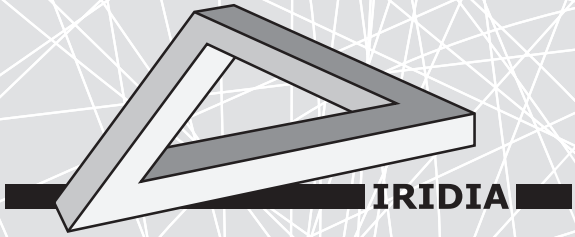

Université Libre de Bruxelles

Institut de Recherches Interdisciplinaires
et de Développements en Intelligence Artificielle

SLS-DS 2009: Doctoral Symposium on Engineering Stochastic Local Search Algorithms

Frank Hutter and Marco A. Montes de Oca

IRIDIA – Technical Report Series

Technical Report No.
TR/IRIDIA/2009-024

August 2009

SLS-DS 2009

Doctoral Symposium on Engineering Stochastic Local Search Algorithms

Edited by:

Frank Hutter, University of British Columbia, Vancouver, Canada
Marco A. Montes de Oca, Université Libre de Bruxelles, Brussels, Belgium

4 September 2009, Brussels, Belgium

IRIDIA – Technical Report Series
ISSN 1781-3794

Published by:

IRIDIA, *Institut de Recherches Interdisciplinaires et de Développements en Intelligence Artificielle*
Université Libre de Bruxelles
Av F. D. Roosevelt 50, CP 194/6
1050 Bruxelles, Belgium

Technical report number TR/IRIDIA/2009-024

Preface

The second **Doctoral Symposium on Engineering Stochastic Local Search Algorithms** (SLS-DS) was held at the Université Libre de Bruxelles, Belgium on 4 September 2009 integrated into the **SLS 2009 Workshop**. SLS-DS consists of a series of short presentations followed by a poster session. It thereby provides a forum for doctoral students to present their work, obtain guidance from fellow researchers, and to network with other students at a similar stage in their careers. The symposium exposes students to helpful criticism and fosters discussions related to future career perspectives.

The extended abstracts in these proceedings were selected based on relevance, significance, quality, and clarity of presentation. They provide a useful guide to emerging research and new trends in the stochastic local search field. The topics covered include:

- Methodological developments for the implementation of SLS algorithms.
- Automated procedures for selecting operators and tuning parameters of SLS algorithms
- Multi-objective optimization
- In-depth experimental studies of SLS algorithms
- Studies of problem characteristics

We would like to thank everyone who has helped making SLS-DS 2009 a success, in particular the members of the program committee, the additional reviewers, and everyone involved in the local arrangements of SLS and SLS-DS 2009.

Frank Hutter & Marco A. Montes de Oca

Program Chairs, SLS-DS 2009

Program Committee

- Alejandro Arbelaez, Microsoft Research/INRIA Joint center, France
- Mauro Birattari, IRIDIA, Université Libre de Bruxelles, Belgium
- Tom Carchrae, Actenum, Canada
- Matteo Gagliolo, IDSIA, Università della Svizzera Italiana, Switzerland
- Holger H. Hoos, CS Department, University of British Columbia, Canada
- Arnaud Liefooghe, Université Lille 1, France
- Manuel López-Ibáñez, IRIDIA, Université Libre de Bruxelles, Belgium
- Horst Samulowitz, Microsoft Research, UK
- Thomas Stützle, IRIDIA, Université Libre de Bruxelles, Belgium

Additional Reviewers

- Alastair Andrew, University of Strathclyde, UK
- Marenglen Biba, Università degli Studi di Bari, Italy
- Saifullah Bin Hussin, IRIDIA, Université Libre de Bruxelles, Belgium
- Matteo Borrotti, Università di Bologna, Italy
- Stefan Eppe, CoDE-SMG, Université Libre de Bruxelles, Belgium
- Chris Fawcett, University of British Columbia, Canada
- Alvaro Fialho, Microsoft Research/INRIA Joint center, France
- Doug Hains, Colorado State University, USA
- Sabine Helwig, Universität Erlangen-Nürnberg, Germany
- Christian Horoba, Technische Universität Dortmund, Germany
- Franco Mascia, Università degli Studi di Trento, Italy
- Jesica Rivero, Universidad Carlos III de Madrid, Spain
- Francesco Sambo, Università di Padova, Italy
- Andrew Sutton, Colorado State University, USA
- Cristina Teixeira, Universidade do Minho, Portugal
- Pieter Vansteenwegen, Katholieke Universiteit Leuven, Belgium
- Nadarajen Veerapen, Université de Nantes, France
- Zhi Yuan, IRIDIA, Université Libre de Bruxelles, Belgium

Contents

A Model Based Algorithm for Evolutionary Design of Experiments

Matteo Borrotti
Ph. D. School of Statistics,
Faculty of Statistics, University of Bologna, Italy
matteo.borrotti@unibo.it

Abstract

In many optimization problems, such as vehicle routing and scheduling, the large domain and the huge experimental space limit the ability of classical approaches to reach the optimum of a given function. A possible solution is the integration of advanced statistical techniques with optimization algorithms, to avoid the need of assumptions on the domain and to allow the methodology to scale to problems of large scale. With this purpose, we combine approaches from Design of Experiments and metaheuristic algorithms. To study the performance of the proposed techniques, we compare different solutions on a simulated case related to protein engineering in biochemistry.

1 Introduction

Problems in life sciences, such as biochemistry, or materials science, involve a huge number of components that influence the behaviour of the experiments. In real experiments we often have to handle a large set of components; among them, one sometimes has to extract a subset of components that, properly ordered and combined together, allow the experimenter to find the best solutions for a specific problem. The process of selecting the best subset and identifying the best ordering can be seen as an optimization problem. The same problem arises in many different domains, such as: routing, assignment, scheduling and others.
In our specific case, we are facing a class of discrete optimization problem that is characterized by a large number of elements to be considered and a huge experimental space to be tested. More precisely, starting from a set A of $|A| = n$ objects, one has to choose a first subset B from A of $|B| = m$ objects ($m \ll n$). The final solution is then built by ordering the elements in B. The objective function value of the final solution depends on the elements that are chosen and their ordering. In this paper, we assume that the objective function is a black-box function and that it is not explicitly given. In fact, in the problems we tackle, the objective function value of a candidate solution is usually obtained by running a simulation. Typically,

computing the objective function value will be a time consuming task and, hence, we will be strongly limited on the number of function evaluations.
A practical example of this problem is the optimization of co-rotating twin screw extruders [7], used in the polymer compounding industry mainly due to their good mixing capacity. In this example, all the objects are already selected and the aim is only to choose the right ordering. In our more general case, the set of objects has to be first extracted from a larger set of elements and then ordered properly.
A practical example of this second case is the development of Synthetic Proteins, an emerging and exciting field in biochemistry. Protein Engineering and Design (PED) has been the object of many studies, due to its relevance in the research of primordial catalysts of the Origin of Life and for their potential applications in biotechnology for highly selective biotransformation process. In this particular case, we want to assess through *in silico* simulation the possibility of using Evolutionary Design of Experiments to identify proteins sequences with defined properties. Automated *de novo* design of Synthetic Proteins is one of the main goals of computational biochemistry and we will focus our efforts in building a solution for this kind of problems and, more generally, to find a methodology able to tackle the aforementioned optimization problem.

