The Constrainedness of Search

Ian P. Gent Patrick Prosser
Toby Walsh
Department of Computer Science
University of Strathclyde
Glasgow G1 1XH, Scotland
Email: {ipg,pat,tw}@cs.strath.ac.uk

February 16, 1999

Abstract

We propose a definition of 'constrainedness' that unifies two of the most common but informal uses of the term. These are that branching heuristics in search algorithms often try to make the most "constrained" choice, and that hard search problems tend to be "critically constrained". Our definition of constrainedness generalizes a number of parameters used to study phase transition behaviour in a wide variety of problem domains. As well as predicting the location of phase transitions in solubility, constrainedness provides insight into why problems at phase transitions tend to be hard to solve. Such problems are on a constrainedness "knife-edge", and we must search deep into the problem before they look more or less soluble. Heuristics that try to get off this knife-edge as quickly as possible by, for example, minimizing the constrainedness are often very effective. We show that heuristics from a wide variety of problem domains can be seen as minimizing the constrainedness (or proxies for it). Our definition is therefore useful both for studying phase transition behaviour and for developing new heuristics.

1 Introduction

Will a problem be soluble or insoluble? Will it be hard or easy? How can we develop heuristics for new problem domains? All these questions have been the subject of intensive study in recent years in a large number of problem domains including, for example, propositional satisfiability, graph colouring, constraint satisfaction problems, and hamiltonian circuits [7, 43, 51, 57]. Here, we introduce some general methods which help to answer these questions in a wide range of problems. These methods are based on a definition of the constrainedness of an ensemble of combinatorial problems. This definition unifies and generalizes a wide variety of parameters used to study phase transition behaviour in the past. We show that it predicts the location of a phase transition in solubility in a new problem domain, the asymmetric travelling salesperson problem. Measuring the constrainedness of problems during search also provides insight into why problems at phase transitions tend to be hard to solve. Such problems are on a constrainedness "knife-edge". Each successive branching decision gives a subproblem with a similar constrainedness as the original problem, neither more obviously soluble or insoluble. Only deep in search does the constrainedness eventually change and the problem look more or less soluble. Heuristics that try to get off this knife-edge as quickly as possible by, for example, minimizing the constrainedness are often therefore very effective.

The paper is structured as follows. We begin in Section 2 with a introduction to phase transition behaviour in search problems. In Section 3, we define the constrainedness of search problems. We then show how our definition generalizes parameter used in the past to locate phase transition behaviour in a wide variety of domains (Section 4), and in a new problem class, the asymmetric travelling salesperson problem (Section 5). To model phase transition behaviour at finite sizes, we borrow the technique of finite-size scaling from statistical mechanics to rescale our constrainedness parameter (Sections 6 and 7). We then show that problems at the phase transition are on a constrainedness "knife-edge" (Section 8) and that we can predict the shape of this knife-edge using a simple lower bound (Section 9). We investigate heuristics that try to get off this knife-edge as quickly as possible by minimizing and maximizing the constrainedness (Sections 10 and 11). We also show that many existing heuristics can be seen as minimizing constrainedness or proxies for it (Section 12). Finally, we describe related work (Section 13) and draw conclusions (Section 14).

2 Phase transitions

Many different search problems display phase transition behaviour [7, 23, 43]. Consider, for instance, colouring a graph with a fixed number of colours so that neighbouring nodes have different colours (see Figure 1). If the nodes in the graph are loosely connected, then problems tend to be soluble and it is usually easy to guess one of the many solutions. If nodes are highly connected, then problems tend to be insoluble and it is usually easy to identify why we have too few colours. At intermediate levels of connectivity, problems can be hard to solve since they are neither obviously soluble nor insoluble. For ensembles of randomly generated graphs, there is a rapid transition between soluble and insoluble problems as we vary their connectivity, with the hardest graph colouring problems tending to occur around the transition [7].

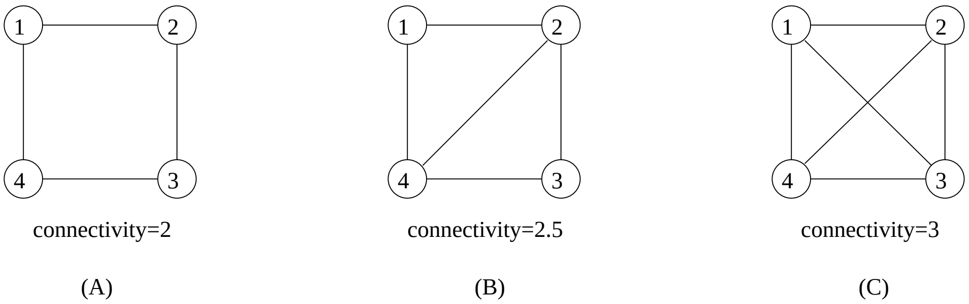

Figure 1. Colouring graphs with four nodes using three colours: red, blue and green. Nodes connected by an edge must have different colours. The connectivity of a node is the number of edges connected to the node. The connectivity of a graph is the average connectivity of its nodes. **(A)** An under-constrained and soluble problem that requires only two of the three colours to solve. **(B)** A problem which is just soluble. It has an unique solution up to symmetry. **(C)** An over-constrained and insoluble problem consisting of a clique of four nodes. This requires more than the permitted three colours.

We can use connectivity to develop a simple but effective heuristic for graph colouring that colours the most constrained nodes first. The motivation is to work on the hardest part of the problem first. Consider colouring the nodes in Figure 1B without such a heuristic, using instead their numerical order. We might colour node 1 red, then node 2 blue, and node 3 green. We would then be unable to colour node 4 without giving it the same colour as one of its neighbours. Instead, suppose we seek to colour the most constrained nodes first. Both informally and, as we show later, under our formal definition, the constrainedness of a graph is directly related to its connectivity. This then suggests the heuristic of colouring the nodes in decreasing order of their connectivity. As nodes 2 and 4 have the highest connectivity, they are coloured first. If we colour node 2 red and node 4 blue, then nodes 1 and 3 can be coloured green. Ordering the nodes by their connectivity focuses on the hardest part of the problem, leaving the less constrained and easier parts till last.

3 Constrainedness

Given a new search problem, how do we identify parameters like connectivity which measure the constrainedness and which can be used to develop heuristics for finding a solution? This is a problem that has challenged other researchers. For example, Minton asks:

> *".. It is instructive to consider what sort of theory would be required to be able to* prove *that, in any given circumstance, a variable is most likely to be most constraining .."* [author's emphasis] p.861 [42]

Minton argues that this is too hard a problem to solve analytically, and proposes instead an empirical approach. For example, the MULTI-TAC system constructs a set of candidate branching rules and benchmarks them to determine their effectiveness as heuristics on a representative set of problems [41]. Our approach to this problem is more analytical and is motivated by studies of transitions in solubility. By comparing the parameters introduced in a variety of domains like graph colouring and satisfiability, we are able to propose a general definition of the "constrainedness" of an ensemble of problems.

We assume that each problem in an ensemble has a state space S with $|S|$ elements and a number, Sol of these states are solutions. Any point in the state space can be represented by a N-bit binary vector where $N = \log_2(|S|)$. This logarithm provides us with a convenient measure of problem size. The constrainedness of an ensemble of problems depends on both the problem size and the expected number of solutions. Problems that are expected to have a large number of solutions for their size are under-constrained. Problems that are expected to have no or just a few solutions for their size are over-constrained. Let $\langle Sol \rangle$ be the expected number of states that are solutions averaged over each problem in the ensemble. We define the constrainedness, κ, of an ensemble by,

$$\kappa \quad =_{\text{def}} \quad 1 - \frac{\log_2(\langle Sol \rangle)}{N} \tag{1}$$

As it is often relatively straightforward to compute or, at least, estimate $\langle Sol \rangle$, we can often determine the constrainedness κ without great difficulty. It is very important to note that this definition is for the constrainedness of an *ensemble* of problems, not of an individual problem. In following sections, we will show that this definition generalizes, unifies, and extends a large body of previous work on phase transition behaviour in randomly generated problems. In most of these studies, problems are generated by some well defined procedure. This defines an ensemble of problems and hence the constrainedness κ. Recently, Hogg has proposed a method based on computing the approximate entropy for estimating from which ensemble a problem instance is likely to be drawn [27]. Such a method can be used to estimate κ when presented with just a single problem instance.

An alternative method to calculate κ is in terms of ρ, the probability that a randomly selected state is a solution. Now $\rho = \langle Sol \rangle / 2^N$. Hence $\langle Sol \rangle = \rho.2^N$. Thus an alternative, and often more convenient definition is,

$$\kappa \quad =_{\text{def}} \quad -\frac{\log_2(\rho)}{N}$$

Given that $\rho \in [0, 1]$ and $N > 0$, the constrainedness, κ lies in the range $[0, \infty)$. A value of $\kappa = 0$ corresponds to a completely unconstrained ensemble in which every state is expected to be a solution, $\langle Sol \rangle = |S| = 2^N$. A value of $\kappa = \infty$ corresponds to a completely constrained ensemble in which no states are expected to be solutions, $\langle Sol \rangle = 0$. In constraint satisfaction, a phase transition between soluble and insoluble problems has been predicted when $\langle Sol \rangle \approx 1$ [57, 51]. Generalizing this prediction using (1), we predict that a phase transition occurs when $\kappa \approx 1$. If $\kappa < 1$, problems are under-constrained and are typically soluble. If $\kappa > 1$, problems are over-constrained and are typically insoluble. The equality $\kappa \approx 1$ only gives a first approximation of the location of the phase transition: we will see that κ is typically between about 0.5 and 1 at the phase transition. More refined estimates can be achieved either by taking account of the variance in the number of solutions at the phase boundary [57, 51], or by finding an equivalent problem with fewer solutions at its phase transition [32, 13].

There is a subtle difference between the prediction of a phase transition at $\kappa \approx 1$ and at $\langle Sol \rangle \approx 1$. While $\langle Sol \rangle$ at the phase transition can grow exponentially with N, the value of κ tends to vary more slowly. For example, with random 3-SAT problems, the expected number of solutions at the phase boundary grows as approximately $2^{0.18\mathrm{N}}$ [33]. By comparison, κ at the phase boundary tends to vary much more slowly. As we show later, the variation in κ at the phase boundary can be modelled by the technique of finite size scaling using a low-order polynomial in N. In addition, as we show in the next section, κ subsumes "order parameters" used by a number of different authors to identify phase transition behaviour in a variety of problem classes. For the first time, we can see that these assorted parameters all measure the same thing.

4 Comparison with existing parameters

This definition of constrainedness generalizes parameters introduced to study phase transition behaviour in propositional satisfiability, constraint satisfaction, graph colouring, number partitioning and hamiltonian circuit problems. We predict that it will prove useful in many other domains.

