Technical Report CSM-371, Department of Computer Science, University of Essex, April 2002
Presented in the UK Local Search Workshop, OR Society (UK),
City University, London UK, 16-17 April 2002, Paper T15

Extending Guided Local Search – Towards a Metaheuristic Algorithm With No Parameters To Tune

Edward Tsang, Patrick Mills and John Ford
Department of Computer Science, University of Essex, Wivenhoe Park, Colchester CO4 3SQ, UK
Phone: +44 01206 87{2771,2774,2787}
E-mail: {millph,edward,fordj}@essex.ac.uk
URL: http://cswww.essex.ac.uk/CSP/

Technical Report CSM-371
April 2002

Abstract. Guided Local Search is a general penalty-based optimisation method that sits on top of local search methods to help them escape local optimum. It has been applied to a variety of problems and demonstrated effective. The aim of this paper is not to produce further evidence that Guided Local Search is an effective algorithm, but to present an extension of Guided Local Search that potentially has no parameter to tune. Compared to other algorithms, Guided Local Search is relatively easy to apply, as there is only one major parameter (λ) to set. In some applications, performance of Guided Local Search is insensitive to the value of this parameter. Nevertheless, the value of this parameter can affect the performance of Guided Local Search in some problems. In this paper, we show how (a) an aspiration criterion and (b) random moves may be added to Guided Local Search to reduce the sensitivity of its performance to the parameter value. The extended Guided Local Search is tested on the SAT, weighted MAX-SAT and Quadratic Assignment Problems with positive results.

Keywords: local search, meta-heuristics, SAT, MAX-SAT, quadratic assignment problem (QAP)

1 Introduction

Guided Local Search (GLS) is a general penalty-based optimisation method [36]. It belongs to a class of methods called meta-heuristics, which sit on top of local search methods to help them escape local optimum. Other meta-heuristics include various forms of Tabu Search [10, 11, 11] and Simulated Annealing [1, 16].

GLS has been applied to a variety of problems and demonstrated effective (e.g. see [35, 37, 38, 39, 29]). The aim of this paper is not to produce further evidence on the effectiveness of GLS. The aim here is to present an extension of GLS that potentially has no parameter to tune.

Compared to other meta-heuristic algorithms, GLS is relatively easy to apply, as there is only one major parameter, namely λ (see explanation below), to set. In some applications, performance of GLS is insensitive to the value of λ. Nevertheless, the value of λ can affect the performance of GLS in some problems. In this paper, we show how GLS can be extended to make it relatively insensitive to the value of λ; these extensions include:

(a) Linking λ to the fitness of the first local optima found;

(b) Adding aspiration to GLS

(c) Adding random moves to GLS.

For each extension, we attempt to (at least partially) answer the following questions:

(i) Does it help GLS in any application at all?

(ii) If it does help, when and why?

(iii) Does it harm GLS in any applications?

The promise of these extensions is supported by empirical evidence from applying the extended Guided Local Search to the SAT, weighted MAX-SAT and Quadratic Assignment Problems.

2 Background: Guided Local Search

Guided Local Search (GLS) [36] was an extension of GENET [5, 6, 34, 42, 42], a neural network approach to constraint satisfaction. While GENET attempts to find any solution that satisfies all the constraints, GLS attempts to find optimal solutions, according to a given function.

GLS borrowed the idea of penalties in OR (e.g. see Koopman [17], Stone [30] and Luenberger [23]). As a meta-heuristic method, GLS sits on top of local search algorithms. In a local search, one searches in the space of candidate solutions[1]. To apply GLS, one defines a set of features for the candidate solutions[2]. GLS associates a *cost* and a *penalty* to each feature. The costs can normally be defined by the objective function[3]. The penalties are initialised to 0 and will only be increased when the local search reaches local optimum.

