrained than expected from random generation. Thus we examine ApEn in random and clustered ensembles for two well-studied problems: graph coloring and satisfiability.

## 4.1 Graph Coloring

A graph coloring problem consists of a graph, a specified number of colors, and the requirement to find a color for each node in the graph such that adjacent nodes (i.e., nodes linked by an edge in the graph) have distinct colors. Viewed as a CSP, each node in the graph is a variable. The minimized nogoods are pairs of adjacent nodes assigned the same color. With $b$ colors, each edge gives $b$ such nogoods. Thus the number of constraints is uniquely determined by the number of edges $e$ in the graph, or the average degree: $\gamma \equiv 2e/n$. In this paper we use $b = 3$ colors, in which case the hardest problems are concentrated near $\gamma = 4.5$ for random graphs [5].

A subproblem with $m$ variables defines a subgraph with each edge adding $b$ minimized nogoods. Hence the subproblem equivalence relation tests for identical subgraphs so ApEn is determined by the number of times each distinct subgraph with $m$ nodes appears in the graph.

For example, $m = 2$ gives $D = 2$ distinct 2-node subgraphs: those with and without an edge. The number of subgraphs with an edge equals the number of edges $e$ in the full graph, giving a frequency $f = e/\binom{n}{2}$. The other distinct subgraph, without an edge, has frequency $1 - f$. Thus, $\Phi^{(2)} = -f \ln f - (1 - f) \ln(1 - f)$ which is 0.185 for every 100-node graph with $\gamma = 4.5$. Because $\Phi^{(2)}$ depends only on $e$, we must use $\Phi^{(m)}$ with $m \geq 3$ to distinguish graphs with the same $n$ and $e$. Thus we use def(2), which involves $\Phi^{(3)}$, in our experiments.

The random graph ensemble [2] takes each graph with $n$ nodes and $e$ edges as equally likely. A clustered ensemble [13] can be formed by grouping the nodes into a binary tree, which induces an ultrametric distance between each pair of nodes, i.e., the number of levels up the tree required to reach a common ancestor. A clustered graph is generated by choosing edges between nodes at ultrametric distance $d$ with relative probability $p^d$. Random graphs result when $p = 1$, while $p < 1$ makes edges more likely between nearby nodes, giving a hierarchical clustering which shifts the location of the phase transition [13].

To evaluate search cost, we used conflict-directed backtracking based on the Brelaz heuristic [3] which assigns the most constrained nodes first (i.e., those with the most distinctly colored neighbors), breaking ties by choosing nodes with the most uncolored neighbors (with any remaining ties broken randomly). For each node, the smallest color consistent with the previous assignments is chosen first, with successive choices made when the search is forced to backtrack. As a simple optimization, we never change the colorings for the first two selected nodes to avoid unnecessarily repeating the search with a permutation of the colors. When forced to backtrack we jump back to the most recent node involved in the conflict [9, 23], a form of conflict directed backjumping. We measure the search cost by the number of assignments examined until the first solution is found or, when no solutions exist, until no further possibilities remain.

## 4.2 k-SAT

The satisfiability problem consists of a propositional formula in $n$ binary variables, and the requirement to find an assignment to the variables that makes the formula true. The formula is usually represented as a conjunction of clauses, each of which is a disjunction of some variables, possibly negated. For k-SAT problems, each clause has $k$ variables. The minimized nogoods correspond to the clauses in the formula: each clause excludes a single assignment to the $k$ variables that appear in the clause [28]. For 3-SAT, the hardest problems are concentrated near $c = 4.2n$, where $c$ is the number of clauses [8].

If $m = k$, each set of $m$ variables is either together in a clause or not, so each such subproblem has one of two possible constraint hypergraphs. Since the $m$ variables could appear in more than one clause, the subproblems can have differing numbers of nogoods. Thus, unlike graph coloring, k-SAT can have different $\Phi^{(m)}$ values even when $m$ is the same size as the problem constraints.