4.1 Satisfiability

In propositional satisfiability (or SAT), we are given a formula with n variables and l clauses and wish to find an assignment of truth values to the variables such that each clause is satisfied. A clause of length i rules out just one of the 2^i possible tuples of values of the variables in the clause. The assignment ruled out sets each literal in the clause to false. A clause of length i therefore permits a fraction $(1 - \frac{1}{2^i})$ of the 2^n possible truth assignments. If there are l_i clauses of length i then, assuming that the clauses are independently generated,

$$\langle Sol \rangle \quad = \quad 2^n . \prod_i (1 - \frac{1}{2^i})^{l_i}$$

Hence,

$$\kappa \quad = \quad - \sum_i \frac{l_i}{n} . \log_2(1 - \frac{1}{2^i}) \qquad (2)$$

If all clauses are of a single length k (as in the random k-SAT problem class) then,

$$\kappa \quad = \quad -\frac{l}{n} . \log_2(1 - \frac{1}{2^k})$$

Note that this is directly proportional to l/n, the ratio of clauses to variables. The ratio l/n has been used as an "order parameter" for studying phase transitions in many different problem classes as a phase transition in solubility often occurs around a critical value of l/n [43, 21].

For random 2-SAT, the phase transition has been proven to occur at $l/n = 1$ [8, 25], which corresponds to $\kappa \approx 0.42$. For random 3-SAT, the phase transition has been shown to occur experimentally around $l/n = 4.3$ [43, 10], which corresponds to $\kappa \approx 0.82$. Theoretical bounds put the phase transition for random 3-SAT in the interval $3.003 < l/n < 4.598$ [15, 34]. This corresponds to the interval $0.58 < \kappa < 0.89$. For random 4-SAT, the phase transition has been shown to occur experimentally around $l/n = 9.8$ [21], which corresponds to $\kappa \approx 0.91$. For large k, the phase transition for random k-SAT occurs at a value of l/n close to $-1/\log_2(1 - \frac{1}{2^k})$ [33]. This corresponds to $\kappa \approx 1$ as predicted.

Phase transition behaviour has also been observed in satisfiability problem classes with clauses of mixed lengths. In the constant probability model, a literal is included in a clause with probability p, independently of the inclusion of other literals. As empty and unit clauses typically makes problems much easier, Hooker and Fedjki have proposed that such clauses should be discarded and longer clauses generated in their place [30]. Gent and Walsh show that if the expected clause length is kept roughly constant by varying p as $1/n$ then the solubility phase transition occurs around a constant value of l/n [21]. They approximate the binomial distribution of clause lengths by a Poisson distribution with parameter $2np$ adjusting for the omission of empty and unit clauses. As in [21], we let $2np = 3$ to give problems with an average clause length of 3 before removal of empty and unit clauses. This gives,

$$\kappa \approx -\frac{l}{n} \log_2(1 - \frac{e^{-3/2} - \Phi(0) - \frac{1}{2}\Phi(0)}{1 - \Phi(0) - \Phi(1)})$$

where $\Phi(k) = e^{-3}3^k/k!$. The phase transition for this problem class occurs around $l/n = 2.80$, which corresponds to $\kappa \approx 0.53$.

In [21], Gent and Walsh propose the mixed SAT model in which l clauses in n variables are generated with respect to a probability distribution $\phi(i)$ on their length i. For example, in the 2-3-SAT model, $\phi(2) = \phi(3) = \frac{1}{2}$ and binary and ternary clauses are generated with equal probability. The frequency of occurrence of an integer in the name of a mixed SAT model reflects the frequency of occurrence of clauses of this length in the problem. Hence in the 2-4-4-SAT problem class, $\phi(2) = \frac{1}{3}$ and $\phi(4) = \frac{2}{3}$, and clauses of length 2 appear with probability $\frac{1}{3}$ and of length 4 with probability $\frac{2}{3}$. Gent and Walsh conjecture that the location of the phase transition for mixed SAT problems is approximated by the parallel sum,

$$\frac{1}{c} \approx \sum_i \frac{\phi(i)}{c_i}$$

where c is the ratio of l/n at the mixed SAT phase transition and c_k is the ratio of l/n at the random k-SAT phase transition [21]. For example, the phase transition in random 2-3-SAT occurs around $l/n = 1.76$ compared to a parallel sum prediction of 1.62. As another example, the phase transition in random 2-4-4-SAT occurs around $l/n = 2.74$ compared to a prediction of 2.49. As a final example, the phase transition in random 3-4-SAT occurs around $l/n = 5.88$ compared to a prediction of 5.91.

For the mixed SAT problem class, a simple calculation gives,

$$\kappa = -\frac{l}{n}.\sum_i \log_2(1 - \frac{1}{2^i}).\phi(i)$$

As usual, we predict the phase transition in solubility around $\kappa \approx 1$. This prediction improves as the length of clauses in the mixed SAT problems increases. For example, the phase transition in random 2-3-SAT occurs around $\kappa = 0.53$, in random 2-4-4-SAT around $\kappa = 0.55$, and in random 3-4-SAT around $\kappa = 0.84$. Note that $-1/\log_2(1 - \frac{1}{2^k})$ is the approximate location of the random k-SAT phase transition. Hence,

$$\kappa \approx \frac{l}{n}.\sum_i \frac{\phi(i)}{c_i}$$

This adds support to Gent and Walsh's parallel sum conjecture for the location of the mixed-SAT phase transition.

4.2 Graph colouring

In graph colouring, we are given a graph with n nodes and e edges, and wish to colour it with m colours so that neighbouring nodes have different colours. Each edge rules out m of the m^2 possible pairs of colours for the nodes at either end of the edge. That is, each edge permits a fraction $(1-\frac{1}{m})$ of the m^n possible colourings. Assuming that the edges are independently generated,

$$\langle Sol \rangle \quad = \quad m^n.(1-\frac{1}{m})^e$$

Hence,

$$\kappa = \frac{e}{n}\log_m(\frac{m}{m-1})$$

This is a constant, namely $\log_m(\frac{m}{m-1})/2$, times the average degree of a node in the graph. The average degree has been used as an order parameter for describing the phase transition in colouring problems [7]. A phase transition has been observed in random 3-colouring problems at an average degree of 4.6 [28], corresponding to $\kappa = 0.84$. In random 4-colouring problems, the phase transition occurs around an average degree of 8.7 [53], corresponding to $\kappa = 0.90$. In random 5-colouring problems, the phase transition occurs around an average degree of 13.1 [53], corresponding to $\kappa = 0.91$.

4.3 Constraint satisfaction

A constraint satisfaction problem (CSP) consists of a set of variables V and a set of constraints C, and we wish to find values for the variables so that the constraints are not violated. Each variable $v \in V$, has a domain of values M_v of size m_v. Each constraint $c \in C$ of arity a restricts a tuple of variables $\langle v_1, \ldots, v_a \rangle$, and rules out some proportion p_c of possible values from the cartesian product $M_{v_1} \times \ldots \times M_{v_a}$. We call p_c the "tightness" of a constraint. To avoid trivial problems we insist that all arities are at least one, but make no further restrictions. Problems may have variables with many different domain sizes, and constraints of many different arities and tightnesses.

The state space has size $\prod_{v \in V} m_v$. Each constraint rules out a proportion p_c of these states, so we have

$$\langle Sol \rangle = (\prod_{v \in V} m_v) \times (\prod_{c \in C}(1-p_c))$$

Substituting this into (1) gives

$$\kappa \quad = \quad \frac{-\sum_{c \in C}\log_2(1-p_c)}{\sum_{v \in V}\log_2(m_v)} \tag{3}$$

In binary CSP's (in which constraints only have a binary arity), a standard means of generating test problems is to have n variables each with the same domain size of m. Given a constraint density of p_1, exactly $p_1 n(n-1)/2$ constraints are chosen, each with a tightness of p_2 [47, 51]. Such problems are described by the tuple, $\langle n, m, p_1, p_2 \rangle$. Using these values, (3) gives

$$\kappa = \frac{n-1}{2} p_1 \log_m\left(\frac{1}{1-p_2}\right)$$

This has been used as a parameter for binary constraint satisfaction problems [20]. For $\langle 20, 10, p_1, p_2 \rangle$ problems, the phase transition occurs between $0.75 \leq \kappa \leq 1$. For $\langle n, 3, p_1, 2/9 \rangle$ problems (which resemble 3-colouring problems in having three values), the phase transition occurs around $\kappa \approx 0.62$. For $\langle 10, m, 1, p_2 \rangle$ problems, the phase transition occurs around $\kappa \approx 1.02$. For $\langle n, n, 1, p_2 \rangle$ problems (which have a complete constraint graph like n-queens problems), the phase transition occurs around $\kappa \approx 0.99$.

4.4 Number partitioning

In 2-way number partitioning, we have n numbers drawn uniformly and at random from the range $(0, l]$ and wish to find a partition into two bags with the same sum. To study this problem, Gent and Walsh have developed an "annealed" theory [22, 24] in which they average probabilities independently over the different binary digit positions. They call this an annealed theory by analogy with the annealed theory of materials which averages independently over sources of disorder. Both give good approximations in the limit.

We expect 1/2 the possible partitions of the n numbers to add up to a particular parity in any binary digit position. Assuming independence between the $\log_2(l)$ digit positions, a fraction $(1/2)^{log_2(l)}$ of the 2^n partitions will add up to the same sum. The expected number of exact partitions is therefore

$$\langle Sol \rangle = 2^n . \left(\frac{1}{2}\right)^{\log_2(l)}$$

This gives

$$\kappa = \frac{\log_2(l)}{n}.$$

A phase transition in the probability of a perfect 2-way partition has been observed around $\kappa = 0.96$ [22, 24]. Using some complex analysis based on statistical thermodynamics, Mertens predicts the location of this phase transition around a fixed value of a parameter given by $\kappa + O(1/nl)$ [40].

In m-way number partitioning, we have n numbers drawn uniformly and at random from the range $(0, l]$ and wish to find a partition into m bags with the same sum. As there are m^n possible partitions of n numbers into m bags, $N = n \log_2 m$ We assume that the numbers have a sum which is an exact multiple of m. A similar but slightly more complex argument can be given when the sum is not an exact multiple. At each digit position, the first $m-1$ bags must each have a given sum modulo 2. This is expected to happen with probability $p = (1/2)^{m-1}$ as there is a 50% chance that any pair have the same sum modulo 2. The last bag is guaranteed to have the right digit sum if the first $m-1$ do so we can ignore it. Assuming independence between the $\log_2(l)$ binary digit positions, a fraction $p^{\log_2(l)}$ of the m^n partitions will add up to the same sum. The expected number of perfect partitions is,

$$\langle Sol \rangle \;=\; m^n.(\frac{1}{2})^{(m-1)\log_2(l)}.$$

Hence,

$$\kappa \;=\; \frac{(m-1)\log_m(l)}{n}.$$

In [22], Gent and Walsh identify a sharp phase transition in the probability of a perfect 3-way partition again around $\kappa = 0.96$.