2 Methods

The ultimate aim is to test the possibility of exploiting bio-inspired algorithms combined with advanced statistical techniques to search a discrete sequence space for a target structure.
The simulation consists in a number of test experiments on an artificially generated dataset of 95 domains, that will be combinatorially recombined to generate synthetic proteins composed of 200 amino acids (4 domains). A protein domain is a part of protein sequence that folds autonomously within a protein chain. Each domain forms a compact three-dimensional structure and often can be independently stable and folded. Many proteins consist of several structural domains. One domain may appear in a variety of evolutionarily related proteins. Domains vary in length from between about 25 amino acids up to 500 amino acids in length, in our case being 50 amino acids long. Because they are self-stable, domains can be "swapped" by genetic engineering between one protein and another to make synthetic proteins. The dataset of domains will consist of 91 completely random protein domains and 4 natural protein domains. The latter, when recombined in the proper order, generate a protein of know function and structure (*i.e.* Protein Data Bank[1]: 1AGY). The ultimate goal is to identify the 4 natural

[1]The Protein Data Bank (PDB) archive contains information about experimentally-determined structures of proteins, nucleic acids and complex assemblies.

domains among the 95 domains and to order them appropriately, to generate the natural protein; the search space of possible sequences derived from the dataset is rather large ($95^4 = 8.1 \times 10^7$). This approach is based on the work of [5] and [8].
Each individual protein of 200 amino acids length, assembled using the 95 domains, will be evaluated, in a subsequent validation step, for its secondary structure using the *PSIPRED* software [4]. *PSIPRED* output will be used to compare the secondary structure of synthetic proteins against the target protein (*i.e.* 1AGY). Synthetic proteins will be aligned against the target protein and a score will be computed considering individual amino acid secondary structure identity, weighted for confidence of prediction.
The problem of selecting 4 domains among 95 and assigning to them the correct ordering is then mapped to a binary optimization problem in 95×4 dimensions. The optimization algorithm will exploit a statistical model to score each solution and, after each run, will propose a set of candidate proteins; the proteins will then be evaluated in the validation step and results of the validation will be used to update the statistical model. The process will be iterated until a certain stopping criterion is reached.
The research process can then be divided into 3 main steps that are described below.

Model Selection and Validation

In this simulated case, no *a priori* information about the phenomenon is available; thus, we will base our knowledge on the information extracted from the proteins, previously simulated and tested, using complex statistical models. The task here is to see which method is the best for modeling the surface of the problem depending on the elements.
Different statistical models are tested using a benchmark function, built *ad hoc* for the practical problem presented. Candidate models are: multiple regression, weight regression, local polynomial model.
Being the validation phase extremely time consuming, we can even consider to exploit relatively expensive approaches, from a computational point of view, to model the problem. Due to the inherent complexity of protein folding, multiple experiments must be run simultaneously to dissect the interplay between different variables. For this reason, any single experimental batch will be composed of 95 synthetic proteins (the maximum number of proteins that can be experimentally tested at one time). The evaluation of these 95 proteins takes 2-3 days, using state-of-arts prediction software.
This comparison between models allows us to understand the ability of statistical modelling to predict the target and to compare its performance to stochastic local search algorithms.

Model Based Algorithm for Evolutionary Design of Experiments

After having selected the most suited model for our problem, we will build the optimization algorithm on top of it, exploiting the model output as a fitness function. We will compare a number of different optimization algorithms, such as Ant Colony Optimization [1], population-based and stochastic local search algorithms [3], on the same set of initial experimental points and with the same strategy for updating the set; this will allow us to identify the best approach for searching in the experimental space.
The search space will then be explored with a mixed approach, exploiting both information from the predictive statistical model and the abilities of the Stochastic Local Search algorithm: the iterated refinement of the statistical model will provide the optimization algorithm with a fitness function that increases in accuracy during the optimization process.

Generalization of the problem

We will generalize the problem of selecting and ordering of a sequence (solution) of elements chosen from a huge number of candidate elements, comparing different optimization algorithms and different statistical modelling techniques with complex benchmark functions able to simulate the general features of our problems.

3 Conclusion and state of the research

In our work, we intend to tackle the general problem of selecting the correct subset from a huge set of possible elements and defining their positions in the selected sequence. The main feature of this kind of problems is the large experimental space, due to the huge library of candidate elements to be selected, the different ways in which elements can be composed together, the different laboratory protocols and the complex high order interaction network among the elements.
Usually researches try to solve these problems using classical techniques from statistical Design of Experiments [6], which are not effective when the number of independent variables increases; moreover, they assume the presence of priori knowledge on the systems. In all practical examples these assumptions tend to be too strict. The main objective of this work is to define a new way of designing experiments based on the combination of advanced statistical techniques and optimization algorithms, in order to find the best experimental design and to reduce cost and time.
We started from the ideas presented in Theis et al [9], in which a biochemical problem is solved with an evolutionary design of experiments based on a genetic algorithm (GA), and from the approach used by Forlin et al. [2], in which the potentialities of adopting this kind of evolutionary approaches for

high dimensional mixture experiments are described.
At the moment, a simulated dataset of domains is ready to be analysed and we are going to finish the first step of the research process, the selection and the evaluation of the statistical models. Moreover, we are defining the search strategy that has to be applied in the second step, the implementation of the Model Based Algorithm for Evolutionary Design of Experiments. When these two goals are accomplished, we will concentrate on the generalization of the strategy.

References

[1] M. Dorigo and T. Stützle. *Ant Colony Optimization*. MIT press, 2004.

[2] M. Forlin, I. Poli, D. De March, N. Packard, G. Gazzola, and R. Serra. Evolutionary experiments for self-assembling amphiphilic systems. *Chemometrics and Intelligent Laboratory Systems*, 90(2):153–160, 2008.

[3] H. Hoos and T. Stützle. *Stochastic Local Search. Fundations and Applications*. Elsevier, 2005.

[4] L. J. McGuffin, K. Bryson, and D. T. Jones. The psipred protein structure prediction server. *Bioinformatics*, 16:404–405, 2000.

[5] G. Minervini, G. Evangelista, L. Villanova, D. Slanzi, D. De Lucrezia, I. Poli, P. L. Luisi, and F. Polticelli. Massive non natural proteins structure prediction using grid technologies. *Bioinformatics, to appear*, 2009.

[6] D. C. Montogomery. *Design and Analysis of Experiments*. Wiley and Son, New York, 2005.

[7] C. Teixeira, J. A. Covas, T. Stützle, and A. G. Cunha. Application of pareto local search and multi-obejctive ant colony algorithms to the optimization of co-rotating twin screw extruders. In *Proc. of the EU/Meeting, Porto, Portugal*, pages 115–120, 29 - 30 April, 2009.

[8] F. Tekaira, e. Yeramian, and B. Dujon. Amino acids composition of genomes, lifestyle of organisms and evolutionary trends: a global picture with correspondence analysis. *Gene*, 297:51–60, 2002.

[9] M. Theis, G. Gazzola, M. Forlin, I. Poli, M. Hanczyc, and M. A. Bedau. Optimal formulation of complex chemical systems with a genetic algorithm. In *Proc. of the European Conference on Complex System, Oxford*, 25 - 29 September, 2006.

A Study of Pheromone Modification Strategies for using ACO on the Dynamic Vehicle Routing Problem

Sabrina M. de Oliveira
IRIDIA, CoDE, Université Libre de Bruxelles, Brussels, Belgium
sabrina.oliveira@ulb.ac.be

Abstract

When solving instances of the Vehicle Routing Problem (VRP), it is almost always assumed that one has access to all the information relative to the problem instance in advance and that it does not change over time. However, in most real-world vehicle routing problems information is not always static or known a priori: new costumers, new orders or unexpected variations, such as a vehicle's travel time, can occur during or after planning the initial routes. The Dynamic Vehicle Routing Problem (DVRP) is a special VRP class in which it is explicitly considered that information can be obtained and processed in real time. Despite all the progress achieved so far, one issue remains open: how to design optimization methods that are able to adapt to the constantly changing optima? This work is a glance at how the DVRP has been tackled with Ant Colony Optimization (ACO) algorithms and how some pheromone update strategies can improve the performance of ACO algorithms to solve vehicle routing problems in real-time.

1 Introduction

The consolidation of a global economy and the increasing development of information technology have made remarkable changes in the Supply Chain Management process. Companies need now to deal with information in real time to be competitive in the global market.

In this context, goods distribution is a prime example of how the use of real-time information can differentiate one company from another by its ability to react efficiently to failures or other type of disturbances.

Distribution problems in real-time are usually modeled as instances of the DVRP [7], an NP-hard combinatorial optimization problem that has been formulated in order to improve vehicle routing decisions under uncertainties encountered in the real world.

In recent years, many different versions of dynamic problems have been studied and formulated [7]. Despite all the progress achieved so far, there is still need for algorithms that are able to find and track high quality solutions

in dynamic environments. The successful performance of ACO algorithms when applied to the static vehicle routing problem [2] inspired its application on the DVRP. In this paper, we briefly describe how the DVRP has been tackled with ACO algorithms. We focus in particular on methods that modify the pheromone update process in order to make the algorithms more suitable to the dynamic nature of this class of problems.