Given an objective function *g* that maps every candidate solution *s* to a numerical value, GLS defines a function *h* that will be used by local search (replacing *g*):

$$h(s) = g(s) + \lambda \times \sum(p_i \times I_i(s)) \qquad (1)$$

where *s* is a candidate solution, λ is a parameter to the GLS algorithm, *i* ranges over the features, p_i is the penalty for feature *i* (all p_i's are initialised to 0) and I_i is an indication of whether s exhibits feature *i*:

$$I_i(s) = 1 \text{ if } s \text{ exhibits feature } i\text{; 0 otherwise.} \qquad (2)$$

[1] The solution representation issue is significant, though it is not the subject of our discussion in this paper.

[2] For example, in the traveling salesman problem, a feature could be “whether the candidate tour visits city B immediately after visiting city A”.

[3] For example, in the traveling salesman problem, the cost of the above feature can simply be the distance between cities A and B.

Sitting on top of local search algorithms, GLS helps them to escape local optima in the following way. Whenever the local search algorithm settles in a local optimum, GLS augments the cost function by adding penalties to selected features.

The novelty of GLS is mainly in the way that it selects features to penalize. The intention is to penalize "unfavourable features" or features that "matter most" when a local search settles in a local optimum. The feature that has high cost affects the overall cost more. Another factor that should be considered is the current penalty value of that feature. The utility of penalizing feature *i*, $util_i$, under a local optimum s^*, is defined as follows:

$$util_i(s^*) = I_i(s^*) \times c_i / (1 + p_i) \qquad (3)$$

where c_i is the cost and p_i is the current penalty value of feature *i*. In other words, if a feature is not exhibited in the local optimum (indicated by I_i), then the utility of penalizing it is 0. The higher the cost of this feature (the greater c_i), the greater the utility of penalizing it. Besides, the more times that it has been penalized (the greater p_i), the lower the utility of penalizing it again. In a local optimum, the feature(s) with the greatest *util* value will be *penalized*. When a feature is penalized, its penalty value is always increased by 1. The scaling of the penalty is determined by λ.

Following is the general GLS procedure:

Procedure **GLS** (input: an objective function *g;* a local search strategy ***L***; features and their costs; parameter λ)

1. Generate a starting candidate solution randomly or heuristically;
2. Initialise all the penalty values (p_i) to 0;
3. Repeat the following until a termination condition (e.g. a maximum number of iterations or time limit) has been reached:
   3.1. Perform local search (using ***L***) according to the augmented function *h* (which is *g* plus the penalty values, as defined in (Eq. 1) above) until a local optimum s* has been reached;
   3.2. For each feature i exhibited in s* compute $util_i = c_i / (1 + p_i)$
   3.3. Penalize every feature i such that $util_i$ is maximum: $p_i = p_i + 1$;
4. Return the best candidate solution found so far according to the objective function *g*.

Apart from helping the local search to escape local optimum, penalties also helps GLS to rationalise its search effort: parts of the search space that exhibit penalised features will not be searched as thoroughly as space that do not exhibit any penalised features[4].

Naturally the choice of the features and the setting of λ may affect the efficiency of a search. In many of the problems that we have studied, the features come directly from the objective function. The only parameter that needs serious tuning is λ. This is the focus of this paper. The aim is to reduce the need to fine tune this parameter for individual problems or different problem classes.

3 Using the first local optima to set the parameter λ

As shown in equation (1), if the value of λ is too small, then the penalty terms will not have significant effect on the augmented function. The consequence is GLS will take a long time to escape local optimum. If the value of λ is too high, the penalty terms will dominate the search. Any penalty posed on a feature that might be exhibited by an optimal solution could seriously prevent the search from finding this optimal solution.

Our aim is to involve as little domain-specific knowledge as possible in setting λ. When Voudouris and Tsang applied GLS to the travelling salesman problem, they λ is set to the following value [39]:

[4] However, not all penalised features will be avoided all the time. Therefore, penalties can be seen as *soft taboos* in Tabu Search [9, 10, 11]. In that sense, it can be argued that GLS is a form of Tabu Search.

$$\lambda = a \times g(s^*) / N \quad (4)$$

where s* is the first local optima found by the local search; g(s*) is the cost of s* in the travelling salesman problem; N is the number of cities in the problem; a is new parameter to tune; from the range (0, 1]. In other words, the problem of tuning λ becomes the problem of tuning a, which we shall refer to as the *λ-coefficient*.