In the class of all the 4-variable 2-SAT problems, an example is one with the two clauses $v_2 \vee v_3$ and $v_2 \vee \neg v_4$ so $v_1$ is unconstrained. For $m = 3$, there are 4 subproblems. The first, with variables 1, 2 and 3, contains only the first clause, involving the second and third subproblem variables, so the subproblem has a single nogood. The second subproblem, with variables 1, 2 and 4, contains only the second clause, again involving the second and third subproblem variables, and one nogood. When these subproblems' variables are labeled from 1 to 3, the second subproblem has the same variables in clauses as the first, so they satisfy the equivalence relation since they also both have one nogood. The third subproblem, with variables 1, 3 and 4, has no clauses, and is thus distinct from the first two. Finally, the fourth subproblem, with variables 2, 3 and 4, contains both clauses, has two nogoods and is distinct from all the other subproblems. Thus three distinct subproblems exist, with the first appearing twice and the rest appearing once, giving frequencies $f_1 = \frac{2}{4}$ and $f_2 = f_3 = \frac{1}{4}$. Hence, $\Phi^{(3)} = -\frac{2}{4} \ln \frac{2}{4} - 2\left(\frac{1}{4} \ln \frac{1}{4}\right)$ or $\frac{1}{2} \ln 8$. This example also illustrates that not all variables need appear in the constraints.

The random k-SAT ensemble selects the specified number of clauses randomly. For a clustered ensemble, we suppose that half the variables are more likely to be involved in clauses than the others. Specifically, for 3-SAT, we select half the clauses from among those that involve only the first $n/2$ variables, and then take the remaining clauses from among those that involve at least one of the second half of the variables. This results in a different form of clustering than the multilevel hierarchical clustering graph ensemble, allowing a broader range of empirical evaluation of ApEn behavior.

The SAT problems were solved with GSAT [25], an incomplete search method, to contrast with the exhaustive backtrack method used for the graph coloring experiments. In its simplest form, GSAT consists of a number of trials, which start from a randomly selected assignment and proceed in a series of steps. At each step, we count the number of conflicts in the neighbors of the current assignment and move to a neighbor with the fewest conflicts. If the search reaches a local minimum or a prespecified limit on the number of steps, a new trial is started. In the searches reported here, this limit was twice the number of variables in the problem. We performed 1000 trials to repeatedly find solutions, which allows estimating the average cost to find a solution by the total number of steps in all the trials divided by the number of times a solution was found. For the cases used here, typically several hundred trials for each problem found a solution.

# 5 Applications

Having introduced ApEn for search, we illustrate its use with two problems (graph coloring and SAT), ensembles (random and clustered), and search methods (complete and incomplete). Experiments with other combinations and problem sizes than those reported here gave similar results.

## 5.1 Identifying Typical Instances

ApEn distinguishes typical instances of random or clustered ensembles as shown in Fig. 1 and 2 for graph coloring and SAT, respectively. These figures show the cumulative distribution, i.e., the fraction of a sample of instances for which $\mathrm{def}(m)$ exceeds each value. The distributions are well separated. This separation is particularly significant for the small SAT problems in Fig. 2, in spite of the larger spread due to the small size. A similar distinction is seen with larger values of $m$ in both the graph coloring and SAT problems. Thus ApEn distinguishes typical instances of different ensembles that have significantly different transition points and typical search costs [13].

The subproblem equivalence relation used here is based on a single, arbitrary, ordering of the variables. An alternative allows permutations of the variables when comparing subproblems. For graph coloring, this amounts to testing for isomorphic, rather than identical, subgraphs. Although isomorphism is a more natural comparison of subproblems, it is more expensive to compute and, as a coarser grouping of subproblems, less able to distinguish ensembles. Repeating Fig. 1 using isomorphism has less distinct distributions, but is otherwise qualitatively similar.

