Morphing: Combining Structure and Randomness

Ian P. Gent
Department of Computer Science
University of Strathclyde
Glasgow G1 1XL
Scotland
ipg@cs.strath.ac.uk

Holger H. Hoos
Department of Computer Science
University of British Columbia
Vancouver V6T 1Z4
Canada
hoos@cs.ubc.ca

Patrick Prosser and **Toby Walsh**
Department of Computer Science
University of Strathclyde
Glasgow G1 1XL
Scotland
{pat,tw}@cs.strath.ac.uk

Abstract

We introduce a mechanism called "morphing" for introducing structure or randomness into a wide variety of problems. We illustrate the usefulness of morphing by performing several different experimental studies. These studies identify the impact of a "small-world" topology on the cost of coloring graphs, of asymmetry on the cost of finding the optimal TSP tour, and of the dimensionality of space on the cost of finding the optimal TSP tour. We predict that morphing will find many other uses.

Introduction

Structures that occur in the real world problems tend to be neither completely regular nor completely random. Consider delivering parcels round Manhattan. One-way streets, traffic, road-works and a host of other factors make the problem more complex than navigating on a simple grid. To model the impact of one-way restrictions on such a delivery problem, we took a two-way grid and slowly introduced one-way streets into it at random (using a "morphing" process described in the next section). We did not introduce one-way streets completely at random but so that they eventually alternate in direction.

The result of this experiment was a little surprising. As expected, adding one-way streets increased the distance needed to deliver parcels to a random set of locations. However, the median cost to find the optimal route dropped. It appears to be easier to navigate in cities with one-way streets than cities with no one-way restrictions. The reason may be that one-way streets often leave few choices as to the optimal route. Whilst median cost tended to drop, higher percentiles in search cost were often larger. The reason may be that we can occasionally make a very bad choice and have to backtrack a long way.

To model structures like a Manhattan grid with a mixture of one-way and two-way streets, this paper introduces a general purpose mechanism called "morphing". We show that morphing has many other applications. It provides a simple dial with which we can introduce structure or randomness into problems. Morphing operations can be defined on almost any type of structure. For example, in our Manhattan delivery problem, we morphed directed graphs. However, we can also morph distance matrices in traveling salesperson (TSP) problems, undirected graphs in coloring problems, clauses in satisfiability (SAT) problems, and relations in constraint satisfaction problems (CSPs). Morphing provides us with a powerful tool to study the impact of structure and randomness on these and many other types of problem.

Morphing

Given two structures, S_1 and S_2, we define either a type A, type B or type C morph from S_1 to S_2. In a type A morph, we take substructures from S_1 with probability $(1-p)$ and from S_2 with probability p. In a type B morph, we take a fraction $1-p$ of the substructures from S_1, and a fraction p of the substructures from S_2. In a type C morph, we assume the existence of suitable operations for addition and scalar multiplication, and compute $(1-p).S_1 + p.S_2$. We will often define the morphing operation so that if both structures have a substructure in common then this is also found in any morph. Here are some examples of the three types of morphs on a variety of different structures.

matrix morph (type A): the substructures are the entries; to morph between two $n \times m$ matrices we consider each entry in turn, and include the entry from the first matrix with probability $1-p$, otherwise we include that from the second matrix.

graph morph (type B): the substructures are the edges and gaps (absence of edges) between nodes; to morph between two n node graphs, G_1 and G_2, we take all edges in common, and a fraction $1-p$ of the remaining edges from the first graph, and p from the second.[1]

vector morph (type C): to morph between two vectors, $\vec{v}_1 = (x_1, y_1, \ldots)$ and $\vec{v}_2 = (x_2, y_2, \ldots)$, we construct the vector $(1-p).\vec{v}_1 + p.\vec{v}_2 = ((1-p).x_1 + p.x_2, (1-p).y_1 + p.y_2, \ldots)$.

satisfiability morph (type A): the substructures are clauses; to morph between two SAT problems, we include clauses from the first instance with probability $1-p$, and clauses from the second with probability p.

[1] We assume that the nodes in the two graphs share the same names.

set morph (type B): the substructures are the elements of the set; to morph between two sets, S_1 and S_2, we take all elements in common, $S_1 \cap S_2$, and a fraction $1-p$ of the remaining elements from S_1, and p from S_2.

function morph (type C): to morph between the function $f_1(x)$ and the function $f_2(x)$, we construct the function $(1-p).f_1(x) + p.f_2(x)$.