2 Dynamic Vehicle Routing Problems

A vehicle routing problem can be considered dynamic if any important information that defines a problem instance changes over time. On the contrary, if all information is received or known before planning starts and does not change thereafter, it is considered to be a static problem [7]. Thus, we can say that every DVRP is a series of static VRPs if we consider that a change in one parameter leads to another instance. As an example, consider a set of customers to be served by a vehicle. Some services are known a priori and an initial set of routes is built covering these services. However, during the day new service orders arrive and are included in the current routes. This gives rise to a new problem instance.

For solving dynamic routing problems two main strategies are found in the literature, both solving a series of static problems: (i) a new problem is defined each time a new event occurs [1], or (ii) a new problem is defined at predefined time intervals [6].

3 Ant Colony Optimization for the Dynamic Vehicle Routing Problem

In a practical scenario, the best of the two above mentioned approaches depends on the time that is available for solving the new problem and the amount of information that changes from one instance to the next.

The simplest way to handle dynamic changes is to restart the optimization algorithm and solve a new static VRP. However, if the changes to the current instance are small, it should be a good strategy to transfer knowledge from the previous optimization run to the new one. On the other hand, restarting the optimization process from scratch could prevent the algorithm from getting stuck in a local optimum, though at the cost of more computation time.

Based on the above mentioned constraints, pheromone modification strategies have started to be proposed to make ACO a successful tool for optimization in dynamic environments. For a complete review about ACO see [2].

3.1 Pheromone Modification Strategies

The research interest about pheromone modification strategies for dynamic problems was started by Guntsh and Middendorf [4] [5] who tried different strategies for decreasing the pheromone values on the pheromone update procedure in order to improve the performance of Ant Colony System [2] on instances of the dynamic travelling salesman problem. Global pheromone modification strategies were proposed as a way to reduce the influence of past decisions when building a new solution after the insertion or deletion of a city into an instance of the problem tackled. These strategies are:

Restart-Strategy assigns a reset-value at each city j of the problem by equalizing the pheromone values to some degree,

η-Strategy and **τ-Strategy** assume that good solutions to changed instances will differ only locally. In both strategies, the pheromone equalization value is proportional to the distances between the cities, that is, pheromone values is maintained to a higher degree on edges that are "closer" to inserted/deleted cities in the current instance. τ-Strategy uses pheromone based information as reference, while η-Strategy uses the heuristic information.

All the strategies presented above work by distributing a value $Y_i \in [0,1]$ to each city i. These values are then used to reinitialize the pheromone values τ_{ij} on edges incident to i according to the equation

$$\tau_{ij} \rightarrow (1 - Y_i) \cdot \tau_{ij} + Y_i \cdot \tau_0, \tag{1}$$

where τ_0 is the initial pheromone value.

The experiments using the proposed pheromone strategies have shown that when applied to the dynamic TSP, the variation of pheromone update values have a direct and positive influence to problem solutions. After parameter tuning all these pheromone update strategies have achieved better results than restarting the algorithm for each static problem.

Inspired by Guntsh and Middendorf [4] [5], Montemanni [6] implemented a simpler pheromone modification strategy for a vehicle routing problem with dynamic demands where a parameter γ_r replaces the parameter ρ on the global pheromone update procedure [2]. For each pair of customers which appear both in the previous and in the current static problem, the pheromone matrix entry is re-initialized using γ_r. Entries of pheromone matrix corresponding to pairs of new customers are initialized by the starting pheromone value. Entries of the new pheromone matrix corresponding to pairs of new customers are initialized to τ_0.

$$\tau_{ij} = (1 - \gamma_r) \cdot \tau_{ij}^{old} + \gamma_r \cdot \tau_0, \tag{2}$$

where τ_{ij}^{old} represents the value of τ_{ij} in the previous static problem. Thus, the pheromone values are not completely reinitialized.

In [6] different values for γ_r were proposed. As in [4] [5], the obtained results also highlighted the importance of the pheromone conservation procedure in the algorithm for building good solutions.

Other pheromone strategies for dynamic environments can be found in the literature. So far, they are mostly applied to routing in telecommunications networks. When applied to vehicle routing the TSP is chosen for application. Eyckelhof [3] for example, presented several ways to adapt the pheromone matrix both locally and globally in an Ant System algorithm applied for the dynamic TSP where the travel time between the cities was dynamic.

4 Conclusions

We briefly described approaches for tackling the dynamic vehicle routing problem using Ant Colony Optimization algorithms. We paid special attention to methods that reuse information from previously found solutions as it has been reported that such an approach gives better results than restarting the algorithms.

Despite these promising results it is still unclear how new pheromone update strategies affect the runtime behavior of the algorithms as well as their convergence properties. Moreover, there is so far no methodological approach for determining the right balance between reusing information from one instance to another and restarting algorithms anew. Another avenue of research is the hybridization of ACO algorithms with other approaches for the DVRP.

References

[1] L. Bianchi. Notes on dynamic vehice routing problem. state of the art. Technical Report 05–01, IDISIA, 2000.

[2] M. Dorigo and T. Stützle. *Ant Colony Optimization.* MIT Press, Cambridge, MA, 2004.

[3] C. Eyckelhof and M. Snoek. Ant systems for a dynamic tsp - ants caught in a traffic jam. In M. Dorigo, G. D. Caro, and M. Samples, editors, *Ant Algorithms - Third International Workshop, ANTS 2002*, pages 88–99. Springer Verlag, Berlin, Germany, 2002.

[4] M. Guntsch and M. Middendorf. Pheromone modification strategies for ant algorithms applied to dynamic TSP. In E. J. W. Boers, J. Gottlieb, P. L. Lanzi, R. E. Smith, S. Cagnoni, E. Hart, G. R. Raidl, and H. Tijink, editors, *Applications of Evolutionary Computing: Proceedings of*

EvoWorkshops 2001, volume 037 of *lncs*, pages 213–222. Springer Verlag, Berlin, Germany, 2001.

[5] M. Guntsch, M. Middendorf, and H. Schmeck. An ant colony optimization approach to dynamic TSP. In L. S. et al., editor, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 860–867. Morgan Kaufmann Publishers, San Francisco, CA, 2001.

[6] R. Montemanni, L. M. Gambardella, A. E. Rizzoli, and A. V. Donati. Ant colony system for a dynamic vehicle routing problem. *Journal of Combinatorial Optimization*, 10(4):327–343, 2005.

[7] H. N. Psaraftis. Dynamic vehicle routing: status and prospects. *Annals of operations research*, 61(1):143–164, 1995.

Adaptive Operator Selection in EAs with Extreme - Dynamic Multi-Armed Bandits

Álvaro Fialho and Marc Schoenauer
Microsoft Research – INRIA Joint Centre, Orsay, France
FirstName.LastName@inria.fr

Abstract

The performance of evolutionary algorithms is highly affected by the selection of the variation operators to solve the problem at hand. This paper presents a brief review of the results that have been recently obtained using the "Extreme - Dynamic Multi-Armed Bandit" (Ex-DMAB), a technique used to automatically select the operator to be applied between the available ones, while searching for the solution. Experiments on three well-known unimodal artificial problems of the EC community, namely the OneMax, the Long k-Path and the Royal Road, and on a set of a SAT instances, are briefly presented, demonstrating some improvements over both any choice of a single-operator alone, and the naive uniform choice of one operator at each application.

1 Adaptive Operator Selection

Adaptive methods use information from the history of evolution to modify parameters while solving the problem. This paper focuses on the *Adaptive Operator Selection* (AOS), i.e., the definition of an on-line strategy able to autonomously select between different variation operators each time one needs to be applied. Fig. 1 illustrates the general scheme for achieving this goal, from which we can derive the need of defining two main components: the *Credit Assignment* - how to assess the performance of each operator based on the impact of its application on the progress of the search; and the *Operator Selection* rule - how to select between the different operators based on the rewards that they have received so far.

2 Extreme - Dynamic Multi-Armed Bandit

The two ingredients of the *Adaptive Operator Selection* method proposed by us are: an *Operator Selection* rule based on the Multi-Armed Bandit paradigm, and a *Credit Assignment* mechanism based on extreme values.

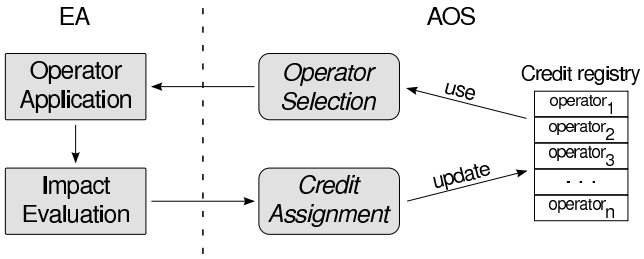

Figure 1: General *Adaptive Operator Selection* scheme.