Defining ApEn using problem structure is important for discriminating among ensembles. This is seen by comparison with another representation commonly used in discussions of algorithmic complexity [7],

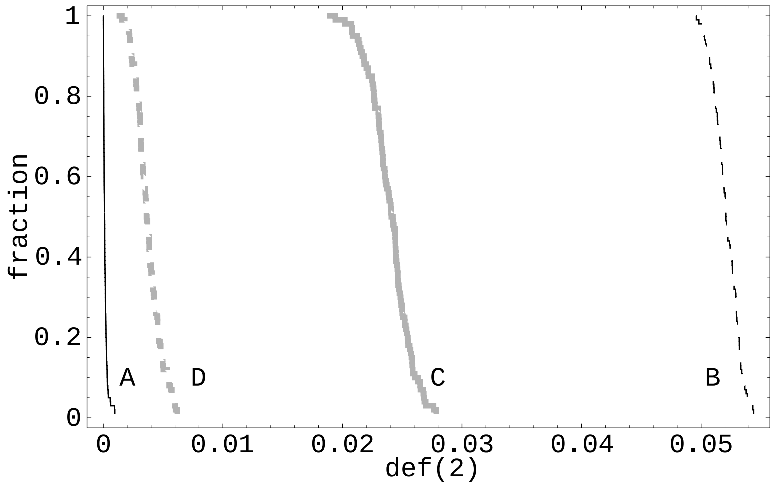


Figure 1: Cumulative def(2) distribution for 100-node graphs with $\gamma = 4.5$, defined with respect to the largest value of ApEn(2) for such graphs: $\text{ApEn}_{\text{max}} = 0.3698$. The curves each have 100 graphs but from different ensembles: random (A, solid black) and hierarchically clustered with $p$ equal to 0.1 (B, dashed black), 0.5 (C, solid gray) and 2.0 (D, dashed gray).

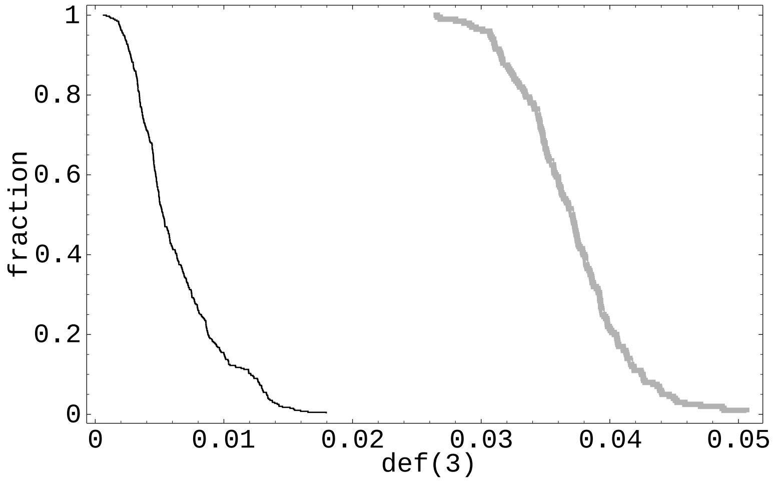


Figure 2: Cumulative def(3) distribution for 3-SAT problems with 20 variables and 80 distinct clauses, defined with respect to the largest value of ApEn(3) for such problems: $\text{ApEn}_{\text{max}} = 0.7731$. The curves each have 100 instances but from different ensembles: random (black) and clustered (gray).

but which completely ignores problem structure. Specifically, a graph's edges are a particular subset of all possible edges among its nodes. These subsets can be ordered, e.g., lexicographically [19]. Each graph then corresponds to the position, in this ordering, of its set of edges. Viewing this integer position index as a sequence of bits, allows ApEn to be computed [21], giving another definition of ApEn for graphs. Experiments similar to those of Fig. 1 show this representation does *not* distinguish the ensembles. This dependence on the representation is similar to that reported for digit sequences [21], and indicates the problem sizes here are not large enough for algorithmic complexity to be usefully applied: asymptotically, such changes in representation do not affect compressibility [7].

## 5.2 Are Hard Cases Atypical?

Within a given ensemble, the search cost varies considerably among the instances. An example is shown in Fig. 3. This behavior led to a claim that the rare high cost instances are distinctly nonrandom compared to other members of the ensemble, and the exceptionally hard cases are part of a finite set of exceptions that does not persist to larger problem sizes [27]. An alternate view is the exceptionally high costs are due to the search algorithm making unlucky choices early in the search, resulting in exhaustive search of many assignments before returning to correct the early mistake. In this case, the high costs would be due to the algorithm choices rather than intrinsic characteristics of the problem instances [24].