Two types of morph have already been studied in some detail.

random graphs: type A and B morphs between complete graphs and empty graphs give, respectively, the G_{np} and G_{nm} problem classes[2] (Bollobás 1985);

2+p-SAT problems: type B morphs between 2-SAT and 3-SAT problems give the $2+p$-SAT problem class (Monasson *et al.* 1996), used to study changes in phase transition behavior as we move from P to NP.

In the rest of this paper, we look at four new applications: distance matrix morphs (to identify the impact of asymmetry on TSP problems); coordinate vector morphs (to study the impact of increasing the dimensionality of a TSP problem); ring lattice morphs (to model a recently identified topological structure called "small-worldiness"); and quasigroup morphs (to study in more detail the relationship between such small-worldiness and search cost).

Morphing and Small Worlds

Morphing provides us with a powerful tool to study topological structures like "small worldiness" that occur in many real-world graphs (Watts & Strogatz 1998). A small world graph has nodes that are highly clustered yet path lengths between them that are small. By comparison, random graphs with a similar number of nodes and edges have short path lengths but little clustering, whilst regular graphs like lattices tend to have high clustering but large path lengths. (Walsh 1998) shows that graphs associated with many search problems (e.g. exam timetabling, register allocation, quasigroup problems, ...) often have this topology. One reason for the occurrence of a small world topology is that it only takes a few short cuts between neighborhood cliques to turn a large world (in which the average path length between nodes is large) to a small world (in which the average path length is small). Walsh argues that a small world topology can make search problems very difficult since local decisions quickly propagate globally.

Testing for a Small World

To formalize the notion of a small world, Watts and Strogatz define the clustering coefficient and the characteristic path length. The path length is the number of edges in the shortest path between two nodes. The characteristic path length, L is the path length averaged over all pairs of nodes. The clustering coefficient is a measure of the cliqueness of the local neighborhoods. For a node with k neighbors, then at most $k(k-1)/2$ edges can exist between them (this occurs if they form a k-clique). The clustering of a node is the fraction of these allowable edges that occur. The clustering coefficient, C is the average clustering over all the nodes in the graph. Watts and Strogatz define a small world graph as one in which $L \gtrsim L_{rand}$ and $C \gg C_{rand}$ where L_{rand} and C_{rand} are the characteristic path length and clustering coefficient of a random graph with the same number of nodes n and edges e. Walsh refines this definition, by proposing the proximity ratio μ, C/L normalized by C_{rand}/L_{rand}. Small world graphs are those in which $\mu \gg 1$.

Morphing *v.* Rewiring

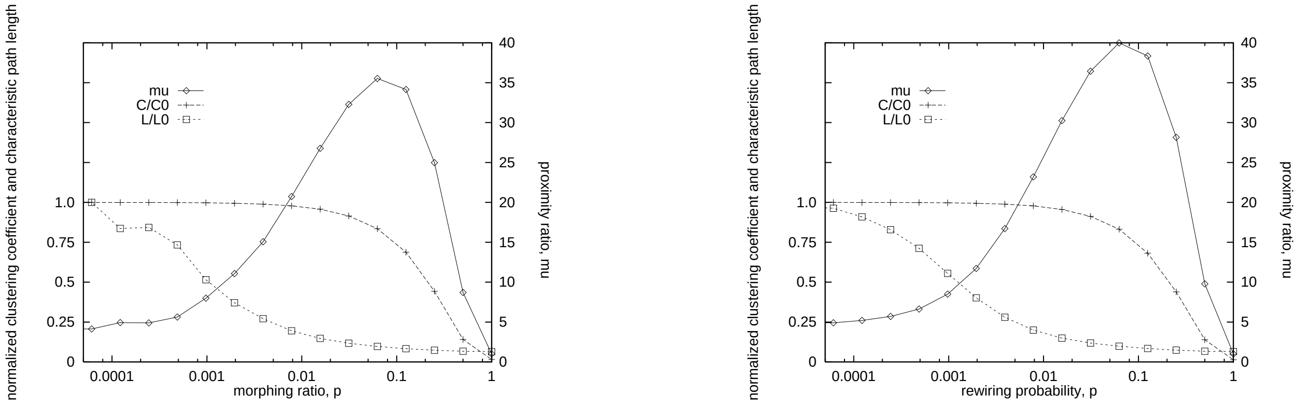
To model small world graphs, (Watts & Strogatz 1998) and (Walsh 1998) use a rather complex method for randomly rewiring a ring lattice that circles the lattice a number of times, rewiring edges that are an increasing distance apart. This method gives graphs with a mixture of both structure and randomness. Morphing provides a much simpler and more general mechanism for constructing small world graphs. We simply morph between a clustered graph with large path lengths (e.g. a ring lattice) and a random graph (which has short path lengths but little clustering). The simplicity of the morphing process brings a variety of benefits. For example, the theoretical analysis of morphs is likely to be much easier than that of rewired graphs.