4.5 Hamiltonian circuits

Given an undirected graph, we wish to decide whether there is an ordered sequence of nodes such that each pair of nodes in the sequence is connected by an edge, as well as the first and last nodes. Here, we consider graphs with n nodes, and e edges distributed randomly through the graph. Frank, Gent and Walsh have calculated κ for this ensemble [14]. The probability that the first edge of any circuit is in the graph is $2e/n(n-1)$. That leaves $e-1$ edges and $(n(n-1)/2)-1$ places to put them. So the next edge is in the graph with chance $(e-1)/[(n(n-1)/2)-1]$. Then the next with chance $(e-2)/[(n(n-1)/2)-2]$. Since there are n edges under investigation we get an overall probability of

$$\rho = \prod_{i=0}^{n-1} \frac{(e-i)}{(n(n-1)/2)-i}$$

Since we may designate a starting point arbitrarily, and because we may take circuits in either direction, the number of distinct potential circuits is $(n-1)!/2$. Hence, $\langle Sol \rangle$ is $\rho(n-1)!/2$ and N is $\log_2((n-1)!/2)$. Thus,

$$\kappa = -\frac{\sum_{i=0}^{n-1} \log_2 \frac{(e-i)}{(n(n-1)/2)-i}}{\log_2((n-1)!/2)} \tag{4}$$

For graphs with up to 30 nodes, the phase transition occurs around $\kappa \approx 0.7$. Koršunov [36, 37] proves that if we fix $e/(n\log n)$, then as $n \to \infty$, there is almost certainly a circuit if $e/(n\log n) > \frac{1}{2}$, and almost certainly not if $e/(n\log n) < \frac{1}{2}$. This suggests that $e/(n\log n)$ is a more suitable parameter than κ. However, Frank *et al.* show that the difference between the two is slight for the size of problems considered in [14]. This suggests that constrainedness can be a useful parameter even in problem classes where it is asymptotically incorrect. It is interesting to speculate that the discrepancy between the two parameters may be related to the fact that hard problems do not seem to occur at the phase transition [55].

4.6 Comparing domains

We thus see that our definition of κ generalizes a number of parameters introduced in a variety of problem classes. This suggests that "constrainedness" is a fundamental property of problem ensembles. In addition to unifying existing parameters, we can now compare problems between classes. For example, the phase transition in 3-SAT problems occurs at $l/n = 4.3$ [43, 10] which corresponds to $\kappa \approx 0.82$, roughly comparable to that in 3-colouring at $\kappa \approx 0.84$, while the phase transition in number partitioning occurs at $\kappa \approx 0.96$. This suggests that number partitioning problems at the phase transition may in some sense be more constrained. Computational results appear to support this claim. For example, with problems at the phase transition CKK, one of the best algorithms known for number partitioning [35], runs on average as approximately $2^{0.85N}$ [24] whilst Crawford's TABLEAU algorithm, one of the best algorithms known for satisfiability, runs on average as approximately $2^{0.05N}$ [11]. The definition of κ also allows us to treat a wider range of problems within a class. For example, we now deal with problems having mixed arity constraints, mixed domain sizes and mixed constraint tightnesses. This permits the computation of κ during search as domain sizes change and constraints are removed. We will see the value of this in a later section.

5 The travelling salesperson problem

We have shown that this definition of constrainedness generalizes parameters previously introduced in a variety of problem classes, including satisfiability, graph colouring, constraint satisfaction, and number partitioning. We now give a case study which uses this definition of constrainedness in a new problem class. We consider the asymmetric travelling salesperson problem (ATSP) with inter-city distances drawn from a normal distribution with mean μ and standard deviation σ. We focus on the decision problem of determining if there is a tour of length d or less which visits all n cities. Most computational studies of the travelling salesperson problem have been on the optimisation rather than the decision problem [7, 58]. Although a phase transition has been observed for the decision problem of the two-dimensional Euclidean travelling salesperson problem [23], the parameter used was based on an asymptotic result and we do not understand its relation to constrainedness.

The state space S contains all $(n-1)!$ possible distinct tours (one city is designated the starting point arbitrarily). Each of these tours has some length l. As the sum of n normal distributions, l has a normal distribution with mean $n\mu$ and standard deviation $\sigma\sqrt{n}$. If we normalize l to $\hat{l} = (l - n\mu)/\sigma\sqrt{n}$ then $\hat{l}$ is distributed normally with mean 0 and standard deviation 1. The probability that a randomly chosen tour has a length l less than or equal to some given length d is

$$prob(l \leq d) \quad = \quad \int_{-\infty}^{\hat{d}} \frac{e^{-x^2/2}}{\sqrt{2\pi}} dx$$

A standard handbook of integrals [1] gives the equality

$$\int_{-\infty}^{z} \frac{e^{-x^2/2}}{\sqrt{2\pi}} dx \quad = \quad \frac{e^{-z^2/2}}{\sqrt{2\pi}} \left(\frac{1}{|z|} + O(\frac{1}{|z|^3}) \right)$$

The optimal tour length will tend to have $\hat{d} << 0$ so the error term will be small. Accordingly we use the approximation

$$prob(l \leq d) \approx \frac{e^{-\hat{d}^2/2}}{|\hat{d}|\sqrt{2\pi}}$$

Multiplying this by $(n-1)!$, the number of distinct tours, gives $\langle Sol \rangle$, the expected number of tours less than or equal to d. Substituting this into (1) gives,

$$\kappa \quad = \quad \frac{\hat{d}^2 \log_2(e)/2 + \log_2(|\hat{d}|\sqrt{2\pi})}{\log_2(n-1)!}$$

Although further approximations could be made, for example using Stirling's approximation, this definition can easily be calculated numerically by computer.

We expect a phase transition in the decision problem when $\kappa \approx 1$. We tested this experimentally using a branch and bound algorithm with the Hungarian heuristic for branching [6]. For n=6 to 48, we randomly generated 1000 problems with inter-city distances independently and normally distributed with μ=10^6 and σ=10^5. Figure 2 shows the probability that there was a tour less than distance d, plotted against κ. There is a clear phase transition from soluble to insoluble problems that becomes sharper with more cities. Except for problems with 6 cities, there is a critical value of $\kappa = 0.75$ which gives the probability of a tour existing of 0.45 ± 0.04 at all sizes.

6 Finite size scaling

We can use the constrainedness, κ, to predict the shape as well as the location of phase transitions. Phase transitions in physical systems have been successfully

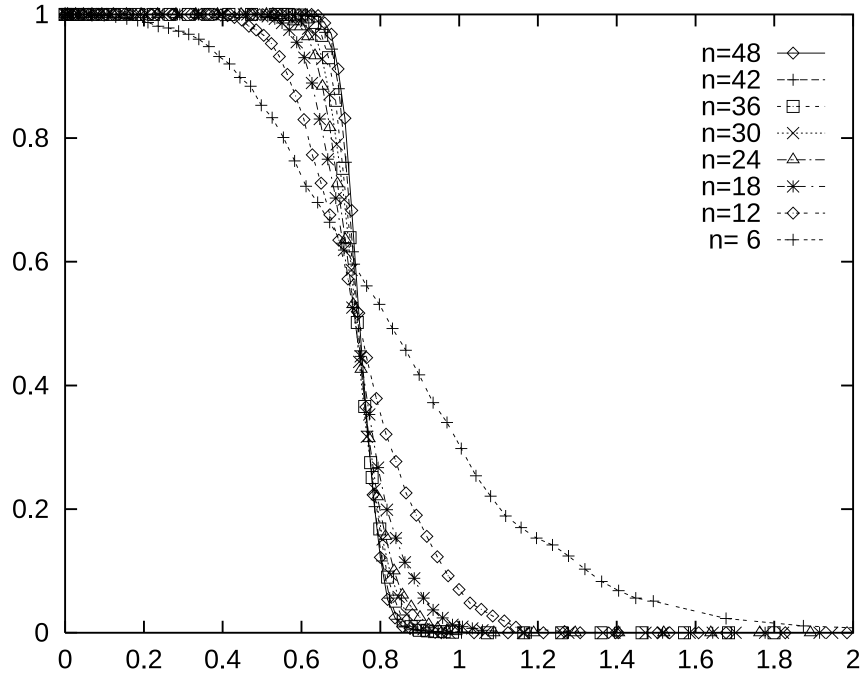

Figure 2. Probability of tour of required length existing in ATSP, plotted against κ for 6 to 48 cities.

described using finite size scaling methods [3]. Around a critical temperature T_c, problems of all sizes tend to be indistinguishable except for a change of scale given by a power law in a characteristic length. Here we propose that the constrainedness, κ, plays the role of temperature whilst the problem size, N, plays the role of the characteristic length. This analogy suggests that around some critical constrainedness κ_c, problems of all sizes will tend to be indistinguishable except for a simple change of scale given by a power law in N. For example, we conjecture that a macroscopic property like the probability of a solution existing averaged over an ensemble of problems will obey the equation,

$$prob(Sol > 0) = f(\frac{\kappa - \kappa_c}{\kappa_c} \,.\, N^{1/\nu}) \tag{5}$$

where f is some fundamental function, $\frac{\kappa-\kappa_c}{\kappa_c}$ is analogous to the reduced temperature $\frac{T-T_c}{T_c}$, and $N^{1/\nu}$ provides the change of scale. Such scaling has been shown to model the probability of a solution in finite size phase transitions in satisfiability [33], constraint satisfaction [20], and number partitioning [22]. In physical systems, the size dependency usually depends on a correlation length. Points separated by more than the correlation length behave independently. Despite the fact that lengths appear in travelling salesperson problems, the size dependency obeys a simple power law with the problem size, N. Similar power law behaviour in N has been seen in satisfiability [33, 21], constraint satisfaction [20], and number partitioning problems [22, 24].

To test this scaling conjecture for the ATSP, in Figure 3 we replot our data against the parameter $\frac{\kappa-\kappa_c}{\kappa_c} \,.\, N^{1/\nu}$ using $\kappa_c = 0.75$ and $\nu = 2$, both values derived from examination of the data. If (5) holds, the curves will line up when plotted against this rescaled parameter. As predicted, except at $n = 6$, finite size scaling models the probability of a tour existing. A discrepancy at small problem sizes has also been seen in other classes such as satisfiability [33] and suggests that finite size scaling provides a very useful but incomplete description of scaling behaviour.