Larger travelling salesman problems tend to have larger optimal values. The use of g(s*) ensures that the penalty terms are scaled to the cost of the original function to be optimised. Using Eq. (4) avoids tuning λ for individual problems. There is hope for finding a value for a that suits all travelling salesman problems.

Can this strategy of setting λ be generalised to other problems? Notice that domain knowledge still plays a part in Eq. (4): g(s*) / N is the average cost of the edges in the tour represented by the local optima s*. Since the value of a is no greater than 1, each penalty term in the augmented function will have comparable values to the terms in the original function in the travelling salesman problem.

One possibility to generalise this strategy to setting λ for other problems is to make:

$$\lambda = a \times C \quad (5)$$

where C is a value that roughly reflects the cost of the problem. Learning from our experience in the travelling salesman problem (and later the quadratic assignment problem, QAP), C should be roughly comparable to the average value of each term in the given cost function (g).

To allow for inaccuracy in C, we do not limit the value of the λ-coefficient a to (0, 1]. Finding a value for λ-coefficient is still easier than finding a value for λ, as C has taken care of the scaling factor. Some of the remaining questions are:

(a) How sensitive would the performance of GLS be with regard to the value of the λ-coefficient?
(b) Are there means to reduce the sensitivity?
(c) How easy is it to find good values for the λ-coefficient?

A number of extensions have been looked at, with the intention to improve the performance of GLS. In the following sections, we shall present extensions for answer questions (b) above.

4 Adding Aspiration to Guided Local Search

Aspiration moves were used in Tabu Search [10, 11, 11]. Given a local search ***L*** and a function *g* to minimize[5], an *aspiration criterion* for GLS is defined as follows:

> Let *bsf* be the best solution found so far according to *g*. A move from a candidate solution *s* to *s'* is accepted by the local search ***L*** if g(*s'*) < g(*bsf*) but h(*s'*) > h(*s*), where *h* is the augmented cost function.

In other words, an *aspiration move* in GLS is a move that would normally be rejected by the local search (which operates under the augmented cost function), but only accepted if the aspiration criterion is adopted.

The intuition behind aspiration moves is this: since *g* is the function to optimise, there is no reason to ignore any good solutions just because we use an augmented function (*h*) to help escape local optimum and rationalise our search effort.

GLS uses utilities to heuristically pick features to penalise. There is no guarantee that the feature being penalised will not appear in an optimal solution. If a feature f_i is penalised, and f_i is actually exhibited by an optimal solution s*, then it may take many more penalties on other features before the search could enter the basin of attraction[6] of s*. This problem is potentially more serious if λ-coefficient is set to a relatively high value. Aspiration moves temporarily removes the effect of the penalties. If aspiration moves were to

[5] An aspiration criterion for maximization problems can be defined similarly.

[6] Given a local search algorithm ***L***, the basin of attraction for a local optima s* is the set of all candidate solutions that necessarily lead to s* under ***L***.

be useful to GLS, one would expect them to be more useful when GLS uses relatively high λ-coefficient values. This conjecture is empirically supported (see summary below, or details in [25]).

5 Adding Random Moves to Guided Local Search

Randomness (sometimes referred to as "noise") plays an important part in heuristic search (e.g. see Walksat [28, 28, 13]), especially Simulated Annealing [1, 16]. A *random move* in GLS means a move by the local search to a random neighbour, regardless of its fitness (according to the original or augmented function).

Random moves helps to move out of basins of attraction. It is a simple way to escape local optimal. Therefore, one could say that the function of random moves overlaps that of penalties in GLS. However, random moves could compensate GLS should the λ-coefficient be set to a value that is too small. On the other hand, too much randomness defeats the purpose of local search. Therefore, some of the questions are:

- Would random moves really help GLS when λ-coefficient is too small?
- Would random moves degrade the performance of GLS when the λ-coefficient value is not too small?
- How much randomness should one allow in GLS?
- Does this level of randomness need tuning from problem to problem, or from problem class to problem class?