In Figure 1, we compare randomly rewiring a ring lattice with morphing between a ring lattice and a random graph. A ring lattice is a ring of n nodes, each connected to the k nearest neighbours. In this, and subsequent experiments, we use type B morphs. However, we observe very similar results with type A morphs. The characteristic path length, the clustering coefficient and proximity ratio all vary in a very similar manner, although there is a slight decrease in the maximum value of the proximity ratio, μ with morphing. At small p, we also see granularity effects. In type B graph morphs, p is the fraction of edges to include from one of the graphs. We therefore round to the nearest whole number. By comparison, p in random rewiring is a probability and so is not rounded. We also ran an experiment morphing between a ring lattice and a random graph using type A morphs. As this gave very similar results, we omit these graphs.

Local Search Procedures

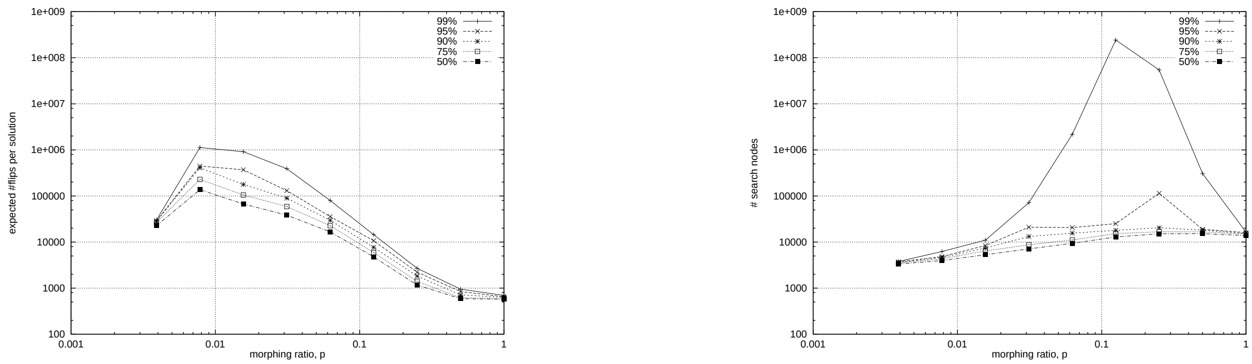
To study the impact of small-worldiness on local search behavior, we encoded the problem of coloring small world graphs into SAT instances and solved these using WalkSAT, one of the most popular and efficient stochastic local search algorithms for SAT. As in the last section, we generated graphs by morphing between a ring lattice and a random graph with the same number of nodes and edges. To ensure that problems were of a manageable size for our algorithms, we used graphs with $n = 100$ and $k = 8$. Because of the regular ring lattice structure, the minimal chromatic number, c of these graphs is 5. However, some graphs with higher chromatic numbers do occur. Initial experiments indicated that the cost to color graphs with $c > 5$ is at least one order of magnitude lower than graphs with $c = 5$. To give a more homogeneous test set, we therefore filtered out the instances with $c \neq 5$ using a complete graph coloring

[2] We are very grateful to Joe Culberson for making this observation to us in a private communication.