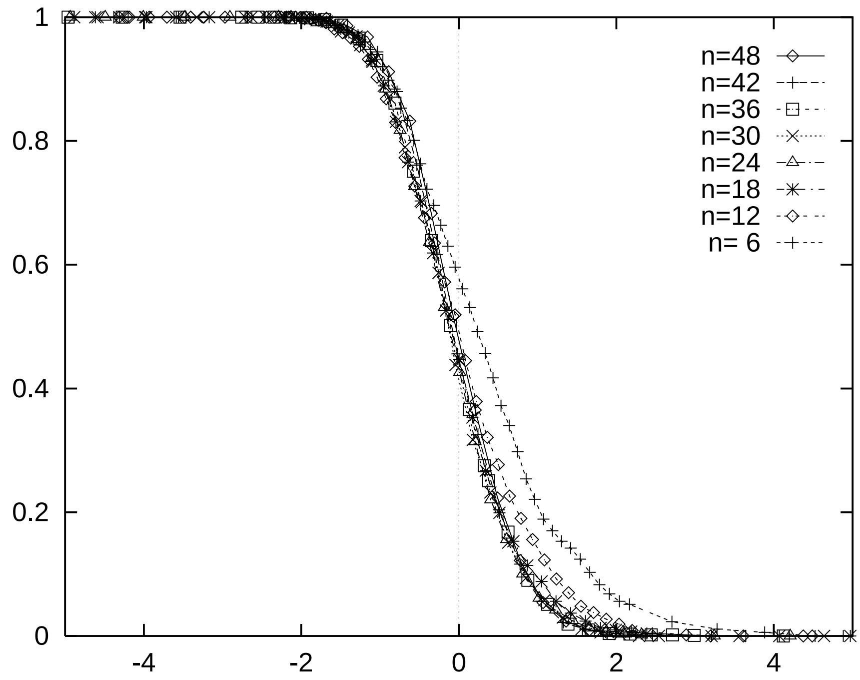

Figure 3. Probability of tour of required length existing in ATSP, against $\frac{\kappa-\kappa_c}{\kappa_c} \,.\, N^{1/\nu}$ for 6 to 48 cities.

This case study clearly illustrates that our definition of constrainedness is useful in new problem classes. A phase transition occurs, as predicted, at $\kappa \approx 1$. And as expected, by means of finite size scaling we are able to model scaling behaviour of the phase transition.

7 Search cost

Modelling using finite size scaling can be applied to measures of algorithmic behaviour such as search cost [49, 20]. For the ATSP, we use the 90th percentile of the number of leaf nodes searched, as lower percentiles such as median cost were always trivial since almost no backtracking occurred. As in many other problem classes, e.g. satisfiability [43], search cost displays a distinctive easy-hard-easy pattern through the phase transition. We fitted the data to a simple model of exponential growth. We put aside the extremes of our data, i.e. n=6 and 48, and constructed a model for values of the rescaled parameter from -3 to 3 in steps of 0.25 using n=12 to 42. We then compared the modelled data with the observed, as well as the search costs the model predicts at 6 and 48 cities. Figure 7 shows the accuracy of the modelling and predictions. For example, the model successfully predicts that at $n = 6$ we will explore just a single leaf node throughout the phase transition. It is remarkable that a simple model of exponential growth does so well when search costs never reaches very large values, and granularity therefore plays an important role. [23] reports a phase transition in the Euclidean travelling salesperson problem, but the parameter used there, $\hat{d}/\sqrt{n}$ was derived from asymptotic considerations and not from κ. It would be interesting to see if modelling of search cost for Euclidean problems works well against this parameter. More generally, it would be valuable to combine this kind of analysis of behaviour as problem size scales, with analyses which study the distribution of search costs at a single problem size, such as those carried out by Frost, Rish and Villa, and Hoos and Stützle [16, 48, 31].

8 Constrainedness within search

A general rule of thumb in solving search problems is to tackle the hardest part first. Many heuristics therefore try to branch on the most constrained variable. To test their effectiveness at this, we measured the constrainedness of a problem during search. We ran experiments in several different domains, using both random and non-random problems. In each case, we observe a constrainedness "knife-edge" in which critically constrained problems tend to remain critically constrained. Here we report the results for propositional satisfiability. However, results were similar in the other domains which included graph colouring and number partitioning [56].

We use the Davis-Putnam procedure with unit propagation but no pure literal deletion. We branch with MOM's heuristic, picking the literal that occurs most often in the minimal size clauses, breaking ties with a static numerical order. Depth is measured by the number of assignments. Similar results are obtained when depth is measured by the number of branch points, and with other branching heuristics including random branching. In each experiment, we simply follow the heuristic down the first branch, averaging over 1000 problems. To reduce variance, we use the same ensemble of problems in all experiments. We adopt the convention that initial parameters are in capitals and that values measured during search are in lower case. For instance, if we generate random 3-SAT prob-

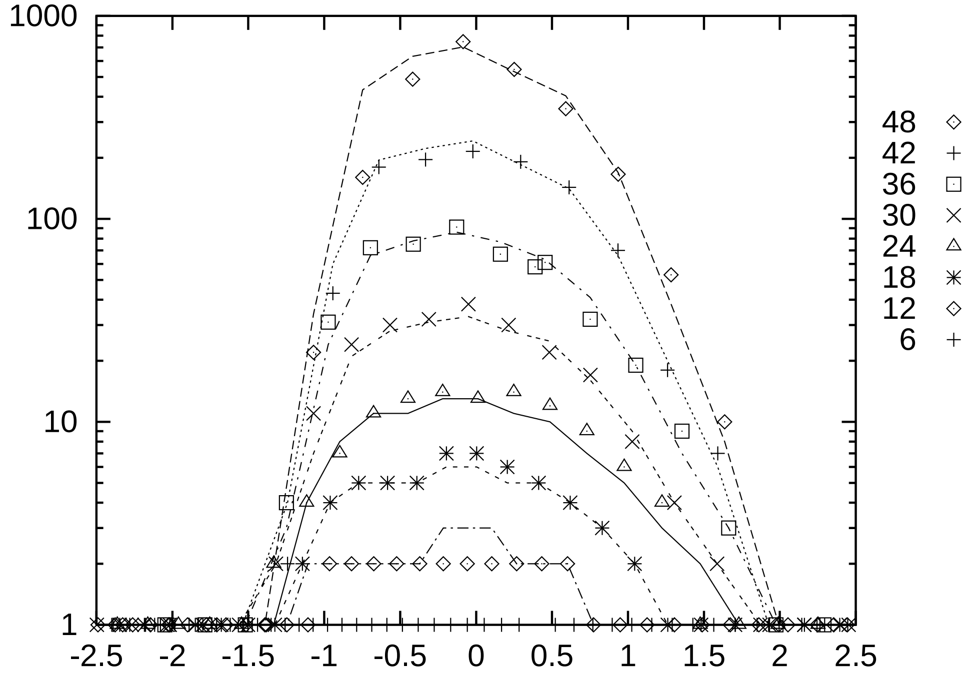

Figure 4. Modelled/predicted (lines) and observed (points) 90th percentile of leaf nodes searched to solve ATSP instances (both on y-axis) against the rescaled parameter $\frac{\kappa - \kappa_c}{\kappa_c} . N^{1/\nu}$ (x-axis) with $\kappa_c = 0.75$, $\nu = 2$.

lems from the middle of the phase transition with an initial ratio of clauses to variables, L/N of 4.3 then during search, the average clause length, k, and ratio of clauses to variables, l/n, may be different from their starting values of 3 and 4.3 respectively.

In Figure 5, we plot the ratio of clauses to variables during search for random 3-SAT problems from the middle of the phase transition with an initial clause to variable ratio, $L/N = 4.3$. Since not all heuristic branches extend to large depths, there is some noise at the end of each graph. As search deepens, the ratio of clauses to variables drops approximately linearly. The gradient of this decay is inversely proportional to N. Plotting l/n against the fractional depth, i/N therefore gives graphs of similar slope despite N varying from 100 to 500. Other experiments show that the rate of decay increases as we increase the initial ratio of clauses to variables, L/N.

These results might seem to suggest that problems become less constrained as search progresses. However, the average clause length also decreases, and shorter clauses will tighten the constrainedness of problems. In Figure 6, we plot the average clause length during search for the same random 3-SAT problems from the middle of the phase transition. Again, we see an approximate linear decay, with the gradient inversely proportional to N. Other experiments show that

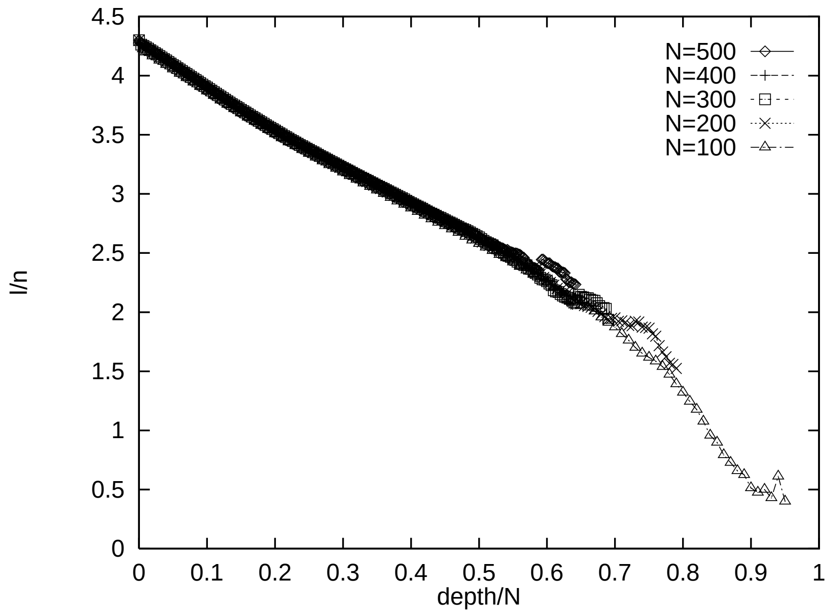

Figure 5. Ratio of clauses to variables, l/n on the heuristic branch (y-axis) against the fractional depth (x-axis) for random 3-SAT problems with an initial ratio of clauses to variables, $L/N = 4.3$.

the average clause length decreases as we decrease the initial ratio of clauses to variables, L/N. Which of these two factors wins? Does the decrease in clause size tighten the constrainedness faster than the decrease in the ratio of clauses to variables loosens it? To answer these questions, we estimated κ during search by assuming that the current subproblem is taken from a random ensemble in which problems have the same number of clauses, the same number of variables, and the same distribution of clause lengths. We can then use (2) to estimate κ.