We extended GLS by giving it, in every move, a probability *Pr* to make a random move. We conducted pilot experiments to test the effect of different combinations of *Pr* and λ-coefficient [25]. It was found that *Pr* = 0.2 worked well with most λ-coefficients.

With *Pr* = 0.2 fixed, we extensively tested the effectiveness of this level of randomness in GLS, varying λ-coefficient. Empirical results are summarised in the next section (see [25] for details). They basically support the fact that randomness helped GLS when the value of λ-coefficient is too small, without seriously degrading the performance of GLS when λ-coefficient is not too small. Random moves worked particularly well with aspiration in QAPs. Our experimental results so far seem to indicate that *Pr* = 0.2 worked well in GLS in general, without needing any fine-tuning to suit individual problems or problem classes.

6 Extended Guided Local Search

Readers should be reminded that our goal is to extend GLS so that there is no need to fine tune λ for individual problems or problem classes. Our first extension is to turn the problem of tuning λ into tuning λ-coefficient. Our analysis and experiments suggested that, at least in some problem classes, aspiration moves enhances the performance of GLS when λ-coefficient is too large, while random moves enhances the performance of GLS when λ-coefficient is too small. Therefore, we give the name Extended Guided Local Search to the following algorithm:

GLS with λ-coefficient replacing λ, aspiration and random moves

We tested the following algorithms:

- GLS with λ-coefficient replacing λ
- GLS with λ-coefficient replacing λ and aspiration moves (GLS+A)
- GLS with λ-coefficient replacing λ and random moves (GLS+R); and
- GLS with λ-coefficient replacing λ and aspiration and random moves (GLS+A+R)

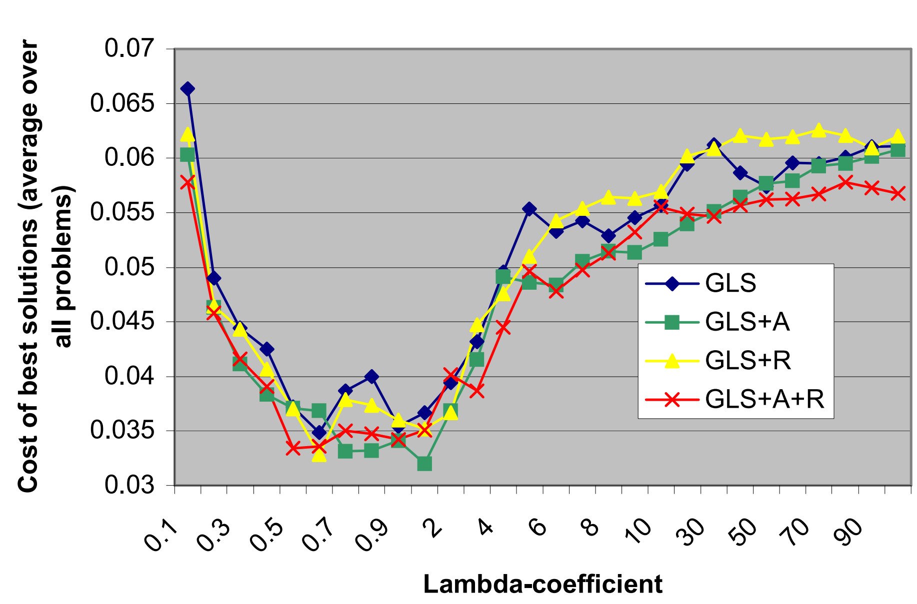
on three classes of problems (see [25] for details):

- SAT: the *satisfiability* problem where Boolean variables represent the truth values of propositions and constraints are expressed in conjunctive normal form; the task is to find truth values for all the variables satisfying all the constraints [8];
- Weighted MAX-SAT: this is an *optimisation* problem which is based on the SAT problem, except that each clause is given a weight; the task is to minimize the total weights of the unsatisfied clauses [14, 24];

- Quadratic Assignment Problem (QAP): this is an extensively studied *optimisation* problem in Operations Research [4, 25].