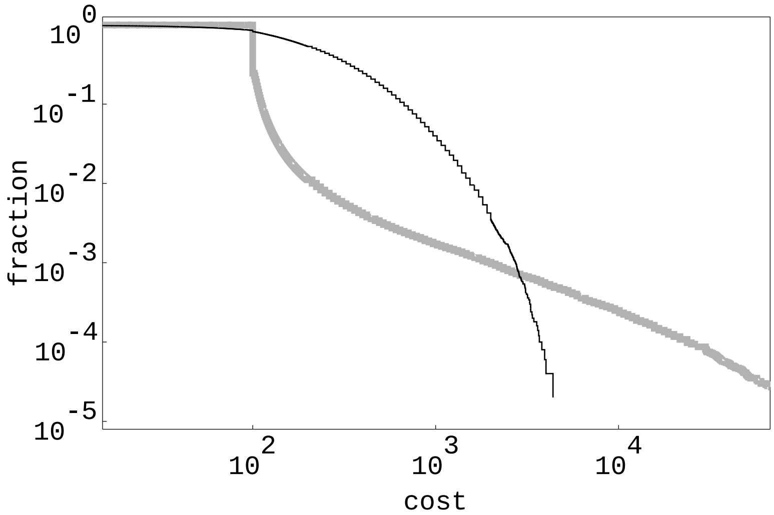


Figure 3: Cumulative search cost distribution for 100-node random graphs with $\gamma = 3.5$ (gray) and 4.5 (black) using the Brelaz heuristic, on a log-log scale. Median costs are 100 and 215, respectively.

To examine this question with ApEn for the random graph ensemble, we generated graphs that fully sample the cost distributions of Fig. 3. Specifically, we first generated 100 random graphs. We then examined additional graphs, collecting only those with a cost of at least 1000, until a second set of 100 graphs was created. This process was repeated with minimum costs of $10^4$ and $10^5$ for $\gamma = 3.5$, and minimums of 5000 and $10^4$ for $\gamma = 4.5$, giving a total of 400 graphs for each value of $\gamma$, with representatives from the high cost tail of the distributions. The result, in Fig. 4, shows no correlation between the cost and ApEn. Using isomorphic instead of identical subgraphs for the equivalence relation gives qualitatively similar behavior. For SAT problems, the distribution of GSAT search costs is shown in Fig. 5. These costs are also uncorrelated with ApEn, as shown in Fig. 6.

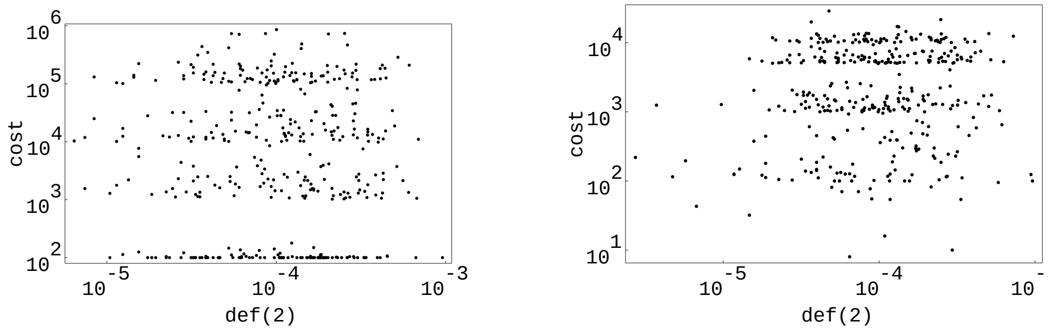


Figure 4: Graph coloring search cost vs. def(2) for 100-node random graphs with $\gamma = 3.5$ (top) and 4.5 (bottom) on a log-log scale. Maximum ApEn(2) for graphs with 175 and 225 edges is 0.305771 and 0.369815, respectively. Vertical groupings of the points reflect the cost distribution sampling described in the text. Qualitatively similar plots are seen using def(3).