(a) morphing graphs (b) random rewiring

Figure 1. Characteristic path length, clustering coefficient (left axis, normalized by the values for a regular lattice) and proximity ratio (right axis) for graphs generated (a) by morphing between a ring lattice and a random graph and (b) by random rewiring of a ring lattice. As in (Watts & Strogatz 1998), we use $n = 1000$ and $k = 10$. We vary $\log_2(p)$ from -15 to 0 in steps of 1, and generate 100 graphs at each value of p. A logarithmic horizontal scale helps to identify the interval in which the characteristic path length drops rapidly, the clustering coefficient remains almost constant, and the proximity ratio, μ peaks. To aid comparison, the same x and y scales are used in both figures.



(a) WalkSAT local search procedure (b) the "satz" procedure

Figure 2. Search cost to color small world graphs; **(a)** percentiles of expected number of flips required by the local search procedure, WalkSAT; **(b)** percentiles of nodes visited by the complete search procedure, satz. The small world graphs are generated by morphing between a ring lattice and a random graph with $n = 100$, $k = 8$, and p varying $\log_2(p)$ from -8 to -2 in steps of 1. The proximity ratio, μ peaks around $\log_2(p) \approx -4$ similar to Figure 1. 100 5-colourable graphs were used at each value of p.

algorithm provided by Joe Culberson. We then encoded the remaining problems into SAT, with each propositional variable representing a particular color being assigned to a node. We solved each instance with WalkSAT, measuring the expected number of flips to find a solution over 100 runs of the algorithms. We set the cutoff parameter large enough that a solution was found in each run, and optimised the noise parameter for each value of p to give approximately minimal local search cost. Interestingly, the optimal noise setting appears to be positively correlated with p, i.e., the more random structure in graphs, the less greediness needed in WalkSAT to obtain optimal performance.

Figure 2 (a) shows the higher percentiles in the local search cost for varying p. All the percentiles peak at $\log_2(p) = -7$ (including the lower percentiles not shown in the graph). Near to this point, the variability in search cost across the test set is also maximal and the variation coefficient (standard deviation/mean) = 3.05. Whilst the distribution of the expected search costs across the test set around this region, is heavy-tailed, search costs on individual instances appear to best fit an exponential distribution. This suggests that the restart mechanism does not improve performance on these problems. Note also that when decreasing $\log_2(p)$ from -7 to -8 all percentiles drop sharply, for the higher percentiles by more than an order of magnitude. This indicates that for regularly structured graphs, slightly perturbing the structure drastically increases the expected local search cost (the expected number of flips for solving the regular ring lattice graph, i.e., $p = 0$, is 19043.38). However, as we introduce more randomness into problems, the performance of local search procedures improves.

problem	n	l	density	C	C_{rand}	L	L_{rand}	μ
`hanoi4`	718	3934	1.9%	0.462	0.0187	6.713	2.796	10.278
`ssa0432-003`	435	1027	1.0%	0.445	0.0779	4.273	5.536	4.406
`bf0432-135`	424	1031	1.0%	0.442	0.0833	5.552	4.296	4.108
`par16-1-c`	317	1264	1.8%	0.364	0.0407	3.984	3.521	7.899
`aim-100-1-6-yes1-1`	100	160	7.2%	0.302	0.0747	2.740	2.534	3.745

Table 1. Characteristic path lengths, clustering coefficients and proximity ratios for satisfiability problems with n variables and l clauses from the DIMACS benchmark library. Edge density is the fraction of possible edges in the constraint graph. Problems are both satisfiable and unsatisfiable. `hanoi4` is a encoding of the Towers of Hanoi planning problem. `ssa0432-003` is an encoding of a circuit analysis problem with a "single–stuck–at" fault. `bf0432-135` is an encoding of a circuit analysis problem with a "bridge–fault". `par16-1-c` is an encoding of a problem in learning the 16 bit parity function. `aim-100-1-6-yes1-1` is an artificially generated problem with a single satisfying assignment.

Complete Methods

Comparison of our results with results reported in (Walsh 1998) for coloring randomly rewired graphs using a complete graph coloring algorithm, suggests that the performance of local and complete search methods may differ significantly on small world problems. Local and systematic search methods may therefore complement each other. With a small amount of randomness in problems, complete methods may have fewer difficulties than local search, while with more randomness, the situation seems to reverse. To test this hypothesis, we ran a second series of experiments using "satz" (Li & Anbulagan 1997), one of the best complete search algorithms for SAT.