2.1 Operator Selection: Dynamic Multi-Armed Bandits

The explored idea, firstly proposed in [3], is that the selection of an operator can be seen as yet another Exploration vs. Exploitation dilemma, but this time at operator-selection level: there is the need of applying as much as possible the operator known to have brought the best results so far, while nevertheless exploring the other possibilities, in case one of the other operators becomes the best option at some point. Such dilemma has been intensively studied in the context of *Game Theory*, in the so-called Multi-Armed Bandit (MAB) framework. Among the existent MAB variants, the *Upper Confidence Bound* (UCB) [1] was chosen to be used, for being proved optimal w.r.t. maximization of the cumulative reward.

More formally, the UCB algorithm works as follows. Each variation operator is viewed as an *arm* of a MAB problem. Let $n_{i,t}$ denote the number of times the i^{th} arm has been played up to time t, and let $\hat{p}_{i,t}$ denote the average empirical reward received until time t by arm i. At each time step t, the algorithm selects the arm maximizing the following quantity:

$$\hat{p}_{j,t} + C\sqrt{\frac{2\log\sum_k n_{k,t}}{n_{j,t}}} \tag{1}$$

The first term of this equation favors the best empirical arm (exploitation) while the second term ensures that each arm is selected infinitely often (exploration); this algorithm has also been described briefly as "be optimistic when facing the unknown", as the second term of Equation 1 can also be viewed as some kind of variance, and the user should choose the arm that might lead to the highest value.

In the original setting [1], all rewards, and hence also their empirical means $\hat{p}_{j,t}$ are in $[0, 1]$. However, since this is not the case in the AOS context, a *Scaling factor* C is needed, in order to properly balance the trade-off between both terms.

Another important issue is that the original MAB setting is static, while the AOS scenario is dynamic, i.e., the quality of the operators is likely to change along the different stages of the search. Even though the exploration

term in the UCB algorithm ensures that all operators will be tried infinitely many times, after a change in their ordering, it might take a long time before the new best operator catches up. To cope with dynamic environments, it was proposed [7] to use a statistical test that efficiently detects changes in time series, the *Page-Hinkley* (PH) test [10], coupled with the UCB algorithm. Basically, as soon as this test detects a change in the rewards distribution, the MAB algorithm is restarted from scratch, allowing it to quickly re-discover the new best operator.

2.2 Credit Assignment: Extreme Value Based

The idea of using extreme values was proposed as the *Credit Assignment* mechanism, based on the assumption that attention should be paid to extreme, rather than average events, in agreement with [11]. The credit assigned to the operator is the maximum of the impacts caused by the operator application over a sliding window of the last $\mathcal{W}$ applications.

The measurement of such impact depends on the nature of the problem at hand. In unimodal problems, we have been using the fitness improvement; while in the multimodal SAT problems, an engineered aggregation of fitness improvement and diversity, called Compass [9], was applied.

3 Summary of Results

Experiments with *Ex-DMAB* in unimodal benchmark problems have been presented in [4, 5, 6], in which the fitness improvement was used to measure the impact of the operator application. The *Ex-DMAB* has been used to adapt a (1+λ)-EA, by efficiently choosing on-line between 4 mutation operators for solving the OneMax problem [3]; and has been tried on yet another unimodal benchmark problem, the Long k-Path [5], this time efficiently selecting between 5 mutation operators. In both cases, the optimal operator selection strategy was extracted by means of Monte-Carlo simulations, and the Ex-DMAB showed to perform statistically equivalent to it; while significantly improving over the naive (uniform selection) approach. It has also been used to adapt a (100,100)-EA with 4 crossover and 1 mutation operators on the Royal Road problem [6], also performing significantly better than the naive approach. For the three problems, we have also used other AOS combinations as baseline for comparison, namely Adaptive Pursuit, Probability Matching and the static MAB (without restarts), coupled with Extreme or Average rewards. Ex-DMAB was shown to be the best option.

A different analysis was also done in the light of SAT problems, in [8]. Since these problems are mostly multimodal, the reward used was the Compass [9], which aggregates both the fitness improvement of the offspring, and the diversity that this offspring brought by being inserted in the population. Significantly better results were achieved w.r.t. the naive approach, and also

to the original Compass and Ex-DMAB combinations. The application of such approach in multimodal functions should be more deeply investigated in the future.

4 Discussion and Perspectives

A current drawback concerns the tuning of the *Ex-DMAB* hyper-parameters, the window size $\mathcal{W}$, the scaling factor C and the change detection test threshold γ – actually, they are being off-line tuned by means of F-Race [2]. Although its good performances rely on such expensive procedure, *Ex-DMAB* was found to outperform the main options opened to the naive EA user, namely (i) using a fixed or deterministic strategy (including the naive, uniform selection, strategy; or (ii) using a different AOS strategy. Furthermore, *Ex-DMAB* involves a fixed and limited number of parameters, whereas the number of operator rates increases with the number of operators.

Further research will aim at addressing the above weaknesses. Firstly, we shall investigate better how the threshold γ and the scaling factor C relate, as both cooperate to control the exploration vs exploitation trade-off. Another possible direction is the analysis of a rank-based reward, instead of the absolute value that is currently being used; in this way, it will not be problem-dependent anymore, and a robust setting for these parameters might be found. A different direction that might also be analyzed is the use of this selection technique in another context, selecting between different algorithms instead of operators.

References

[1] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2/3):235–256, 2002.

[2] M. Birattari, T. Stützle, L. Paquete, and K. Varrentrapp. A racing algorithm for configuring metaheuristics. In *Proc. GECCO'02*, pages 11–18. Morgan Kaufmann, 2002.

[3] L. Da Costa, A. Fialho, M. Schoenauer, and M. Sebag. Adaptive operator selection with dynamic multi-armed bandits. In *Proc. GECCO'08*, pages 913–920. ACM, 2008.

[4] A. Fialho, L. Da Costa, M. Schoenauer, and M. Sebag. Extreme value based adaptive operator selection. In *Proc. PPSN'08*, pages 175–184. Springer, 2008.

[5] A. Fialho, L. Da Costa, M. Schoenauer, and M. Sebag. Dynamic multi-armed bandits and extreme value-based rewards for adaptive operator

selection in evolutionary algorithms. In *Proc. LION'09*. Springer, 2009 (to appear).

[6] A. Fialho, M. Schoenauer, and M. Sebag. Analysis of adaptive operator selection techniques on the royal road and long k-path problems. In *Proc. GECCO'09*, pages 779–786. ACM, 2009.

[7] C. Hartland, N. Baskiotis, S. Gelly, O. Teytaud, and M. Sebag. Change point detection and meta-bandits for online learning in dynamic environments. In *Proc. CAp'07*, pages 237–250. Cepadues, 2007.

[8] J. Maturana, A. Fialho, F. Saubion, M. Schoenauer, and M. Sebag. Extreme compass and dynamic multi-armed bandits for adaptive operator selection. In *Proc. CEC'09*, pages 365–372. IEEE, 2009.

[9] J. Maturana and F. Saubion. A compass to guide genetic algorithms. In *Proc. PPSN'08*, pages 256–265. Springer, 2008.

[10] E. Page. Continuous inspection schemes. *Biometrika*, 41:100–115, 1954.

[11] J. M. Whitacre, T. Q. Pham, and R. A. Sarker. Use of statistical outlier detection method in adaptive evolutionary algorithms. In *Proc. GECCO'06*, pages 1345–1352. ACM, 2006.

An Automatically Configured Modular Algorithm for Post Enrollment Course Timetabling

Chris Fawcett, Holger H. Hoos
Computer Science Department,
University of British Columbia, Vancouver, BC, Canada
{fawcettc,hoos}@cs.ubc.ca

Marco Chiarandini
Department of Mathematics and Computer Science,
University of Southern Denmark, Odense, Denmark
marco@imada.sdu.dk

Abstract

Timetabling tasks form a widely studied type of resource scheduling problem, with important real-world applications in schools, universities and other educational settings. In this work, we focus on post-enrollment course timetabling, the problem that was covered by Track 2 of the recent 2nd International Timetabling Competition (ITC2007). Following an approach that makes strong use of automated exploration of a large design space of modular and highly parameterised stochastic local search algorithms for this problem, we have obtained a solver that achieves consistently better performance than the top-ranked solver from the competition. This represents a substantial improvement in the state of the art for post-enrollment course timetabling.