These problem classes are very different in nature. The landscape of SAT problems mainly comprises large plateaus. Although the MAX-SAT problem appears similar to the SAT problem, its landscape is much more ragged. The QAP has a more complex cost function than MAX-SAT, as the cost of each assignment depends on the values assigned to other variables. The performance of the GLS variants on MAX-SAT and QAP are shown in Figures 1 and 2. On SAT, all GLS variants exhibit similar performance.

Figure 1: GLS Applied to MAX-SAT



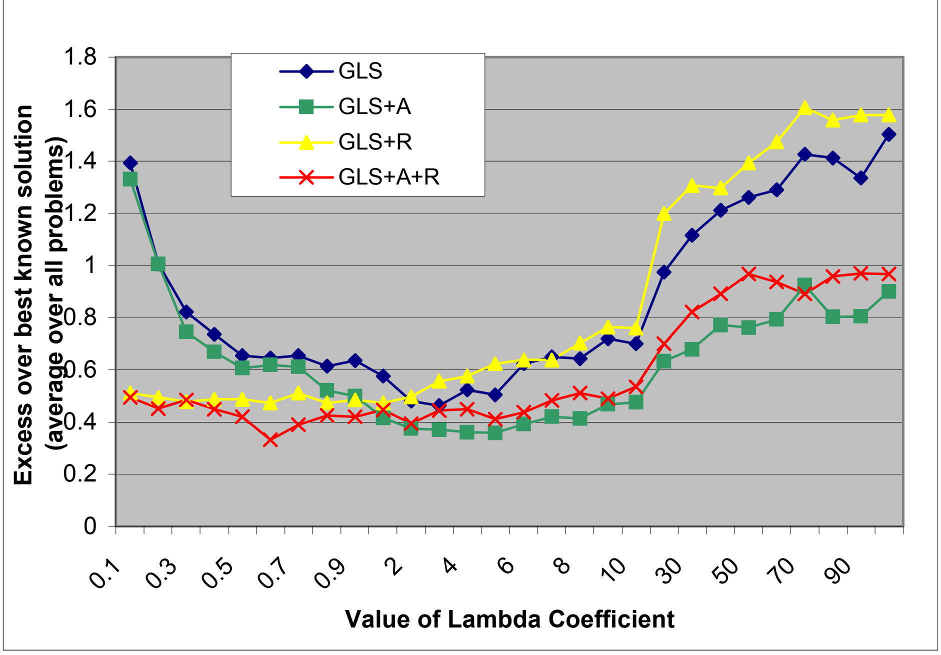


Figure 2: GLS applied to QAP

Results in figure 1 and 2 show that aspiration and random moves complement each other. The performance of GLS+A+R is quite close to the better of GLS+A and GLS+R as λ-coefficient changes its value. This suggests that aspiration and random moves do not interfere with each other. Table 1 summarises the results.

<table>
<tr><th></th><th>SAT</th><th>MAX-SAT</th><th>QAP</th></tr>
<tr><td>GLS+A</td><td rowspan="6">No significant difference among the different variations of GLS</td><td>Under all λ-coefficients, GLS+A produced solutions as good as, and often better, than results by GLS alone on MAX-SAT</td><td>Under all λ-coefficients values, GLS+A produced better results than GLS alone; the gap between GLS and GLS+A grows as λ-coefficients grow, especially after λ-coefficients = 0.8</td></tr>
<tr><td rowspan="4">GLS+R</td><td>When λ-coefficient is ≤ 0.8: GLS+R produced solutions better than those produced by GLS alone</td><td>When λ-coefficient is ≤ 1: GLS+R produced better results than GLS alone</td></tr>
<tr><td>When λ-coefficient is between 0.9 and 6: GLS+R is sometimes better than GLS alone, sometimes not.</td><td rowspan="2">When λ-coefficient is > 1: GLS+R produced worse results than GLS alone</td></tr>
<tr><td>With λ-coefficient > 6, GLS+R mostly performed poorer than GLS</td></tr>
<tr><td>GLS+A+R</td><td>GLS+A+R out-performed GLS alone under all λ-coefficient, with only one exception (λ-coefficient=2).</td><td>GLS+A+R produced better results than GLS alone under all λ-coefficients values. In fact, GLS+A+R performed better than GLS+R in all but one points (λ-coefficient=0.3).</td></tr>
<tr><td colspan="4">Table 1: Summary of performance of different GLS variants in SAT, MAX-SAT and the QAP
Keys: GLS = Guided Local Search; +A = GLS with Aspiration; +R = GLS with Randomness</td></tr>
</table>