Figure 2 (b) shows the higher percentiles of the search cost for satz. The performance of satz is clearly affected by the small-worldiness differently to WalkSAT. Median search cost increases monotonically with p up to $\log_2(p) = -1$ and then drops slightly. By comparison, there is a very distinctive peak in the 99% percentile around $p = 0.1$, in the region where the graphs have maximal small-worldiness. Around this peak, the difference between the 95% and 99% percentiles is huge (up to 4 orders of magnitude), and the distribution of search costs across the test set, as well as on individual instances, displays a heavy-tail. This suggests that a randomization and restart strategy will be effective on these problems (Gomes, Selman, & Crato 1997). Similar extremely hard instances were observed in (Walsh 1998), where a small fraction of randomly rewired graphs with high proximity ratios, μ were found to be extremely difficult for a complete graph coloring algorithm.

These results support the hypothesis that local search procedures tend to have difficulties with relatively regular instances which are easy for complete methods, while complete methods tend to have difficulties with more random instances which are easy for local search procedures. This suggests that it might pay to solve these problem by applying both algorithms simultaneously and terminating when either one finds a solution. Such "portfolios" of algorithms have been proposed in (Huberman, Lukose, & Hogg 1997; Gomes, Selman, & Kautz 1998). With such an approach, the overall performance and, perhaps even more importantly, the robustness of problem solving can be improved. The optimal mix of the algorithms in such a portfolio depends on the exact performance of each individual algorithm. The median run time (as well as the higher percentiles) for WalkSAT and satz cross over approximately where the peak in small-worldiness is located. This suggests that small-worldiness may be a structural property which tends to give *minimal* leverage for portfolio combinations of WalkSAT and satz.

Encoding Small Worlds

In recent years, there has been considerable success encoding many different search problems into SAT and using either a fast local search method like WalkSAT or an efficient complete procedure like satz. Does encoding into SAT preserve or destroy the topological structure? To measure the topology of a satisfiability problem, we construct the constraint graph, in which nodes are the propositional variables, and edges connect any variables that occur together in a clause.

We ran a test on the DIMACS satisfiability benchmark. Many of the encodings of problems in this library have a small world topology. Table 1 gives a representative sample of results. Path lengths are usually comparable to that of a random graph. However, nodes are often more clustered than in random graphs of a similar density. Other types of encoding can disguise the topological structure inherent in a problem. For example, if we encode numbers into SAT using a logarithmic representation on the binary bits (an encoding that has been used for factorization and Hamiltonian circuit problems) then the topological structures may be lost within the complexity of the encoding. However, we conjecture that encodings that are good for search will tend to preserve topological structure. Indeed, they may even emphasize it.

Morphing Quasigroups

A quasigroup is a Latin square, a m by m multiplication table in which each entry appears just once in each row or column. Quasigroups model a variety of practical problems like tournament scheduling and designing drug tests, and have been proposed as the basis of a benchmark for constraint satisfaction algorithms (Gomes & Selman 1997). The constraint graph for such problems consist of $2m$ cliques, one for each row and column, with each clique being of size m. (Walsh 1998) shows that, for large m, the constraint graph of quasigroup problems have a small world topology, with a proximity ratio $\mu \approx m/4$.