In Figure 7, we plot the estimated constrainedness down the heuristic branch for random 3-SAT problems. For $L/N < 4.3$, problems are under-constrained and soluble. As search progresses, κ decreases since problems become more under-constrained and obviously soluble. For $L/N > 4.3$, problems are over-constrained and insoluble. As search progresses, κ increases since problems become more over-constrained and obviously insoluble. At $L/N \approx 4.3$ problems are on the *knife-edge* between solubility and insolubility. As search progresses, κ is roughly constant. Each successive branching decision gives a subproblem which has the same constrainedness as the original problem, neither more obviously satisfiable, nor more obviously unsatisfiable. Only deep in search does κ eventually break one way or the other.

We have also observed similar knife-edge behaviour with a random heuristic, and with an anti-heuristic (that is, one which always branching against the heur-

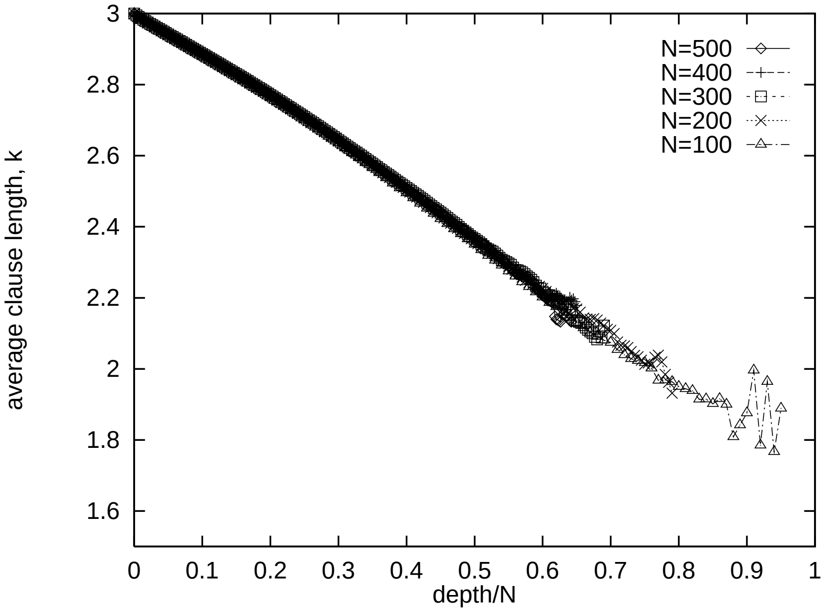

Figure 6. Average clause length, k on the heuristic branch (y-axis) against the fractional depth (x-axis) for random 3-SAT problems with an initial ratio of clauses to variables, $L/N = 4.3$.

istic) except that values of κ are slightly greater. In addition, we observed similar knife-edge behaviour in other problems domains including graph colouring and number partitioning [56]. Figure 7 suggests an interesting analogy with statistical mechanics. At the phase boundary in physical systems, problems tend to be "self-similar". That is, they look similar at every length scale. At the phase boundary in computational systems, problems also display a form of self-similarity. Branching decisions give subproblems that look neither more or less constrained. This helps to explain why such problems are difficult to solve. Branching decisions tell us very little about the problem, giving subproblems that are neither more obviously soluble nor more obviously insoluble. We will often have to search to a large depth either for a solution or for a refutation. By comparison, branching on an over-constrained problem gives a subproblem that is often even more constrained and hopefully easier to show insoluble, whilst branching on an under-constrained problem gives a subproblem that is often even less constrained and hopefully easier to solve.

9 Lower bound on constrainedness

When we branch into a subproblem, the number of solutions remaining cannot increase. The expected number of solutions, $\langle Sol \rangle$ cannot therefore increase.

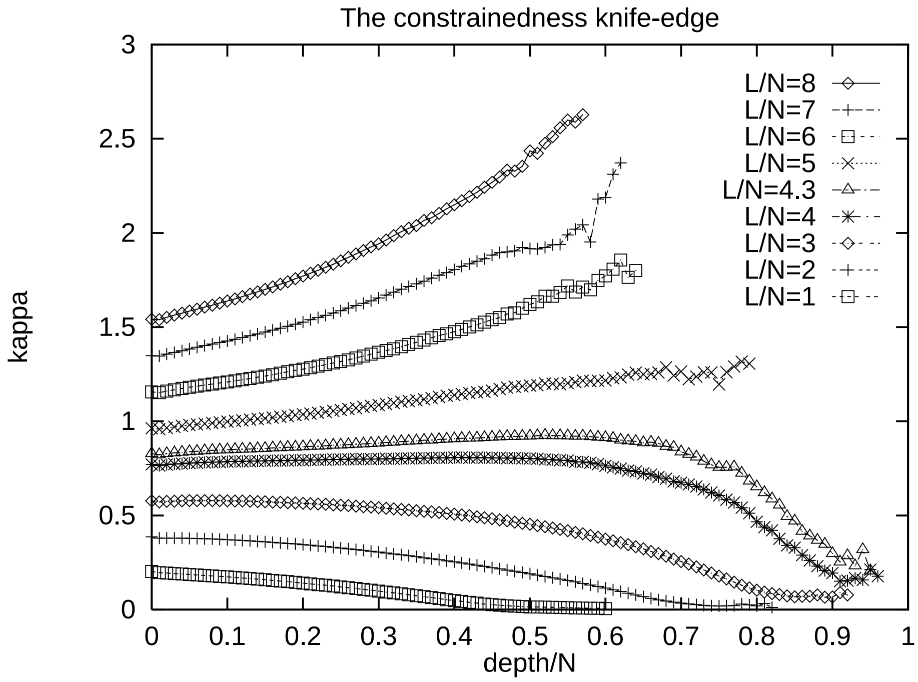

Figure 7. The estimated constrainedness, κ down the heuristic branch (y-axis) against the fractional depth (x-axis) for random 3-SAT problems with 100 variables and varying initial ratio of clauses to variable.

This provides a lower bound on κ that is a good qualitative estimate for how the constrainedness actually varies during search. Let κ_i be the value of κ at depth i. Then,

$$\kappa_0 \quad = \quad 1 - \frac{\log_2(\langle Sol \rangle)}{N}$$

Hence,

$$\log_2(\langle Sol \rangle) \quad = \quad N(1 - \kappa_0)$$

Thus,

$$\begin{aligned} \kappa_i &\geq 1 - \frac{\log_2(\langle Sol \rangle)}{N - i} \\ &= 1 - \frac{N(1 - \kappa_0)}{N - i} \\ &= \frac{N\kappa_0 - i}{N - i} \end{aligned}$$

We can improve this bound slightly by noting that κ is bounded below by zero. Hence,

$$\kappa_i \quad \geq \quad \max(0, \frac{N\kappa_0 - i}{N - i})$$

In Figure 8, we plot this bound on κ for random 3-SAT problems with 100 variables and varying initial ratio of clauses to variable, L/N. We see that in almost every case the behaviour of κ during search observed in Figure 7 is very similar to that predicted by the bound. The exception is at $L/N = 5$, corresponding to $\kappa \approx 0.96$. As $\kappa < 1$, the bound predicts that κ converges to 0 as a solution is found. In reality, however, κ diverges to ∞ as most problems are insoluble.

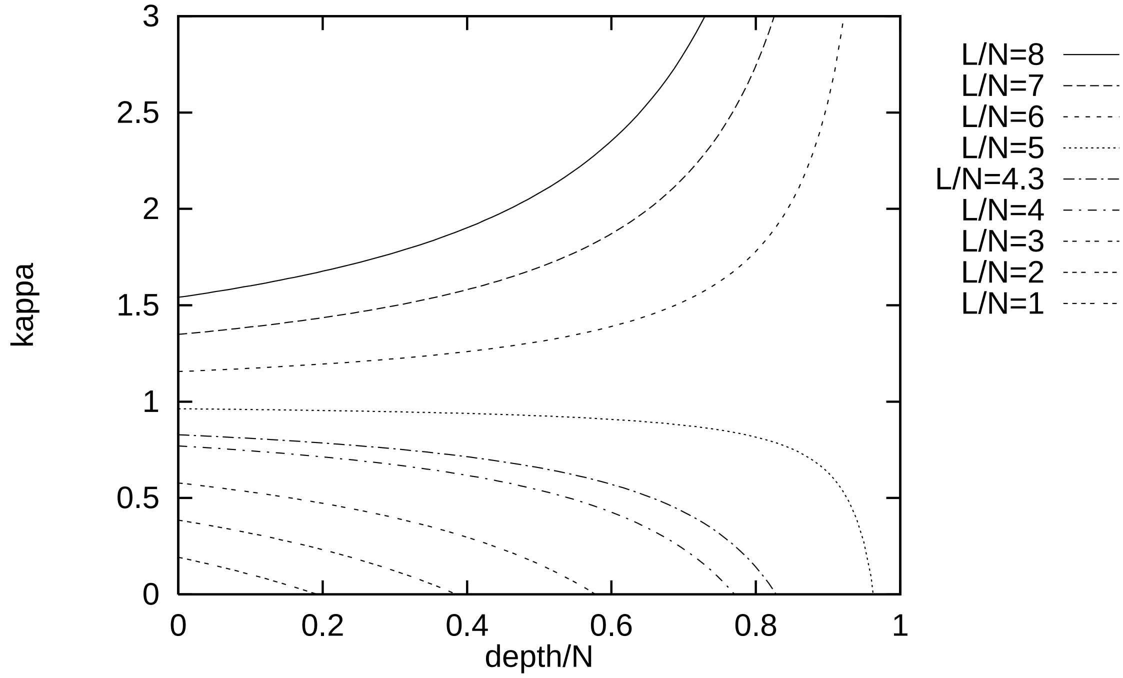

Figure 8. Lower bound on the constrainedness, κ (y-axis) against the fractional depth (x-axis) for random 3-SAT problems with 100 variables and varying initial ratio of clauses to variable.

This lower bound suggests a very simple scaling result. If we let f_i be the fractional depth in the search tree (that is, i/N) then,

$$\kappa_i \quad \geq \quad \max(0, \frac{\kappa_0 - f_i}{1 - f_i})$$

That is, the bound on κ_i is simply a function of κ_0, the initial value and f_i, the fractional depth. It does not depend on the size of the problem. We therefore measured how κ_i scaled with problem size. We found that κ_i remained essentially unchanged if we fixed κ_0 and f_i as problems increased from 100 to 500 variables. This is despite the fact that the size of the state spaces increases by over a factor of 10^{120}.