7 Discussions

How easy is it to use metaheuristic methods? Tabu Search and Simulated Annealing are both relatively easy to use. Sitting them on top of local search methods could potentially improve the effectiveness of the local search. Basic Tabu Search is simple to use. There is only one parameter to set, namely the length of the taboo list. However, Tabu Search is a class of algorithm, within which there are many variations to choose from. To use simulated annealing, one needs to define the cooling schedule. Our aim is to make GLS easier for users by fixing both the control strategy (GLS+A+R) and parameter setting in Extended GLS.

Figure 1 shows that the setting of λ-coefficient is still relevant to the performance of GLS. But experiments in all three applications seem to indicate that performance of GLS is reasonable when *Pr* is set to 0.2 and λ-coefficient is set to1.

8 Conclusions and Future Work

There is no attempt to claim in this paper that GLS will get solutions of the best quality[7]. The aim of this work is to extend GLS to one of the easiest algorithms to use. The basic control strategy has been proved effective for a wide range of applications. The focus here is on how to set the parameter λ.

The Extended Guided Local Search algorithm has been tested on SAT, MAX-SAT and QAP. The basic algorithm and its parameters were kept the same for all problems. The following conclusions can be drawn:

Conclusion 1: Adding aspiration and random moves to Guided Local Search improved its performance under certain λ-coefficient values.

Conclusion 2: In the problems that we have tested so far, aspiration and random moves have not caused any significant degradation in performance in GLS under any λ-coefficient values.

This gives hope for finding an algorithm that does not have any parameter to tune carefully (with *Pr*=0.2 and λ-coefficient=1). We intend to verify these results in more problem classes.

In our experiments, we observe a number of measures to monitor the performance of the extended GLS. The hope is to find out more about when and why GLS and its individual components (including the extensions) work.

GLS-Solver 1.0 is a piece of software that allows researchers to run GLS on SAT, MAX-SAT and QAP problem instances [25]. It includes an option to output detailed monitored measures.

It is worth mentioning that GLS has not only been used to help improving the performance of local search. GLS has been used to guide Genetic Algorithms. The resulting algorithms, Guided Genetic Algorithm (GGA) [21], achieved robust, outstanding, results in the General Assignment Problem [19], the Processors Configuration Problem [18, 20] and the Radio Length Frequency Assignment Problem [22]. Replacing GLS with Extended GLS in GGA would be a worthwhile exercise.

Acknowledgements

Chang Wang and Andrew Davenport played major roles in the development of GENET. Chris Voudouris invented the original Guided Local Search algorithm. Tung Leng Lau applied Guided Local Search in Genetic Algorithms. This project was partially sponsored by EPSRC funded projects GR/H75275 and GR/L20122 and Research Promotion Funds by the University of Essex. James Borrett, Jim Doran and Paul Scott all contributed to this project.

[7] In fact, we have not picked the problems in which GLS performed best – results it found in MAX-SAT were better than other algorithms [24], but it was only as good as WalkSAT [28] in SAT and no better than Taillard [31] in QAP.