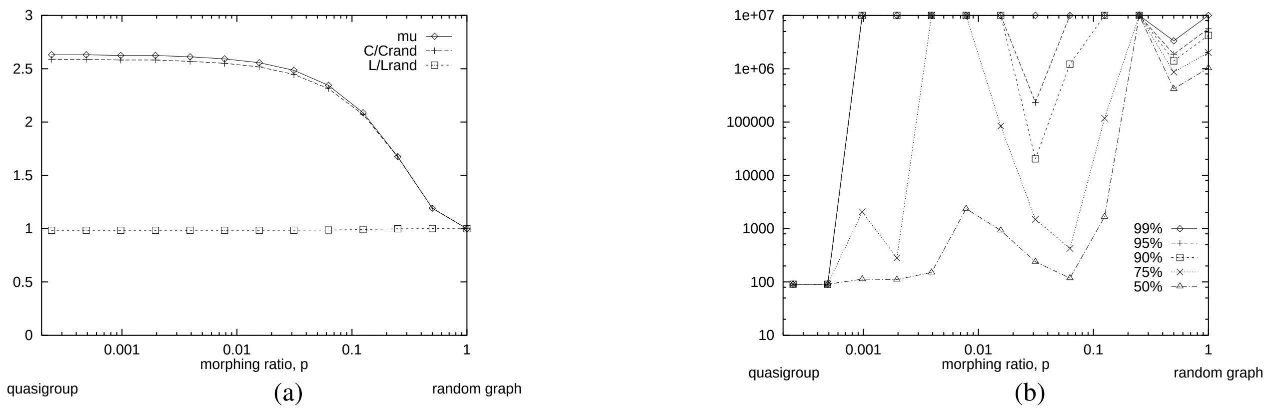


Figure 3. Morphs generated between an order 10 quasigroup and a random graph: **(a)** characteristic path length, clustering coefficient (scaled by their values for a random graph with the same number of nodes and edges) and proximity ratio, μ; **(b)** percentiles in the search cost to color these morphs.

Morphing quasigroups tends to reduce their small-worldiness as we break up the neighborhood clustering. Despite this decrease in small-worldiness, the cost to color such graphs often increases. In Figure 3, we plot the clustering coefficients, characteristic path lengths, proximity ratios and coloring cost for morphs between an order 10 quasigroup and a random graph with the same number of nodes and edges. We used a graph coloring algorithm which is based upon Brelaz's DSATUR algorithm and imposed a search cutoff at 10^7 nodes. As p increases, we break up the neighborhood structure of the quasigroup and the clustering coefficient drops. The characteristic path length is short and stays relatively constant. As a consequence, the proximity ratio, μ drops along with the clustering coefficient.

Coloring costs for these morphs are often very large. This result suggests that other factors in addition to small-worldiness contribute to the hardness of these coloring problems. Detailed analysis of the search costs shows that the heaviest-tailed distributions occur either with random graphs with a small amount of structure (e.g. $\log_2(p) = -2$) or quasigroups graphs with a small amount of randomness (e.g. $\log_2(p) = -10$ and -8). Inbetween (e.g. $\log_2(p) = -4$ and -6), we see less of a heavy-tail. We conjecture that the addition of a little randomness to structure, or a little structure to randomness can confuse search heuristics and make graphs hard to color.

Morphing TSPs

Symmetry *v.* Asymmetry

What impact does symmetry have on the TSP problem? In the Manhattan delivery example from the introduction, adding one-way streets introduces a certain amount of asymmetry into the distance matrix and usually made problems easier. Is it generally true that asymmetry will tend to make TSP problems easier? To test this, we ran an experiment morphing directly between a symmetric and an asymmetric distance matrix.

To reduce variance in inter-city distances, we took a symmetric distance matrix and made it asymmetric by permuting the lower lefthand triangle. We then found the optimal tour for type B morphs between the original symmetric problem and this asymmetric problem. We generated symmetric distance matrices with between 12 and 30 cities, normally distributed with a mean inter-city distance of 10^3 and a standard deviation of 10^2. To find the optimal tour, we used a branch and bound algorithm with the Hungarian heuristic for branching.