1 Introduction

Course and examination timetabling is a resource-constrained scheduling problem encountered by universities and other educational institutions, involving scheduling a set of events into given rooms and timeslots. The resulting schedule is subject to resource and feasibility constraints derived from the availability of rooms, student enrollments in the events, precedence relations between events, and student or teacher preferences. While the feasibility constraints must be strictly satisfied, giving rise to a satisfaction problem, preferences should not be violated whenever possible, which gives rise to an optimization problem. The presence of such hard and soft constraints is typical for many real-world constraint optimisation problems.

In this work, we present a new state-of-the-art solver for a particular problem in timetabling, the Post-Enrollment Course Timetabling Problem,

as considered in Track 2 of the recent 2nd International Timetabling Competition (ITC2007) [13]. Leveraging our previous work which resulted in a solver that placed third in Track 2 of ITC2007 [5], our main contribution lies in the automated approach for designing our new solver, as well as the resulting solver itself, which represents a substantial improvement over the previous best algorithm for the problem (the solver by Cambazard et al. that won Track 2 of ITC2007 [3]).

2 Automated Design Approach and Algorithm Framework

In recent years there has been a considerable amount of methodological research devoted to the issue of configuring the components and tuning the parameters of optimisation algorithms, and especially of heuristic algorithms. Contrary to the traditional approach of trying to minimise the number of user-configurable algorithm parameters, these methods embrace the idea of parameterising as much algorithm functionality as possible [8]. Out of the available procedures for automated algorithm configuration we selected FocusedILS [11, 10], as it is the only procedure we are aware of that has been demonstrated to be effective in dealing with very large, highly discrete design spaces.

Our approach is heavily based on the use of this powerful automated algorithm configuration method, allowing us to search for a performance-optimised design within a very large space of candidate solvers. The same fundamental automated algorithm design approach has been recently used to obtain substantial improvements in the state of the art for solving various types of SAT instances [9, 12] (where the latter piece of work has been undertaken in parallel and to a large degree independently of the work presented here). The space of algorithms for the problem considered in this work is defined by a modular solver framework that is based on stochastic local search (SLS) methods [7] and builds on several ideas from the timetabling and graph coloring literature, as well as on work done for the first timetabling competition in 2003 [4]. The design space for our solvers involves not only traditional numerical parameters, but also choices between preprocessing options and neighbourhoods as well as the diversification strategies employed. This framework contains 18 configuration parameters and design choices, and allows for a total of approximately 10^{13} possible instantiations. The key idea behind this framework is to first solve the feasibility problem and then the optimisation problem by restricting the search to only feasible timetables. Different solution representations and neighbourhoods are interleaved during the search in the two phases, and randomisation together with tabu search and simulated annealing concepts are combined in a flexible and novel way.

The hard constraint solving phase of our configured solver consists of a constructive phase followed by a stochastic local search phase. The constructive phase generates partial assignments using an approach similar to that of Arntzen and Løkketangen [1], inserting events using a topological ordering of the precedence constraints. The local search phase uses a tabu search procedure based on the PARTIALCOL algorithm [2]. At each iteration, an unscheduled event is inserted into the best possible non-tabu timeslot for it, and all events subsequently breaking hard constraints are moved into the list of unscheduled events. After a number of non-improving iterations have been made, a component of the soft constraint solver is used as a diversification mechanism. If this perturbation and subsequent optimisation fails to produce an improvement, a number of events are chosen at random to be unscheduled and the search proceeds as before. When there are no longer any hard constraint violations, the algorithm proceeds to the soft constraint solving phase.

The soft constraint solving phase begins with a first-improvement local search until a local minima is found for all four of the neighbourhoods available in the solver. Next, a simulated annealing phase is applied using both the 2-exchange and swap-of-time-slots neighbourhoods until a specified time limit is reached or an optimal schedule is located. In addition to the usual geometric cooling schedule, the temperature parameter of the annealing procedure is increased after a number of non-improving iterations.

3 Experimental Design and Results

Based on the multi-phase architecture underlying our solver, we applied FocusedILS to first optimise the parameters controlling the behaviour of the hard constraint solver and then to optimise the parameters for the soft constraint solver. The solver was tuned using a runtime cutoff of 600 CPU seconds, and all tuning was performed using a cluster of identical machines, in order to perform multiple independent runs of FocusedILS in parallel. Overall, 360 CPU hours were used for runs of FocusedILS to produce our final solver.

In the course of this work, we also developed a new metric for FocusedILS called the *p-value performance metric*, based on the idea of using empirical solution quality distributions (SQDs) for an existing solver as a target for the algorithm being tuned. This metric was used when tuning the parameters of the soft constraint solver, using empirical SQDs from the Cambazard et al. solver. In addition, we extended FocusedILS to perform more effectively when there is an expectation that some numeric parameters have a convex or unimodal response when the values of the other parameters are fixed. This new version 2.4 of FocusedILS was used for all parameter configuration performed in this work.

Of the 24 available instances from ITC2007 for this problem, the sixteen "public" instances were used for configuration while the eight "private" instances were reserved for testing purposes. Using 100 runs each 600 CPU seconds in length on each of these 24 instances, our tuned solver configuration achieves better median soft constraint violation values than the solver of Cambazard et al. on all of the public instances as well as for five of the eight private instances that were not used in the configuration process. On four instances, the median quality is better by more than an order of magnitude. In addition, we also beat the solver of Cambazard et al. using the rank-based competition metric of ITC2007 in a 2-way race. These results clearly demonstrate a substantial performance improvement compared to the previous state of the art.

In future work, it would be particularly interesting to use these tools in combination with an appropriately expanded version of our solver framework to tackle the two other timetabling problems used in ITC2007 (an examination and a curriculum-based timetabling problem [14, 6]). We have also recently begun applying our timetabling algorithm to the real-world problem of scheduling exams at the University of British Columbia.

Acknowledgements. The authors would like to thank Hadrien Cambazard for providing source code for the solver presented in [3], as well as valuable feedback via e-mail.

References

[1] H. Arntzen and A. Løkketangen. A tabu search heuristic for a university timetabling problem. In *Proceedings of the Fifth Metaheuristics International Conference*, Kyoto, Japan, August 2003.

[2] I. Blöchliger and N. Zufferey. A graph coloring heuristic using partial solutions and a reactive tabu scheme. *Computers & Operations Research*, 35(3):960–975, 2008.

[3] H. Cambazard, E. Hebrard, B. O'Sullivan, and A. Papadopoulos. Local search and constraint programming for the post enrolment-based course timetabling problem. In E. K. Burke and M. Gendreau, editors, *Proceedings of the 7th International Conference on the Practice and Theory of Automated Timetabling (PATAT 2008)*, Montreal, CA, 2008. Université de Montréal.

[4] M. Chiarandini, M. Birattari, K. Socha, and O. Rossi-Doria. An effective hybrid algorithm for university course timetabling. *Journal of Scheduling*, 9(5):403–432, Oct. 2006.

[5] M. Chiarandini, C. Fawcett, and H. H. Hoos. A modular multiphase heuristic solver for post enrollment course timetabling (extended ab-

stract). In *Proceedings of the 7th International Conference on the Practice and Theory of Automated Timetabling (PATAT 2008)*, 2008.

[6] L. D. Gaspero, B. McCollum, and A. Schaerf. The second international timetabling competition (itc-2007): Curriculum-based course timetabling (track 3). Technical Report QUB/IEEE/Tech/ITC2007/CurriculumCTT/v1.0, Queen's University, Belfast, United Kingdom, 2007.

[7] H. Hoos and T. Stützle. *Stochastic Local Search: Foundations and Applications*. Morgan Kaufmann Publishers, San Francisco, CA, USA, 2004.

[8] H. H. Hoos. Computer-aided design of high-performance algorithms. Technical Report TR-2008-16, University of British Columbia, Department of Computer Science, 2008.

[9] F. Hutter, D. Babić, H. H. Hoos, and A. J. Hu. Boosting Verification by Automatic Tuning of Decision Procedures. In *Proceedings of Formal Methods in Computer Aided Design (FMCAD'07)*, pages 27–34, Washington, DC, USA, 2007. IEEE Computer Society.