Bibliography

1. Aarts, E. & Korst, J., Simulated Annealing and Boltzmann Machines, John Wiley & Sons, 1989
2. Battiti, R. & Tecchiolli, G., The Reactive Tabu Search. In ORSA Journal on Computing, 6(2), 1994, 126-140
3. Bishop, C.M., Hinton, G., Neural Networks for Pattern Recognition. Clarendon Press 1995
4. Burkard, R.E., Karisch, S.E. & Rendl, F., QAPLIB - A Quadratic Assignment Problem Library. In Journal of Global Optimization 10, 1997, 391-403
5. Davenport A., Tsang E.P.K., Zhu, K. & Wang C.J., GENET: A Connectionist Architecture for Solving Constraint Satisfaction Problems by Iterative Improvement, in Proceedings of AAAI, 1994, 325-330
6. Davenport, A., Extensions and Evaluation of GENET in Constraint Satisfaction. PhD Thesis, Department of Computer Science, University of Essex, Colchester, UK, July, 1997
7. Fleurent, C. & Ferland, J.A., Genetic Hybrids for the Quadratic Assignment Problem. In Pardalos, P. & Wolkowicz, H., (eds.), Quadratic Assignment and Related Problems, Vol. 16, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, 1994, 173-187
8. Gent, I.P., van Maaren, H. & Walsh, T., SAT2000, Highlights of satisfiability research in the year 2000, Frontiers in Artificial Intelligence and Applications, IOS Press, 2000
9. Glover, F., Tabu search Part I, in Operations Research Society of America (ORSA) Journal on Computing, Vol. 1, 1989, 109-206
10. Glover, F., Tabu search Part II, in: Operations Research Society of America (ORSA) Journal on Computing, Vol. 2, 1990, 4-32
11. Glover, F.W. & Laguna, M., Tabu Search, Kluwer Academic Publishers, 1997
12. GLS-Solver 1.0, Software (under Windows NT/2000) for applying Guided Local Search to SAT, MAX-SAT and Quadratic Assignment Problems, Constraints and Optimisation Research Group, Department of Computer Science, University of Essex, http://cswww.essex.ac.uk/CSP, May 2002
13. Gomes, C.P. & Shmoys, D.B., The promise of LP to boost CSP techniques for combinatorial problems, Proc., Fourth International Workshop on Integration of AI and OR techniques in Constraint Programming for Combinatorial Optimisation Problems (CP-AI-OR'02), Le Croisic, France, 25-27 March 2002, 291-305
14. Jiang Y., Kautz H., and Selman B., Solving Problems with Hard and Soft Constraints Using a Stochastic Algorithm for MAX-SAT, 1st International Joint Workshop on Artificial Intelligence and Operations Research, 1995
15. Kilby, P., Prosser, P., Shaw, P., Guided Local Search for the Vehicle Routing Problem. In Proceedings of the 2nd International Conference on Metaheuristics, July 1997.
16. Kirkpatrick, S., Gelatt, C.D. Jr. & Vecchi, M.P., Optimization by Simulated Annealing, Science, Vol.220, No.4598, May 1983, 671-680
17. Koopman B.O., The theory of search, part III, the optimum distribution of searching effort, Operations Research, Vol.5, 1957, 613-626
18. Lau, T.L., Tsang, E.P.K., Solving the Processor Configuration Problem with a Mutation-Based Genetic Algorithm. In International Journal on Artificial Intelligence Tools (IJAIT), World Scientific, Vol.6, No.4, 567-585, December 1997.
19. Lau, T.L. & Tsang, E.P.K., The Guided Genetic Algorithm and its application to the General Assignment Problems. In IEEE 10th International Conference on Tools with Artificial Intelligence (ICTAI'98), Taiwan, November 1998.
20. Lau, T.L. & Tsang, E.P.K., Solving Large Processor Configuration Problems with the Guided Genetic Algorithm. In IEEE 10th International Conference on Tools with Artificial Intelligence (ICTAI'98), Taiwan, November 1998.
21. Lau, T.L., Guided Genetic Algorithm. PhD Thesis, Department of Computer Science, University of Essex, 1999.