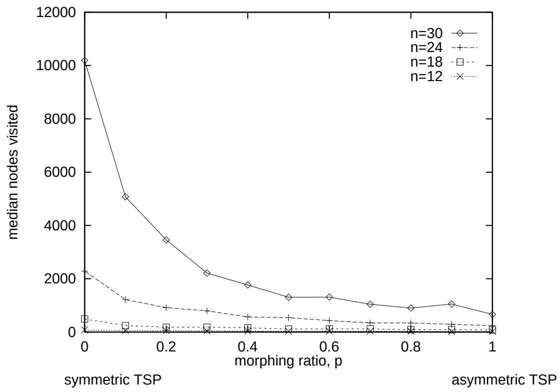
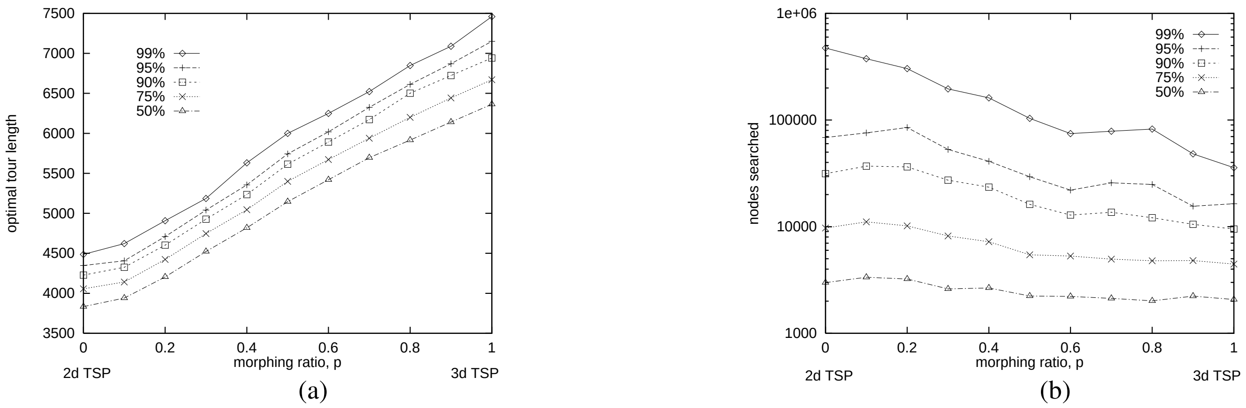


Figure 4. Median cost to find optimal tour for morphs between symmetric and asymmetric TSPs. Other percentiles and mean cost are similar. Inter-city distances are normally distributed with a mean of 10^3 and a standard deviation of 10^2. Each data point is the average of 100 problems.

In Figure 4, we plot the search cost to find the optimal tour. Whilst the optimal tour length decreases by just a few percent as we introduce asymmetry into the problems, the cost to find the optimal tour drops dramatically. We conjecture that asymmetry reduces search by reducing the amount of non-determinism in the problem. If the distance from A to B is significantly shorter than that from B to A, then an optimal tour that includes a leg between A and B will almost certainly visit A first. Unlike the symmetric case, we probably need not consider tours that visit B first.



(a) (b)

Figure 5. Percentiles in (a) optimal tour length and (b) cost to find optimal tour for 20 city TSP problems constructed by morphing between two and three dimensions. In the 3-d case, cities are distributed uniformly and at random within a cube of side 1000. In the 2-d case, we project the 3-d problems down onto the x-y plane. Each data point is the average of 1000 problems.

2-d *v.* 3-d

What impact does the dimensionality of a TSP problem have on optimal tour length and cost to find it? To explore this question, we ran an experiment morphing between 2-dimensional and 3-dimensional TSP problems. We took points randomly distributed in a 3-d cube and flattened them onto the x-y plane. We then constructed type C morphs between the coordinate vectors for the original 3-d problem and those for the flattened 2-d problem. As we increase the morphing ratio p, the points gradually rise off the x-y plane and the z dimension starts to contribute to the problem.

In Figure 5, we plot the optimal tour length and search cost for 20 city problems, with cities distributed uniformly and at random within a cube of side 1000. We again used a branch and bound algorithm with the Hungarian heuristic for branching. As we introduce a third dimension into the problem, cities become further apart and the optimal tour length increases. However, the cost to find the optimal tour tends to drop. For the 95% and lower percentiles, there appears to be a slight rise in cost for small p, but aside from this, search cost drops uniformly. We conjecture that a third dimension increases the variance in inter-city distances, and this makes the optimal tour more obvious.

To conclude, symmetric and 2-dimensional TSP problems appear to be harder than asymmetric and 3-dimensional problems. This is a valuable result as many TSP problems met in practice are likely to be both symmetric and 2-dimensional.

Related Work

Monasson *et al.* introduce the 2+p-SAT model to study the change from a "second order" phase transition for random 2-SAT to a "first order" transition for random 3-SAT (Monasson *et al.* 1996). In the 2+p-SAT model, problems are randomly generated with pl clauses of length 3, and $(1-p)l$ clauses of length 2. Such problems can also be generated by morphing between random 2-SAT problems and random 3-SAT problems. They predict that for $p < 0.413\ldots$, a phase transition in satisfiability will occur around a ratio of clauses to variables of $1/(1-p)$. Achlioptas *et al.* prove this result for $p \leq 0.4$, and provide upper and lower bounds on the location of the phase transition for $p > 0.4$ (Achlioptas *et al.* 1997). This is a rather surprising result since it means that, for $p < 0.4$, the non-binary clauses are "irrelevant" and the binary clauses alone are enough to make problems unsatisfiable.