10 Minimizing constrainedness

The existence of the constrainedness knife-edge may help us design more effective search procedures. For instance, it may be useful to design heuristics that get us off the knife-edge as quickly as possible. Many heuristics attempt this by branching on the most constrained variable, resulting in the least constrained subproblem. Hence, we propose the heuristic of minimizing κ. To test this idea, we performed experiments on randomly generated binary CSP's from the class $\langle n, m, p_1, p_2 \rangle$ described earlier. We encoded minimizing κ as a dynamic variable ordering heuristic within the algorithm FC-CBJ (i.e. forward checking with conflict-directed backjumping)[46]. After instantiating a variable, domain filtering is performed. This may result in a reduction in the size of the domains of future (i.e. uninstantiated) variables and consequently alter the tightness of future constraints (i.e. constraints acting between pairs of future variables). The future sub-problem may then be non-uniform in domain sizes and constraint tightnesses. To measure κ for this reduced problem, we assume it as a representative of the ensemble of problems with the same number of variables, the same domain sizes, and the same number of constraints each of the same tightness as the reduced problem. This is a heuristic assumption which seems to be justified by our results. When considering a variable v_i as the new current variable we remove it and all constraints involving it from the future sub-problem. We then calculate κ for the future sub-problem using equation (3) and take this as the cost of selecting variable v_i. This is done for all future variables and the variable with minimum cost is selected as the current variable.

We compared the minimize-κ heuristic with an encoding of the fail first (FF) principle [26] i.e. selecting the variable with smallest domain. Figure 9 shows the results of experiments performed on $\langle 20, 10, p_1, p_2 \rangle$ problems (i.e. 20 variables, uniform domain size of 10). Constraint density p_1 was varied from 0.2 up to 1.0, for each value of p_1, constraint tightness p_2 was varied to traverse the phase transition in solubility. At each value of p_1 and p_2, we generated 1,000 problems.[1]

The contours shown are for the mean search effort, measured by the number of consistency checks. The minimize-κ heuristic outperforms the FF heuristic, especially around the phase transition. Although not shown, the same holds for median performance. When search effort is measured as the number of trial instantiations of variables, minimize-κ again shows superior mean and median

[1]While technically these instances were subject to the flaw identified by [2], instances generated with these parameters are never flawed in practice [38].

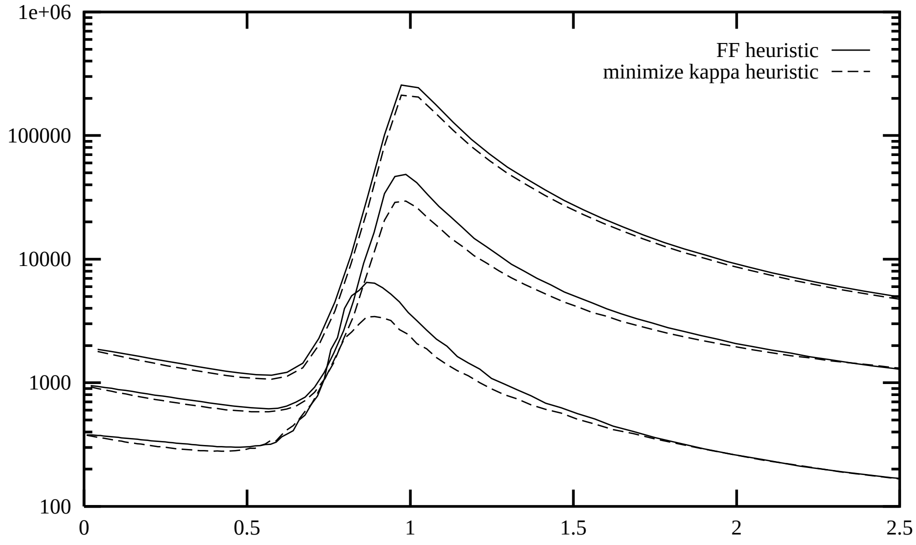

Figure 9. Fail First (FF) and minimize-κ heuristics on $\langle 20, 10, p_1, p_2 \rangle$ problems using FC-CBJ. Mean consistency checks on y-axis, and the constrainedness of problems, κ on x-axis. Contours for $p_1 = 1.0$ (top), $p_1 = 0.5$ (middle), $p_1 = 0.2$ (bottom).

performance. [18] reports more extensive experiments on the minimize-κ heuristic with similar results. At the peak in search costs, paired-sample t-tests gave values of $t = 12.3$ at $p_1 = 0.2$, $t = 24.4$ at $p_1 = 0.5$, and $t = 46.3$ at $p_1 = 1.0$, all in favour of minimize-κ. To check the validity of these values we performed an approximate randomization of the test [9] with a sample of 1000 in each case, which never gave a value above $t = 3.5$. This provides strong statistical evidence that the minimize-κ heuristic is better than the FF heuristic in these problem classes. [54] give results on the same problem classes seen in Figure 9, on a range of algorithm/heuristic combinations. For high values of p_1 they report that FC-CBJ with the FF heuristic was the best combination studied for problems near the phase transition. The fact that the minimize-κ heuristic can do better is strong evidence that it is a good heuristic.

The complexity of (3) leads to significant overheads in computation. As a consequence, the minimize-κ heuristic does not reduce run-times on this problem class. However, it is not difficult to find problems in which it gives dramatic run-time savings. Fail-first ignores the tightness of constraints. This is not a great handicap in Figure 9 since problems start out with a uniform constraint tightness. If we start with problems in which the constraint tightnesses varies significantly, then the minimize-κ heuristic can offer up to four orders of magnitude improvement in performance, and offer run-time savings. To construct problems with varying constraint tightness, we generated problems in which exactly 20% of the constraints have tightness $p_2 = 0.8$ (i.e. tight constraints) and the remainder tightness $p_2 = 0.2$ (i.e. loose constraints). We set $n = 30$ and $m = 10$, and observed a phase transition as we varied the constraint graph density, p_1 from $\frac{1}{87}$ to 1 in steps of $\frac{1}{87}$. Results are plotted in Figure 10. The 50% solubility point is

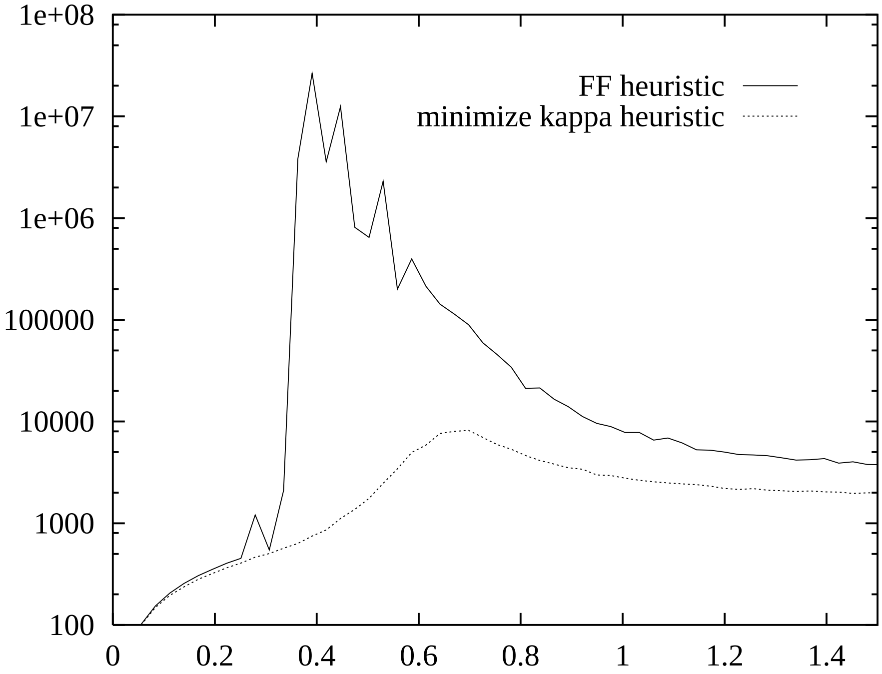

Figure 10. Fail First (FF) and minimize-κ heuristics on binary constraint satisfaction problems using FC-CBJ. Mean consistency checks on y-axis, and the constrainedness of problems, κ on x-axis. Problems have $n = 30$, $m = 10$, varying p_1, and $p_2 = 0.2$ for 80% of the constraints, and $p_2 = 0.8$ for the remainder.

at $\kappa \approx 0.64$ when $p_1 = \frac{23}{87}$. The mean and worst case performance show the existence of exceptionally hard problems for FF. The worst case for FF was 2.7×10^{10} consistency checks at $\kappa \approx 0.39$, in a region where 100% of problems were soluble. This was 8 orders of magnitude worse than the median of 659 checks at this point, and took 87 hours on a DEC Alpha $200^{4/166}$.

11 Maximizing constrainedness

For soluble problems, the existence of the constrainedness knife-edge suggests that we try to get off the knife-edge as quickly as possible by branching into the subproblem that is as under-constrained as possible. That is, we branch into the subproblem that minimizes κ. For insoluble problems, it might be better to branch into the sub-problem that is as over-constrained as possible. That is, we branch into the subproblem that maximizes κ. To test this idea, we ran experiments on some propositional satisfiability and constraint satisfaction problems using a maximize-κ heuristic.

For the satisfiability experiments, we implemented minimize-κ and maximize-κ branching heuristics for the Davis-Putnam procedure. In Table 1, we show how these heuristics compare to MOM's heuristic on hard random 3-SAT problems from the middle of the phase transition. The results support the thesis that, for soluble problems, it pays to minimize κ and for insoluble problems, it pays to maximize κ. However, we should be careful to draw too many conclusions from these results. In the Davis-Putnam procedure, we are exploring a simple binary tree. In other domains, we may be searching trees in which the branching rate varies. For example, when solving constraint satisfaction problems with a forward checking algorithm like FC-CBJ, the branching rate depends on the domain size of the variable being instantiated and this varies during search. Heuristics that maximize κ will tend to branch on variables with large domains, resulting in bushy search trees which could be expensive to explore.

	MOM	$\min-\kappa$	$\max-\kappa$
satisfiable problems	164	104	1487
unsatisfiable problems	3331	7419	2575

Table 1: Median nodes searched by the Davis-Putnam procedure for random 3-SAT problems at $n = 50$ and $l/n = 4.3$ using different heuristics.

For the constraint satisfaction experiments, we were unable to test the $\langle 20, 10, p_1, p_2 \rangle$ problems used in the previous section since they were too hard to solve in general using the maximize-κ heuristic. For example, FC-CBJ failed to solve a $\langle 20, 10, 0.5, 0.37 \rangle$ problem (in the middle of the phase transition) in over 2 million consistency checks using a maximize-κ heuristic. By comparison, FC-CBJ with the FF heuristic took 61,200 consistency checks to solve this problem, and just 39,177 checks with the minimize-κ heuristic. We therefore ran our experiments on problems with fewer variables. As in [17] and [20], we used $\langle 10, 10, 1, p_2 \rangle$ problems varying p_2 from 0.2 to 0.5 with a sample size of 50. In the middle of the phase transition, the maximize-κ heuristic is an order of magnitude worse than the FF or minimize-κ heuristic. As in Section 10, the minimize-κ heuristic is competitive with the FF heuristic, on both soluble and insoluble problems.