22. Lau, T.L. & Tsang, E.P.K., Guided genetic algorithm and its application to radio link frequency assignment problems, Constraints, Vol.6, No.4, 2001, 373-398
23. Luenberger, D., Linear and nonlinear programming, Addison-Wesley Publishing Co., Inc., 1984
24. Mills, P. & Tsang, E.P.K., Guided Local Search for solving SAT and Weighted MAX-SAT Problems. In Journal of Automatic Reasoning, Special Issue on Satisfiability Problems, Kluwer, Vol.24, 2000, 205-223
25. Mills, P., PhD Thesis, Extended Guided Local Search, Department of Computer Science, University of Essex, July 2002
26. Pardalos, P.M., Rendl, F., Wolkowicz, H., The Quadratic Assignment Problem: A Survey of Recent Developments. In P. Pardalos and H. Wolkowicz, editors, Quadratic Assignment and Related Problems, volume 16, pages 1-42. DIMACS Series in Discrete Mathematics and Theoretical Computer Science, 1994
27. Selman, B. & Kautz, H., Domain-independent extensions to GSAT: solving large structured satisfiability problems, Proc., 13th International Joint Conference on AI, 1993, 290-295
28. Selman, B., Kautz, H., Cohen, B., Noise Strategies for Improving Local Search. In Proceedings AAAI-94. 1994
29. Shaw, P., Using constraint programming and local search methods to solve vehicle routing problems, in Maher, M. & Puget, J-F. (ed.), Proceedings, 4th International Conference on Principles and Practice of Constraint Programming – CP98, Pisa, Italy, October 1998, Springer Verlag, Lecture Notes in Computer Science, 1520, 416-431
30. Stone L.D., The process of search planning: current approaches and continuing problems, Operations Research, Vol.31, 1983, 207-233
31. Taillard, E.D., Robust Tabu Search for the Quadratic Assignment Problem. In Parallel Computing, 17,443-455, 1991
32. Taillard, E.D., Comparison of Iterative Searches for the Quadratic Assignment Problem. Location Science, 1994
33. Taillard, E.D. & Gambardella, L.M., Adaptive Memories for the Quadratic Assignment Problem. Research Report, IDSIA Lugano, Switzerland, 1997
34. Tsang, E.P.K. & Wang, C.J., A Generic Neural Network Approach for Constraint Satisfaction Problems. In Taylor, J.G. (ed.), Neural network applications, Springer-Verlag, 1992, 12-22.
35. Tsang, E.P.K. & Voudouris, C., Fast Local Search and Guided Local Search and their application to British Telecom's Workforce Scheduling Problem. In Operations Research Letters, Elsevier Science Publishers, Amsterdam, Vol.20, No.3, 119-127, March 1997
36. Voudouris, C. Guided Local Search for Combinatorial Optimisation Problems, Ph.D. thesis, Department of Computer Science, University of Essex, 1997
37. Voudouris, C. & Tsang, E.P.K., Solving the Radio Link Frequency Assignment Problem using Guided Local Search. In Proceedings of NATO Symposium on Radio Length Frequency Assignment, Sharing and Conservation Systems (Aerospace), Aalborg, Demark, October 1998, Paper 14a.
38. Voudouris, C., Guided Local Search - An Illustrative Example in Function Optimisation. In BT Technology Journal, Vol.16, No.3, July 1998, 46-50
39. Voudouris, C. & Tsang, E.P.K., Guided Local Search and its application to the Travelling Salesman Problem. In European Journal of Operational Research, Anbar Publishing, Vol.113, Issue 2, March 1999, 469-499
40. Wilhelm, M.R. & Ward, T.L., Solving Quadratic Assignment Problems by Simulated Annealing. In IIE Transaction, 19/1, 1987, 107-119
41. Wang, C.J. & Tsang, E.P.K., Solving constraint satisfaction problems using neural-networks, Proceedings, IEE Second International Conference on Artificial Neural Networks, 1991, 295-299
42. Wang, C.J. & Tsang, E.P.K., A cascadable VLSI design for GENET, in Delgado-Frias, J.G. & Moore, W.R. (ed.), VLSI for Neural Networks and Artificial Intelligence, Plenum Press, New York, 1994, 187-196