Gomes and Selman have proposed random quasigroup completion problems as a benchmark that combines together some of the best features of randomly generated instances and highly structured problems (Gomes & Selman 1997). Quasigroup completion is the problem of coloring a partial colored quasigroup. The preassignment of colors perturbs the problem by adding unary constraints. However, the perturbation of the binary constraints performed in morphing quasigroups appears to give more demanding problems. For example, an algorithm that maintains generalized arc-consistency can solve almost all quasigroup completion problems up to order 25 with just a few branches of search (Shaw, Stergiou, & Walsh 1998). By comparison, even with such powerful propagation techniques, quasigroup morphs of order 25 are often very hard to color.

Conclusions

We have proposed a very general mechanism called "morphing" for introducing structure or randomness into a wide variety of problems. Many different types of structures can be morphed including graphs, matrices, vectors, relations, and clauses. To illustrate the usefulness of morphing, we performed several different experimental studies. These studies identify the impact of a "small-world" topology on the cost of coloring graphs, and the benefits of a portfolio of algorithms for solving such problems; of symmetry in the cost matrix on the cost of finding the optimal TSP tour; and of the dimensionality of space on the cost of finding the optimal TSP tour.

What general lessons can be learnt from this study? First, many problems met in practice may be neither completely structured nor completely random but something inbetween. Morphing provides us with a general purpose mechanism for modelling such problems. Second, a mixture of structure and randomness can make problems very hard to solve. A little structure added to a random problem, or a little randomness added to a structured problem may be enough to mislead our search heuristics. And third, morphing provides many of the advantages of random and structured problem classes without some of the disadvantages. As in random problem classes, we can generate large, and statistically significant samples with ease. However, unlike random problems, the problems generated can contain the sort of structures met in practice.

References

Achlioptas, D.; Kirousis, L.; Kranakis, E.; and Krizanc, D. 1997. Rigorous results for random (2+p)-SAT. In *Proceedings of RALCOM-97*, 1–10.

Bollobás, B. 1985. *Random Graphs*. London: Academic Press.

Gomes, C., and Selman, B. 1997. Problem structure in the presence of perturbations. In *Proceedings of the 14th National Conference on AI*, 221–226. American Association for Artificial Intelligence.

Gomes, C.; Selman, B.; and Crato, N. 1997. Heavy-tailed distributions in combinatorial search. In Smolka, G., ed., *Proceedings of Third International Conference on Principles and Practice of Constraint Programming (CP97)*, 121–135. Springer.

Gomes, C.; Selman, B.; and Kautz, H. 1998. Boosting combinatorial search through randomization. In *Proceedings of 15th National Conference on Artificial Intelligence*, 431–437. AAAI Press/The MIT Press.

Huberman, B.A.; Lukose, R.M.; and Hogg, T. 1997. An Economics Approach to Hard Computational Problems. In *Science* 275:51–54.

Li, C., and Anbulagan. 1997. Look-ahead versus look-back for satisfiability problems. In *Proceedings of Third International Conference on Principles and Practice of Constraint Programming (CP97)*, LNCS, 341–355. Springer Verlag.

Monasson, R.; Zecchina, R.; Kirkpatrick, S.; Selman, B.; and Troyansky, L. 1996. Phase transition and search cost in the 2+p-SAT problem. In *Proceedings of 44th Workshop on Physics and Computation (PhysComp96)*. Boston University.

Shaw, P.; Stergiou, K.; and Walsh, T. 1998. Arc consistency and quasigroup completion. In *Proceedings of ECAI-98 Workshop on Non-binary Constraints, Brighton*.

Walsh, T. 1998. Search in a small world. Technical report, Department of Computer Science, University of Strathclyde, APES-07-1998. Available from http://www.cs.strath.ac.uk/ apes/reports/apes-07-1998.ps.gz.

Watts, D., and Strogatz, S. 1998. Collective dynamics of 'small-world' networks. *Nature* 393:440–442.