On very under-constrained and soluble problems, close inspection of the data shows that the maximize-κ heuristic outperforms the FF and minimize-κ heuristics. On such problems, the maximize-κ heuristic selects a variable with a large domain, and forward checking then performs consistency checking against variables with small domains. By comparison, the FF and minimize-κ heuristics,

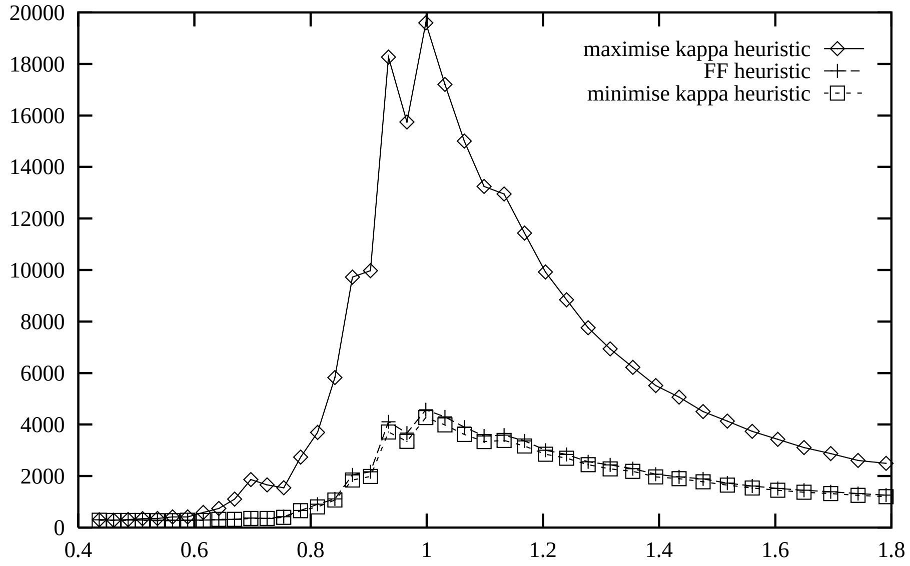

Figure 11. Fail First (FF), minimize-κ and maximize-κ heuristics on $\langle 10, 10, 1, p_2 \rangle$ problems using FC-CBJ. Mean consistency checks on y-axis, and constrainedness of problems, κ on x-axis.

select a variable with a small domain, and forward checking then performs consistency checking against variables with large domains. For very under-constrained problems that can be solved with little or no search, reducing the amount of work performed by forward checking may therefore result in computational savings. By comparison, on more constrained problems, the maximize-κ heuristic performs poorly. As predicted, the heuristic branches on variables with large domains, giving bushy search trees which are expensive to explore. We conjecture that maximize-κ will perform even worse with a chronological backtracking procedure like forward checking. Our results suggest that, whilst minimizing constrainedness can be an effective way to get off the knife-edge for soluble and, in some cases, for insoluble problems, there are domains in which maximizing constrainedness is ineffective even on insoluble problems.

12 Proxies for constrainedness

As mentioned in Section 10, it may be expensive to compute κ. We may therefore use a proxy which is cheaper to compute. For example, in number partitioning, Gent and Walsh have shown that the Karmarkar-Karp heuristic minimizes an estimate for κ that is based on the assumption that the numbers left to partition remain uniformly distributed [22]. The Karmarkar-Karp heuristic is cheap to compute as it merely requires us to maintain the numbers left to partition in sorted order.

In constraint satisfaction problems, if we assume that all constraints in a problem have the same tightness, and that each variable is in the same number of constraints, we can ignore the numerator of (3) as it will be the same whichever variable we instantiate. The variable chosen should then be the one that maximizes the denominator of (3), and is equivalent to instantiating the variable with smallest domain. This is the fail-first (FF) heuristic [26] which is again cheap to compute.

An alternative assumption is that all variables have the same domain size. This is valid when all variables start out with identical domain sizes and we use a backward checking algorithm, i.e. an algorithm that does not perform domain filtering of the future variables. The denominator will now be the same whichever variable we instantiate. If we further assume that all constraint tightnesses are the same, the numerator becomes the cardinality of the set of constraints acting between future variables and between future and past variables. We minimize the numerator of (3) by choosing a variable that has most constraints with past variables. This corresponds to the maximum cardinality heuristic described in [12].

We may take advantage of both numerator and denominator of (3). One way to do this is to choose the variable with smallest domain size (maximizing the denominator) and break ties by choosing the tied variable in most constraints (minimizing the numerator, assuming uniform constraint tightness). This is the Brelaz heuristic [5].

Not all proposed heuristics are as successful as those mentioned above. Indeed in some cases intuition seems to have led designers in exactly the wrong direction. For example, in their backtracking algorithm for the Hamiltonian Cycle problem, Cheeseman, Kanefsky and Taylor selected the node to go to with the *highest* connectivity at each choice point [7]. However, to minimise the numerator of (4), we should maximize the number of edges that remain in the graph after each choice point. This suggests picking the node with the *lowest* connectivity, not the highest, and this is the approach taken by Martello's algorithm [39].

Four state of the art heuristics can thus be seen as proxies for minimizing κ. Domain knowledge may still be needed to convert the idea of minimizing κ into a heuristic with low overheads. However, by considering how to minimize κ, we can remove much of the intuition involved in developing heuristics for a new domain. While intuition is valuable, it can often be misleading or even wrong. Furthermore, intuition about new domains can be hard to achieve. We therefore see this reduction in the role of intuition in heuristic design as a significant contribution.

13 Related work

We are not aware of any other work which both introduces a general measure of constrainedness and uses it to design heuristics. However, a number of other workers have studied one or other aspect, either measures of constrainedness or the design of heuristics based on theoretical principles.

Williams and Hogg present a closely related model for locating phase transition behaviour and predicting search cost in graph colouring and constraint satisfaction problems [57]. Their "deep structure" model focuses on the number of minimized nogoods. The approach presented here is more general as it can be applied to a wider range of problem domains. Indeed, our analysis has made very few assumptions about the computational complexity of the search problems being solved. We have merely assumed that we are looking for a solution within some finite state space. Our framework is therefore applicable to the wide range of NP-complete problems. However, we can also apply these ideas to problems in other complexity classes. For example, we have modelled phase transition behaviour and suggested new heuristics for polynomial problems like establishing arc consistency in constraint satisfaction problems [19]. Whatever the complexity class, our definition of constrainedness may be able to identify phase transition behaviour and suggest heuristics that help us solve search problems.

Musick and Russell model search using an abstracted Markov process that considers just the distance from a solution [44]. They identify regions where problems are easy and outside which it is very hard to find a solution. It would be fruitful to explore the connections between constrainedness, and the transition probabilities of such Markov processes.

Smith proposes a heuristic for binary constraint satisfaction problems that simply maximizes the expected number of solutions, $\langle Sol \rangle$ [50]. Given a choice of two subproblems with equal $\langle Sol \rangle$, the heuristic of minimizing κ will branch into the smaller problem in the expectation that this is less constrained. Experiments so far have failed to show which heuristic, if either, is better [18].

Hooker and Vinay investigate the Jeroslow-Wang heuristic for satisfiability [29]. They propose the "satisfaction hypothesis", that it is best to branch into subproblems that are more likely to be satisfiable, but reject this in favour of the "simplification hypothesis", that it is best to branch into simpler subproblems with fewer and shorter clauses after unit propagation. Minimizing κ is related but not identical to both these hypotheses: in general it will seek out simple problems that are likely to be soluble.

Nudel has proposed some theoretically motivated heuristics for binary CSP's [45]. Two classes of heuristic are presented, global and local. Global heuristics fix the instantiation order at the start of search, whereas local heuristics take account of information made available during search, such as actual domain sizes and constraint tightness. Nudel's local heuristics are thus dynamic variable ordering heuristics. It is interesting to contrast our approach with Nudel's as both give theory-based variable ordering heuristics. Nudel presents measure that estimate the size of the remaining search tree, and then constructs heuristics which seek to minimize these estimates. We have not related our measures directly to the search tree. Instead we have sought to move into areas of the search tree likely to be unconstrained and therefore have solutions. When one makes certain simplifications, both approaches can result in the same heuristic such as the FF heuristic. However, the detailed relationship between the approaches has not yet been fully analysed.

Heuristics are, by their nature, inexact. It can therefore be difficult to decide how rigorously to apply a given theory about heuristic construction. Smith and Grant investigated this problem for the 'fail first' principle in binary CSP's [52], a principle often used to justify the minimum domain size (FF) heuristic. They found that a heuristic which did more work to maximize the probability of an early failure did significantly more search than a simpler heuristic that did less work, even in measures such as nodes searched which are independent of the cost of calculating the heuristics. Heuristics based upon theoretical principles, including that of minimizing constrainedness, have still to address problems such as this.

The constrainedness of a problem depends on the ensemble from which it is drawn. We may not know the ensemble from which a problem is drawn, so naive measurements of κ may mislead us. Hogg uses the "approximate entropy" to distinguish between problems drawn from a clustered ensemble and those from a random ensemble [27]. Approximate entropy may therefore be useful in estimating constrainedness. However, as the approximate entropy depends on the representation used, the role of problem representation is also critical. Further work in this area, perhaps along the lines of [4], is vital if this research is to be of practical value in understanding and solving real problems.

14 Conclusions

Branching heuristics often try to make the most "constrained" choice, whilst hard problems tend to be "critically constrained". We have developed a general definition of the constrainedness of search problems that unifies these two notions of constrainedness. We have shown that our definition of constrainedness generalizes a number of parameters used to study phase transition behaviour in a wide variety of different problem domains. It allows the rapid identification of phase transitions in new problem domains, and the comparison of phase transitions in previously incomparable classes. Our definition also provides insight into why problems at such phase transitions tend to be hard to solve. These problems are on a constrainedness "knife-edge", and we must search deep into the problem before they look more or less soluble. Heuristics that try to get off this knife-edge as quickly as possible by, for example, minimizing the constrainedness are often therefore very effective. Many existing heuristics can be seen as minimizing constrainedness or proxies for it. Our definition of constrainedness therefore offers a unified understanding of many widely disparate heuristics, and provides a principled method for constructing heuristics for new domains.