[10] F. Hutter, H. Hoos, K. Leyton-Brown, and T. Stützle. ParamILS: An automatic algorithm configuration framework. Technical Report TR-2009-01, University of British Columbia, Department of Computer Science, January 2009.

[11] F. Hutter, H. H. Hoos, and T. Stützle. Automatic algorithm configuration based on local search. In *Proc. of the Twenty-Second Conference on Artificial Intelligence (AAAI '07)*, pages 1152–1157, 2007.

[12] A. KhudaBukhsh, L. Xu, H. H. Hoos, and K. Leyton-Brown. SATenstein: Automatically building local search SAT solvers from components. In *Proc. of the Twenty-First International Joint Conference on Artificial Intelligence (IJCAI-09)*, pages 517–524, 2009.

[13] R. Lewis, B. Paechter, and B. McCollum. Post enrolment based course timetabling: A description of the problem model used for track two of the second international timetabling competition. Cardiff Working Papers in Accounting and Finance A2007-3, Cardiff Business School, Cardiff University, August 2007.

[14] B. McCollum, P. McMullan, E. K. Burke, A. J. Parkes, and R. Qu. The second International Timetabling Competition: Examination timetabling track. Technical Report QUB/IEEE/Tech/ITC2007/-Exam/v4.0/17, Queen's University, Belfast, Sep 2007.

Combining F-Race and Mesh Adaptive Direct Search for Automatic Algorithm Configuration

Zhi Yuan, Thomas Stützle and Mauro Birattari
IRIDIA, CoDE, Université Libre de Bruxelles, Brussels, Belgium
{zyuan,stuetzle,mbiro}@ulb.ac.be

Abstract

In this article, we tackle the automatic algorithm configuration problem (ACP), finding the best configuration of an algorithm such that some measure of its performance is optimized. We study the mesh adaptive direct search (MADS) method for the ACP. MADS is an iterative algorithm for global optimization. It does not require any infomation of the evaluation function, therefore the ACP can be regarded as a black-box for evaluation. To handle the stochastic nature of the ACP, we adopted F-Race, to adaptively allocate the evaluation budgets among a population of candidate configurations. We compare the hybrid of MADS and F-Race (MADS/F-Race) to MADS with certain fixed numbers of evaluations, and demonstrate the advantage and robustness of MADS/F-Race over its counterparts in solving the ACP.

1 Introduction

Many algorithms for tackling computationally hard problems have a number of parameters to be determined, which might greatly influence the algorithm performance. Finding a good parameter setting of an algorithm is of great importance in both research and industrial practice, and we call this problem *algorithm configuration problem*.

In this abstract, we focus on the *offline* algorithm configuration problem following the definition from Birattari [3]. In the *training phase*, an algorithm configuration is determined such that the expected solution quality of the algorithm is optimized. The selected configuration is then deployed in the *production phase* where the algorithm is used to solve previously unseen instances. One crucial aspect is generalization, i.e., based on a given set of training instances, the goal is to find high-performing algorithm configurations for (a potentially infinite set of) unseen instances.

The algorithm configuration problem is itself a hard optimization problem. An important aspect of this problem is that it is typically a stochastic optimization problem. There are two main sources of stochasticity, firstly

the algorithm itself is usually stochastic, which is typical for stochastic local search (SLS) algorithms. However, even if the algorithm is deterministic, its performance and search behavior depends on the particular instance to which it is applied. In fact, the particular instance being tackled can be seen as having been drawn according to some underlying, possibly unknown probability distribution, introducing in this way a second stochastic factor.

In the effort to tackle this problem, `F-Race` was proposed [4] and it is particularly well suited for dealing with the stochastic aspect. The essential idea of `F-Race` is to evaluate a given set of candidate configurations iteratively on a stream of instances. As soon as enough statistical evidence is gathered against some candidate configurations, these are eliminated and the race continues with the surviving ones until only one candidate is left. In particular, in each evaluation step, the non-parametric family-wise Friedman test is applied to check whether there is evidence that at least one of the configurations is significantly different from the others. If the null hypothesis of no differences is rejected, Friedman post-tests are applied to eliminate those candidate configurations that are significantly worse than the best one.

There are various ways how the candidate configurations for `F-Race` are generated. In this article, we have adopted an iterative sampling method called mesh adaptive direct search (MADS) [1] for tuning numerical parameters. We show how the hybrid of `F-Race` and MADS, which is denoted `MADS/F-Race`, can help to cope with the stochastic nature of the evaluation function in the context of algorithm configuration problems.

2 The MADS/F-Race

The `MADS` class of algorithms [1] is designed for global optimization problems, which can be in general stated as $\min_{p\in\Omega} f(p)$ for a function $f : \Omega \in \mathbb{R}^d \rightarrow \mathbb{R} \cup +\infty$, and $d = |\Omega|$ denotes the dimension of the variable space. The algorithm does not require derivative information or continuity of f, thus f can be treated as a black-box. For the settings of the `MADS` algorithm, we follow [2].

`MADS` is an iterative algorithm, where each iteration essentially consists of two steps, the `search` step and the `poll` step. In the initial `search` step, d^2 trial points are sampled by latin hypercube, while in the `search` step from the second iteration on, $2d$ trial points are randomly sampled on the *current mesh*, whose coarseness is controlled by a *mesh size parameter* $Delta_k \in \mathbb{R}_+$. The set of mesh points at iteration k is defined as:

$$M_k = \bigcup_{p\in S_k} \{p + \Delta_k z : z \in \mathbb{Z}^d\} \tag{1}$$

where S_k denotes the set of points that have been evaluated in the previous

iterations. If the `search` step does not find an improved point, the `poll` step is activated, where $2d$ trial points are generated, which forms the set called *frame*, defined as:

$$F_k = \{p_k \pm \Delta_k b : b \in B_k\} \tag{2}$$

where $B_k = \{b^1, b^2, \ldots, b^d\}$ is a basis in $\mathbb{Z}^d$. At iteration k, either an improved point is found replacing the incumbent or both steps terminate, the mesh size parameter is updated as Equation 3:

$$\Delta_{k+1} = \begin{cases} \Delta_k/4 & \text{if no improved mesh point is found;} \\ 4\Delta_k & \text{if an improved mesh point is found, and } \Delta_k \leq \frac{1}{4}; \\ \Delta_k & \text{otherwise.} \end{cases} \tag{3}$$

In order to handle the noisy black-box objective function, `MADS` should be adapted by allowing more than one evaluation on the trial points so as to reduce the evaluation variance. A brute-force approach is to perform a fixed number of evaluations. However, it is not a priori known what the appropriate evaluation number is, besides, the same amount of computational resources have to be allocated to the good performing candidates as well as the bad performing ones. To this end, a hybrid of `F-Race` and `MADS` (or `MADS/F-Race`), is proposed to cope with this difficulty. The hybrid can be done in a rather straightforward way. At each iteration of `MADS`, be it the `search` or `poll` step, a population of candidate configurations sampled from `MADS`, together with the incumbent point, will be evaluated by `F-Race`. Each `F-Race` is allowed a budget of 10 times the number of candidates. The objective of `F-Race` in this task is to identify the best point, be it the incumbent or a new improved point, and then the `MADS` parameters are updated accordingly.

3 Experimental results

The experiments are conducted on six problem classes, (i) MMASTSP, the $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System ($\mathcal{MM}$AS) for traveling salesman problem (TSP) with 6 parameters; (ii) MMASTSP_ls, the $\mathcal{MM}$AS with 2-opt local search for TSP, the same parameters as MMASTSP; (iii) ACSTSP, ant colony system (ACS) for TSP with 6 parameters; (iv) ACSTSP_ls, ACS with 2-opt local search fo TSP, the same parameters as ACSTSP; (v) ILSQAP, iterated local search (ILS) for quadratic assignment problem (QAP) with 8 parameters; (vi) SAQAP, simulated annealing (SA) for QAP with 3 parameters. Each class contains 4 different computation time (1, 2, 5, 10 seconds) and 3 different numbers of function evaluations (1000, 2000 and 4000), which makes it in total 12 *domains* per problem class.