Similar plots for the clustered ensembles of graph coloring and SAT show the same qualitative behavior: no correlation within an ensemble, but distinct clusterings of cost and ApEn values for the different ensembles.

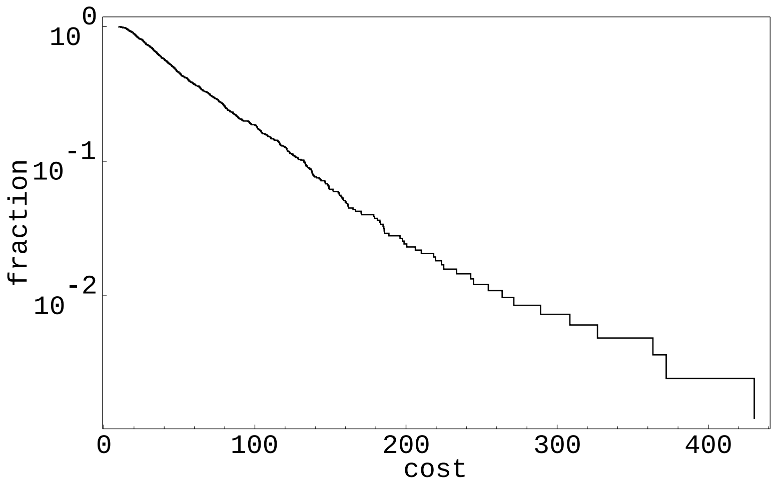


Figure 5: Cumulative search cost distribution for GSAT on random soluble 3-SAT problems for 20 variables and 80 clauses.

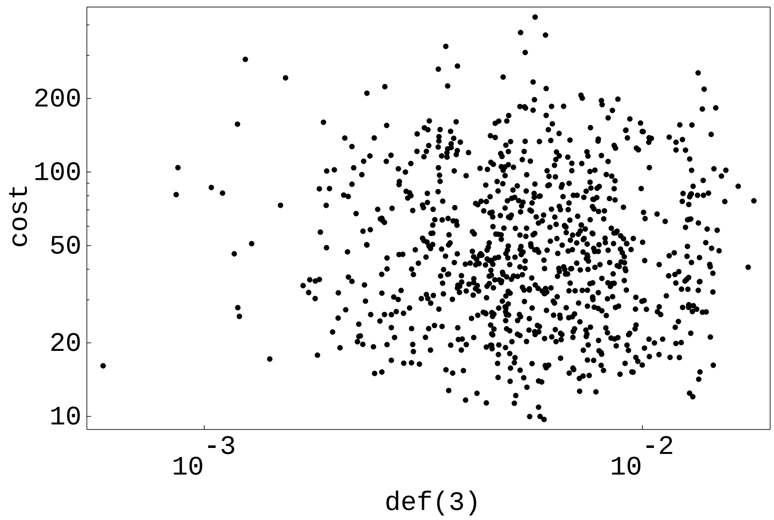


Figure 6: Average search cost of GSAT vs. def(3) for soluble cases of random 3-SAT instances with 20 variables and 80 clauses, on a log-log scale. Similar behavior is seen using def(4).

The small SAT problems examined here allow an exhaustive count of solutions, giving the behavior shown in Fig. 7. Again there is no correlation with the ApEn values. This suggests that further global characteristics of the search problem, determined with exhaustive analysis of state space, such as the number of local minima, would also not be correlated with the ApEn value.

These results on search cost and number of solutions suggest that no additional readily computable problem parameters will distinguish most instances with very different values of these properties within a single ensemble. Instead, much of the high variance is likely to be an intrinsic property of ensembles for which instances can be generated without intractable computations such as exponential search. Furthermore, the results are consistent with the observation mentioned above that rare exceptionally high cost cases in an ensemble are primarily due to algorithm choices rather than intrinsic instance complexity.