Acknowledgements

The authors are members of the APES research group, http://www.cs.strath.ac.uk/~apes. We thank our colleagues in the group at the Universities of Strathclyde and Leeds, most especially Ewan MacIntyre. We also thank the members of the Mathematical Reasoning Group at Edinburgh University. The authors are supported by EPSRC awards GR/L/24014 and GR/K/65706. We thank Bob Craig and Mark Stickel for code, and Vince Darley for correcting an error in the derivation of $\langle Sol \rangle$ for the ATSP.

References

[1] M. Abramowitz and I.A. Stegun, editors. *Handbook of mathematical functions*. Dover.

[2] D. Achlioptas, L.M. Kirousis, E. Kranakis, D. Krizanc, M.S.O. Molloy, and Y.C. Stamatiou. Random constraint satisfaction: A more accurate picture. In G. Smolka, editor, *Proceedings of Third International Conference on Principles and Practice of Constraint Programming (CP97)*, pages 107–120. Springer, 1997.

[3] Michael N. Barber. Finite-size scaling. In *Phase Transitions and Critical Phenomena, Volume 8*, pages 145–266. Academic Press, 1983.

[4] J.E. Borrett and E.P.K. Tsang. On the selection of constraint satisfaction formulations. Technical report CSM-254, Department of Computer Science, University of Essex, October 1995.

[5] D. Brelaz. New methods to color the vertices of a graph. *Communications of ACM*, 22:251–256, 1979.

[6] G. Carpaneto and P. Toth. New branching and bounding criteria for the asymmetric travelling salesman problem. *Management Sci.*, 26:736–743, 1980.

[7] P. Cheeseman, B. Kanefsky, and W.M. Taylor. Where the really hard problems are. In *Proceedings of the 12th IJCAI*, pages 331–337. International Joint Conference on Artificial Intelligence, 1991.

[8] V. Chvatal and B. Reed. Mick gets some (the odds are on his side). In *Proceedings of the 33rd Annual Symposium on Foundations of Computer Science*, pages 620–627. IEEE, 1992.

[9] P.R. Cohen. *Empirical methods for Artificial Intelligence*. MIT Press, 1995.

[10] J.M. Crawford and L.D. Auton. Experimental Results on the Cross-Over Point in Satisfiability Problmes. In *Proceedings of AAAI 1993 Spring Symposium on AI and NP-Hard Problems*, 1993.

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

[12] R. Dechter and I. Meiri. Experimental evaluation of preprocessing algorithms for constraint satisfaction problems. *Artificial Intelligence*, 68:211–242, 1994.

[13] P.E. Dunne and M. Zito. An improved upper bound on the non-3-colourability threshold. *Information Processing Letters*, 65:17–23, 1998.

[14] J. Frank, I.P. Gent, and T. Walsh. Asymptotic and finite size parameters for phase transitions: Hamiltonian circuit as a case study. *Information Processing Letters*, 66(5):241–245, 1998.

[15] A. Frieze and S. Suen. Analysis of two simple heuristics on a random instance of k-SAT. *Journal of Algorithms*, 20:312–355, 1996.

[16] D. Frost, I. Rish, and L. Vila. Summarizing CSP hardness with continuous probability distributions. In *Proceedings of the 14th National Conference on AI*, pages 327–333. American Association for Artificial Intelligence, 1997.

[17] J. Gaschnig. Performance measurement and analysis of certain search algorithms. Technical report CMU-CS-79-124, Carnegie-Mellon University, 1979. PhD thesis.

[18] I.P. Gent, E. MacIntyre, P. Prosser, B.M.Smith, and T. Walsh. An empirical study of dynamic variable ordering heuristics for the constraint satisfaction problem. In *Proceedings of CP-96*, pages 179–193. Springer Verlag, 1996.

[19] I.P. Gent, E. MacIntyre, P. Prosser, P. Shaw, and T. Walsh. The constrainedness of arc consistency. In *3rd International Conference on Principles and Practices of Constraint Programming (CP-97)*, pages 327–340. Springer, 1997.

[20] I.P. Gent, E. MacIntyre, P. Prosser, and T. Walsh. Scaling effects in the CSP phase transition. In *Principles and Practice of Constraint Programming*, pages 70–87. Springer, 1995.

[21] I.P. Gent and T. Walsh. The SAT phase transition. In A G Cohn, editor, *Proceedings of 11th ECAI*, pages 105–109. John Wiley & Sons, 1994.

[22] I.P. Gent and T. Walsh. Phase transitions and annealed theories: Number partitioning as a case study. In *Proceedings of ECAI-96*, pages 170–174, 1996.

[23] I.P. Gent and T. Walsh. The TSP phase transition. *Artificial Intelligence*, 88:349–358, 1996.

[24] I.P. Gent and T. Walsh. Analysis of heuristics for number partitioning. *Computational Intelligence*, 14(3):430–451, 1998.

[25] A. Goerdt. A theshold for unsatisfiability. In I. Havel and V. Koubek, editors, *Mathematical Foundations of Computer Science*, Lecture Notes in Computer Science, pages 264–274. Springer Verlag, 1992.

[26] R.M. Haralick and G.L. Elliott. Increasing tree search efficiency for constraint satisfaction problems. *Artificial Intelligence*, 14:263–313, 1980.

[27] T. Hogg. Which search problems are random? In *Proceedings of 15th National Conference on Artificial Intelligence*, pages 438–443. AAAI Press/The MIT Press, 1998.

[28] T. Hogg and C.P. Williams. The hardest constraint problems: A double phase transition. *Artificial Intelligence*, 69:359–377, 1994.

[29] J. N. Hooker. Testing heuristics: We have it all wrong. *Journal of Heuristics*, 1:33–42, 1995.

[30] J. N. Hooker and C. Fedjki. Branch-and-cut solution of inference problems in propositional logic. *Annals of Mathematics and Artificial Intelligence*, 1:123–139, 1990.

[31] Holger Hoos and Thomas Stützle. Characterizing the Run-time Behavior of Stochastic Local Search. Technical Report AIDA–98–01, FG Intellektik, TU Darmstadt, January 1998.

[32] A. Kamath, R. Motwani, K. Palem, and P. Spirakis. Tail bounds for occupancy and the satisfiability threshold conjecture. *Randomized Structure and Algorithms*, 7:59–80, 1995.

[33] S. Kirkpatrick and B. Selman. Criticial behaviour in the satisfiability of random boolean expressions. *Science*, 264:1297–1301, 1994.

[34] L.M. Kirousis, E. Kranakis, and D. Krizanc. Approximating the unsatisfiability threshold of random formulas. In *Proceedings of the 4th Annual European Symposium on Algorithms (ESA'96)*, pages 27–38, 1996.

[35] R. Korf. From approximate to optimal solutions: A case study of number partitioning. In *Proceedings of the 14th IJCAI*. International Joint Conference on Artificial Intelligence, 1995.

[36] A.D. Korshunov. The main properties of random graphs with a large number of vertices and edges. *Russian Math. Surveys*, pages 121–198, 1985.

[37] A.D. Koršunov. Solution of a problem of Erdös and Rényi on Hamiltonian cycles in nonoriented graphs. *Soviet Math. Dokl.*, 17(3):760–764, 1976.

[38] E. MacIntyre, P. Prosser, B.M. Smith, and T. Walsh. Random constraint satisfaction: Theory meets practice. In *4th International Conference on Principles and Practices of Constraint Programming (CP-98)*, pages 325–339. Springer, 1998.

[39] S. Martello. An enumerative algorithm for finding Hamiltonian circuits in a directed graph. *ACM Transactions on Mathematical Software*, 9:131–138, 1983.

[40] S. Mertens. Phase transition in the number partitioning problem. http://xxx.lanl.gov/abs/cond-mat/9807077, 1998.

[41] S. Minton. Integrating heuristics for constraint satisfaction problems: a case study. In *Proceedings of the 11th National Conference on AI*, pages 120–126. American Association for Artificial Intelligence, 1993.

[42] S. Minton. Is there any need for domain-dependent control information? a reply. In *Proceedings of the 13th National Conference on AI*, pages 855–862. American Association for Artificial Intelligence, 1996.

[43] D. Mitchell, B. Selman, and H. Levesque. Hard and Easy Distributions of SAT Problems. In *Proceedings of the 10th National Conference on AI*, pages 459–465. American Association for Artificial Intelligence, 1992.

[44] R. Musick and S. Russell. How long will it take? In *Proceedings of the 10th National Conference on AI*, pages 466–471. American Association for Artificial Intelligence, 1992.

[45] B. Nudel. Consistent-labeling problems and their algorithms: Expected-complexities and theory-based heuristics. *Artificial Intelligence*, 21:135–178, 1983.

[46] P. Prosser. Hybrid algorithms for the constraint satisfaction problem. *Computational Intelligence*, 9:268–299, 1993.

[47] P. Prosser. An empirical study of phase transitions in binary constraint satisfaction problems. *Artificial Intelligence*, 81:127–154, 1996.

[48] I. Rish and D. Frost. Statistical analysis of backtracking on inconsistent CSPs. In G. Smolka, editor, *Proceedings of Third International Conference on Principles and Practice of Constraint Programming (CP97)*, pages 150–162. Springer, 1997.

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

[50] B.M. Smith. Re: variable choice. *CSP-List Digest*, 75, October 1995. Mailing list, archived at ftp://ftp.cs.city.ac.uk/pub/constraints/archive/csp-list/95.10.75.gz.

[51] B.M. Smith and M.E. Dyer. Locating the phase transition in binary constraint satisfaction problems. *Artificial Intelligence*, 81:155–181, 1996.

[52] B.M. Smith and S.A. Grant. Trying harder to fail first. In *Proceedings of the 13th ECAI*, pages 249–253. European Conference on Artificial Intelligence, 1998.

[53] P. Svenson and M.G. Nordahl. Relaxation in graph coloring and satisfiability problems, 1998. Paper in the xxx.lanl.gov e-Print archive. Available from http://xxx.lanl.gov/ps/cond-mat/9810144.

[54] E.P.K. Tsang, J.E. Borrett, and A.C.M. Kwan. An attempt to map the performance of a range of algorithm and heuristic combinations. In *Hybrid Problems, Hybrid Solutions*, pages 203–216. IOS Press, 1995. Proceedings of AISB-95.

[55] B. Vandegriend and J. Culberson. The Gn,m phase transition is not hard for the Hamiltonian Cycle problem. *Journal of Artificial Intelligence Research*, 9:219–245, 1998.

[56] T. Walsh. The constrainedness knife-edge. In *Proceedings of the 15th National Conference on AI*. American Association for Artificial Intelligence, 1998.

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

[58] W. Zhang and R.E. Korf. An average-case analysis of branch-and-bound with applications: Summary of results. In *Proceedings of 10th National Conference on Artificial Intelligence*, pages 769–775. AAAI Press/The MIT Press, 1992.