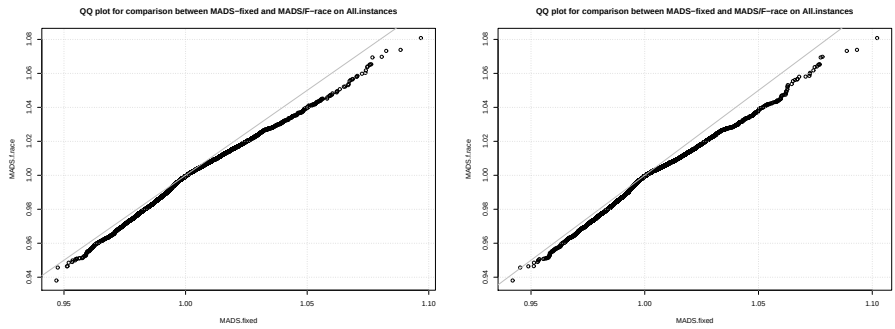

Figure 1: The comparison of normalized costs between random (left) or tuned (right) MADS-fixed and MADS/F-race. In average `MADS/F-Race` is 0.25% better than random `MADS(fixed)`, and 0.17% better than tuned `MADS(fixed)`

In order to apply `MADS(fixed)`, six levels of the evaluation number are pre-assigned: 1, 5, 10, 20, 30, 40. Given no a priori information about the optimal setting of `MADS(fixed)`, we select one setting from the six randomly for each problem domain, and compare it with `MADS/F-Race` across all domains, with blocking on each instance. The comparison is done using the Wilcoxon signed rank test with continuity correction; the α-level chosen is 0.05. The experimental results show that `MADS/F-Race` performs statistically significantly better than `MADS(fixed)` with a randomly selected number of evaluations on each individual domain. The QQplot is shown in Figure 1.

In the sequel, we fine-tune the evaluation number of `MADS(fixed)` by leave-one-out cross-validation. It is done as follows, in each iteration, one domain is picked for testing, and the rest of the data set serve as the training set. The best candidate chosen according to the training set will be performed on the testing domain, and its results are collected into a validation set. The process repeats until each domain has collected its validation data. Then we compare the validation set of `MADS(fixed)` to the target algorithm `MADS/F-Race`. Note that the data collected in the validation set are first trained, while `MADS/F-Race` is not. The unfairness of the comparison shows the robustness of `MADS/F-Race`.

The comparison results show that MADS/F-race significantly (by wilcoxon test) outperforms MADS-fixed with tuned evaluation numbers. The QQplot is shown in Figure 1. Also in the individual problem classes, `MADS/F-Race` obtains significantly better results than tuned `MADS(fixed)` in MMASTSP, ACSTSP, SAQAP, whose variation coefficients are relatively higher than the other three problem classes, on which `MADS/F-Race` has significantly worse results than tuned `MADS(fixed)`. The positive correlation between the problem variation and the performance of `MADS/F-Race` is an interesting topic to investigate in the future.

Acknowledgement This work has been supported by META-X, an ARC project funded

by the French Community of Belgium. The authors acknowledge support from the fund for scientific research F.R.S.-FNRS of the French Community of Belgium, of which they are research associates (MB and TS), or aspirant (ZY), respectively.

References

[1] C. Audet and J. Dennis. Mesh adaptive direct search algorithms for constrained optimization. *SIAM Journal on optimization*, 17(1):188–217, 2007.

[2] C. Audet and D. Orban. Finding Optimal Algorithmic Parameters Using Derivative-Free Optimization. *SIAM Journal on Optimization*, 17:642, 2006.

[3] M. Birattari. *Tuning Metaheuristics: A machine learning perspective*. Springer, Berlin, Germany, 2009.

[4] M. Birattari, T. Stützle, L. Paquete, and K. Varrentrapp. A racing algorithm for configuring metaheuristics. In W. B. Langdon et al., editors, *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference*, pages 11–18, San Francisco, CA, 2002. Morgan Kauffman.

Designing Screws for Polymer Compounding in Twin-Screw Extruders

Cristina Teixeira
Department of Polymer Engineering, IPC,
University of Minho, Guimarães, Portugal
cteixeira@dep.uminho.pt

Abstract

The twin screw configuration problem is a typical problem of polymer industry and consists in the definition of the location of a set of screw elements along the screw axis (sequencing problem), where several and often conflicting objectives must be optimized. An initial study of iterative improvement algorithms to solve the optimization of individual criteria was performed. Different operators, search strategies and neighborhood restrictions were tested in order to define effective algorithms. The knowledge gathered allowed the development of different multi-objective local search algorithms, such as TPLS and PLS. A Multi-Objective Ant Colony Optimization algorithm was also developed. The performance of the multi-objective algorithms was compared with previously developed Multi-Objective Evolutionary Algorithm making use of attainment functions.

1 Introduction

In the last decades, plastics have been used in applications with more rigorous specifications requiring the development of more complex systems. This is been accomplished with the use of co-rotating twin screw extruders. In fact, this type of machines are built connecting different modular screw elements allowing the definition of different configurations and making possible the adaptation to a multiplicity of systems. However, its performance is quite sensitive to the operating conditions and the screw configuration. Thus, each polymer system requires the careful definition of each of the two. The definition of the best screw configuration consists in defining the location of several elements along the screw axis maximizing the entire process. This is denoted as the Twin Screw Configuration Problem (TSCP). Only recently, an optimization methodology was developed to define the best location of a pre-defined number of screw elements [2]. In this work, alternative algorithms for tackling the TSCP were developed. The performance of the algorithms was compared with the previously designed MOEA (Reduced Pareto Set Genetic Algorithm) [2] making use of the attainment functions.

2 Modeling

As represented in Figure 1 screws of twin screw extruders are built connecting screw elements with different geometrical characteristics that induces different flow behaviors: restrictive elements (left handed and kneading blocks) induce restrictions to the polymer flow while transport elements have conveying capability. The goal consists in defining the position along the screw axis of 14 screw elements, in such a way that the process performance is maximized. To evaluate the performance of each screw solution a modeling

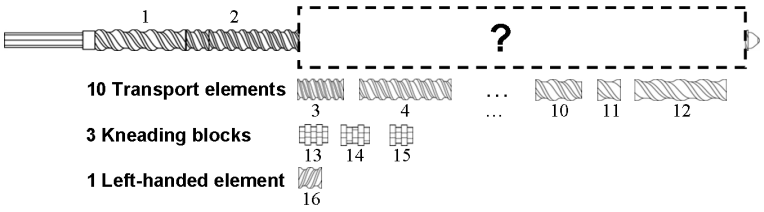

Figure 1: Twin Screw Configuration Problem

routine that takes into account the appropriate physical steps was initially developed [6]: the solid polymer is feed into a hopper and it will flow through transport elements under starved conditions. When a restrictive element is reached, the channel starts to fill up and the melting process takes place. When all polymer is melted, the flow occurs with or without pressure in the rest of the screw elements, depending on whether it is totally or partially filled; overall, pressure is determined by the location of the restrictive elements.

In this example, the specific mechanical energy (SME), viscous dissipation and average strain are the optimization goals considered. SME represents the mechanical energy required to move the screws per unit of material and viscous dissipation is the difference between real and set temperatures. Thus, both objectives should be minimized. Since strain can be related to the degree of mixing, it should be maximized.

3 Twin-Screw Optimization

Given the high computational time required by the evaluation of each solution (1-2 minutes of CPU time) it is necessary to develop algorithms that reach good results with a low number of evaluations. Iterative improvement algorithms were implemented given that they accomplish very well this task. Initially, different operators and pivoting rules were tested in the optimization of individual criterion in order to define an appropriate neighborhoord relation [3]. Given the importance of the order how the neighborhood is

checked the usual sequential order was compared with the initial shift of the restrictive elements. Finally, different neighborhood restrictions as limits of number of neighborhood scans, distance of exchanged elements in the permutation and DLB strategy were analyzed in order to decrease the number of evaluations without a significant loss of solution quality.

Since a good performance was obtained with iterative improvement algorithms, an extension of these algorithms was developed to solve the multi-objective TSCP. For that, two different strategies were studied. The first one converts multi-objective problems into single problems aggregating the several objectives into a single objective function - Two-Phase Local Search (TPLS). The other considers that one solution is accepted as a new solution if it is non-dominated - Pareto Local Search (PLS) [5]. A Multi-Objective Ant Colony Optimization algorithm was also developed and a comparison study of the different parameters, such as, use of one or various pheromone matrices, different types of assignment of screw elements, use of the probability summation rule and use of various colonies was developed [1]. A detailed investigation of the sensitivity of the algorithms performance to changes of their parameters and a comparison to the previously designed MOEA was carried out plotting the differences between the respective empirical attainment functions (EAF)[4].