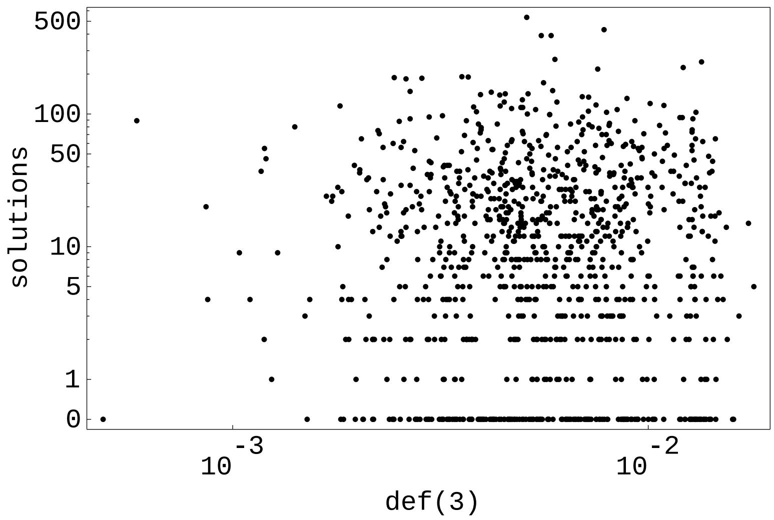


Figure 7: Number of solutions vs. def(3) for random 3-SAT on a log-log scale (except for the problems with 0 solutions included at the bottom of the plot). Similar behavior is seen using def(4).

# 6 Discussion

By distinguishing ensembles, ApEn may improve the use of the phase transition location as a heuristic [6, 10, 20]. For example, it could detect when choices appear to be typical of the ensemble used to design the heuristic, thus giving some indication of the reliability of the heuristic in different situations during the search. ApEn could also estimate the likely applicability of such heuristics based on different ensembles, either to select which one to use or to create [7] a portfolio [12, 16] of methods. ApEn's focus on problem instances complements probabilistic analyses of ensemble averages. Whether ApEn is useful in these contexts remains to be investigated.

ApEn can be generalized to a broader range of situations, such as search problems with continuous parameters, e.g., weights on links in a graph. Requiring exact matches between such values will not distinguish nearly equal values from significantly different ones. One approach to such cases is coarse graining, where the continuous parameter values are grouped into a few discrete bins. Alternatively, one can introduce a metric on the parts, and count the proportion of the parts within a given threshold distance of each one, as was studied for sequences [21].

Another modification, giving a more sensitive comparison of a problem instance and an ensemble, is replacing $\Phi^{(m)}$ by the disparity [7] $\sum f_i \ln(f_i/p_i)$ where $p_i$ is the ensemble probability that a size $m$ subproblem is of distinct type $i$. The disparity is nonnegative and equals zero only when $f_i = p_i$, i.e., the observed frequencies match those expected for the ensemble.

Applying approximate entropy to search has some arbitrariness in the choices of problem representation and the equivalence relation. With choices based on the deep structure of problems, ApEn readily discriminates ensembles even when the problems are too small for asymptotic behaviors. It thus provides a readily computable measure, directly relevant for search, that can help apply insights from ensemble analyses to individual problems.

# Acknowledgments

I thank Don Kimber and Rajan Lukose for helpful discussions.

# References

[1] C. Berge. *Graphs and Hypergraphs*. North-Holland, Amsterdam, 1973.

[2] B. Bollobas. *Random Graphs.* Academic Press, NY, 1985.

[3] Daniel Brelaz. New methods to color the vertices of a graph. *Communications of the ACM*, 22(4):251–256, 1979.

[4] G. Chaitin. Randomness and mathematical proof. *Scientific American*, 232:47–52, May 1975.

[5] Peter Cheeseman, Bob Kanefsky, and William M. Taylor. Where the really hard problems are. In J. Mylopoulos and R. Reiter, editors, *Proceedings of IJCAI91*, pages 331–337, San Mateo, CA, 1991. Morgan Kaufmann.

[6] Scott H. Clearwater and Tad Hogg. Problem structure heuristics and scaling behavior for genetic algorithms. *Artificial Intelligence*, 81:327–347, 1996.

[7] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory.* John Wiley, New York, 1991.

[8] James M. Crawford and Larry D. Auton. Experimental results on the crossover point in random 3SAT. *Artificial Intelligence*, 81:31–57, 1996.

[9] Rina Dechter. Enhancement schemes for constraint processing: Backjumping, learning, and cutset decomposition. *Artificial Intelligence*, 41:273–312, 1990.

[10] Ian P. Gent, Ewan MacIntyre, Patrick Prosser, and Toby Walsh. The constrainedness of search. In *Proc. of the 13th Natl. Conf. on Artificial Intelligence (AAAI96)*, pages 246–252, Menlo Park, CA, 1996. AAAI Press.

[11] Ian P. Gent and Toby Walsh. An empirical analysis of search in GSAT. *J. of AI Research*, 1:47–59, 1993.

[12] C. P. Gomes and B. Selman. Algorithm portfolio design: Theory vs. practice. In D. Geiger and P. Shenoy, editors, *Proc. of the 13th Conf. on Uncertainty in AI (UAI-97)*, pages 190–197, Los Altos, CA, 1997. Morgan Kaufmann.

[13] Tad Hogg. Refining the phase transitions in combinatorial search. *Artificial Intelligence*, 81:127–154, 1996.

[14] Tad Hogg, Bernardo A. Huberman, and Colin Williams. Phase transitions and the search problem. *Artificial Intelligence*, 81:1–15, 1996.

[15] B. A. Huberman and T. Hogg. Complexity and adaptation. *Physica*, 22D:376–384, 1986.

[16] Bernardo A. Huberman, Rajan M. Lukose, and Tad Hogg. An economics approach to hard computational problems. *Science*, 275:51–54, 1997.

[17] Donald E. Knuth. *The Art of Computer Programming.* Addison-Wesley, Reading, MA, 1969.

[18] Alan Mackworth. Constraint satisfaction. In S. Shapiro, editor, *Encyclopedia of Artificial Intelligence*, pages 285–293. Wiley, NY, 1992.

[19] A. Nijenhuis and H. S. Wilf. *Combinatorial Algorithms for Computers and Calculators.* Academic Press, New York, 2nd edition, 1978.

[20] David M. Pennock and Quentin F. Stout. Exploiting a theory of phase transitions in three-satisfiability problems. In *Proc. of the 13th Natl. Conf. on Artificial Intelligence (AAAI96)*, pages 253–258, Menlo Park, CA, 1996. AAAI Press.

[21] Steve Pincus and Rudolf E. Kalman. Not all (possibly) "random" sequences are created equal. *Proc. Natl. Acad. Sci. USA*, 94:3513–3518, 1997.

[22] Steve Pincus and Burton H. Singer. Randomness and degrees of irregularity. *Proc. Natl. Acad. Sci. USA*, 93:2083–2088, 1996.

[23] Patrick Prosser. Hybrid algorithms for the constraint satisfaction problem. *Computational Intelligence*, 9(3):268–299, 1993.

[24] Bart Selman and Scott Kirkpatrick. Critical behavior in the computational cost of satisfiability testing. *Artificial Intelligence*, 81:273–295, 1996.

[25] Bart Selman, Hector Levesque, and David Mitchell. A new method for solving hard satisfiability problems. In *Proc. of the 10th Natl. Conf. on Artificial Intelligence (AAAI92)*, pages 440–446, Menlo Park, CA, 1992. AAAI Press.

[26] Claude E. Shannon and Warren Weaver. *The Mathematical Theory of Communication*. Univ. of Illinois Press, Chicago, 1963.

[27] Dan R. Vlasie. The very particular structure of the very hard instances. In *Proc. of the 13th Natl. Conf. on Artificial Intelligence (AAAI96)*, pages 266–270, Menlo Park, CA, 1996. AAAI Press.

[28] Colin P. Williams and Tad Hogg. Exploiting the deep structure of constraint problems. *Artificial Intelligence*, 70:73–117, 1994.