4 Results and Discussion

Figure 2 represents the comparison between MOEA and TPLS for the maximization of Strain and minimization of SME. The results were obtained running the algorithms 10 times using different seed values and during 3000 evaluations. The left plot represents the region of the Pareto frontier where the EAF obtained with MOEA is higher than that obtained with TPLS. The opposite is represented on the rigth side. As can be seen, the performance of the TPLS algorithm is higher. Identical results were obtained with the PLS algorithm and for the remaining case studies.

Figure 3 shows three solutions taken from the Pareto Frontier obtained with the MO-ACO algorithm for the same situation. Restrictive elements are represented at darker colour. When the SME decreases, the restrictive elements are upstream of the screw configuration in order to melt the polymer as late as possible, and consequently, minimize the energy necessary to rotate the screws. The opposite happens when the maximization of Strain is considered. Therefore the solutions obtained are in agreement with the knowledge about the process and have physical meaning.

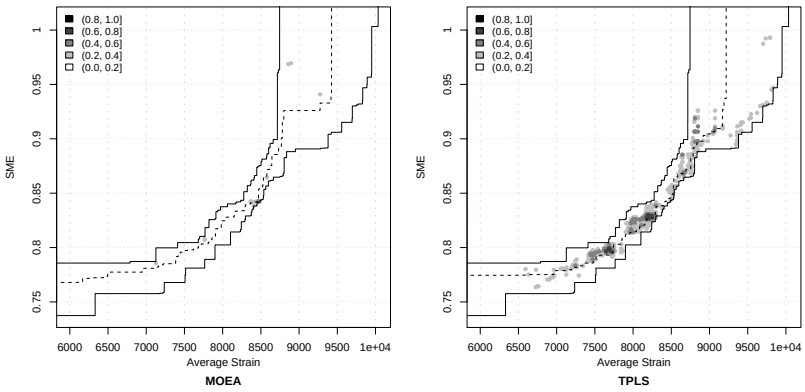

Figure 2: EAFs differences between the results obtained with MOEA and TPLS for the maximization of Strain and minimization of SME.

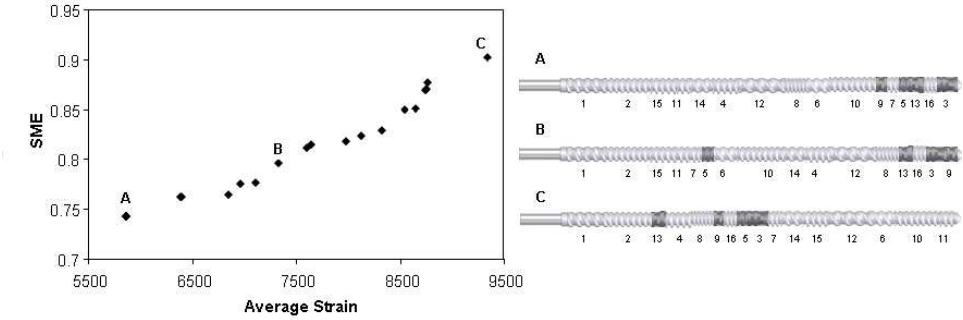

Figure 3: Pareto front for maximization of Average Strain and minimization of SME and the screw configuration for solutions A to C

5 Conclusions

A previous study of iterative improvement algorithms in the optimization of individual criteria was developed. The knowledge gathered allowed the development of different multi-objective algorithms, such as TPLS, PLS and MO-ACO. The good performance obtained with the simple algorithms indicates that the incorporation of iterative improvement methods either in the MO-ACO or in MOEA algorithms can be a good solution to improve the performance of the algorithms reducing simultaneously the required number of evaluations. The solutions obtained have a physical meaning and are in agreement with the extrusion process requirements.

References

[1] C. García-Martínez, O. Cordón, and F. Herrera. A taxonomy and an empirical analysis of multiple objective ant colony optimization algorithms for the bi-criteria TSP. *European J. Oper. Res.*, 180:116–148, 2007.

[2] A. Gaspar-Cunha, J. A. Covas, and B. Vergnes. Defining the configuration of co-rotating twin-screw extruders with multiobjective evolutionary algorithms. *Polymer Engineering and Science*, 43:1159–1173, 2005.

[3] H. Hoos and T. Stützle. *Stochastic Local Search - Foundations and Applications*. Morgan Kaufmann Publishers, 2005.

[4] M. López-Ibáñez, L. Paquete, and T. Stützle. Hybrid population based algorithms for the bi-objective quadratic assignment problem. *Journal of Mathematical Modelling and Algorithms*, 5:111–137, 2006.

[5] L. Paquete and T. Stützle. A study of stochastic local search algorithms for the biobjective QAP with correlated flow matrices. *European J. Oper. Res.*, 169:943–959, 2006.

[6] C. Teixeira, R. Faria, J. Covas, and A. Gaspar-Cunha. Modelling flow and heat transfer in co-rotating twin-screw extruders. In *10th Esaform Conference on Material Forming*, 2005.

Engineering SLS Algorithms for Markov Logic Networks

Marenglen Biba
Department of Computer Science,
University of Bari, Italy
biba@di.uniba.it

Abstract

We present high performing SLS algorithms for learning and inference in Markov Logic Networks (MLNs). MLNs are a state-of-the-art representation formalism that integrates first-order logic and probability. Learning MLNs structure is hard due to the combinatorial space of candidates caused by the expressive power of first-order logic. We present current work on the development of algorithms for learning MLNs, based on the Iterated Local Search (ILS) metaheuristic. Experiments in real-world domains show that the proposed approach improves accuracy and learning time over the existing state-of-the-art algorithms. Moreover, MAP and conditional inference in MLNs are hard computational tasks too. This paper presents two algorithms for these tasks based on the Iterated Robust Tabu Search (IRoTS) schema. The first algorithm performs MAP inference by performing a RoTS search within an ILS iteration. Extensive experiments show that it improves over the state-of-the-art algorithm in terms of solution quality and inference times. The second algorithm combines IRoTS with simulated annealing for conditional inference and we show through experiments that it is faster than the current state-of-the-art algorithm maintaining the same inference quality.

1 Introduction

This paper focuses on Markov Logic Networks (MLNs) [5], a powerful representation for Artificial Intelligence and Machine Learning that has first-order logic and probabilistic graphical models as special cases.

State-of-the-art algorithms for structure learning of MLNs [1, 3] follow systematic search strategies that can lead to local optima and prohibitive learning times. The algorithm in [1] performs a beam search in a greedy fashion which makes it very susceptible to local optima, while the algorithm in [3] works in a bottom-up fashion trying to consider fewer candidates. We propose an algorithm based on the ILS metaheuristic that samples the set of local optima and performs a search in the sampled space. We show that the algorithm achieves improvements over the state-of-the-art algorithms.

Algorithm 1 SearchBestClause

Input: P:set of predicates, MLN:Markov Logic Network, BestScore: current best score, CLS: List of clauses, RDB:Relational Database)
CL_C = Randomly Pick a clause in CLS ∪ P; CL_S = $LocalSearch_{II}$(CL_S); BestClause = CL_S;
repeat
  CL'$_C$ = $Perturb$(CL_S);
  CL'$_S$ = $LocalSearch_{II}$(CL'$_C$,MLN,BestScore);
  if WPLL(BestClause,MLN,RDB) ≤ WPLL(CL'$_S$,MLN,RDB) **then**
    BestClause = CL'$_S$;
    Add BestClause to MLN; BestScore = WPLL(CL'$_S$,MLN,RDB)
  end if
  CL_S = accept(CL_S,CL'$_S$);
until two consecutive steps have not produced improvement on WPLL
Return BestClause

Maximum *a posteriori* (MAP) inference means finding the most likely state of output variables given the state of the input variables and this problem is NP-hard. Since for MLNs, the MAP state is the state that maximizes the sum of the weights of the satisfied ground clauses, it can be found by weighted MAX-SAT solvers. This paper presents a novel algorithm that exploits Iterated Local Search (ILS) [2] and Robust Tabu Search (TS) [7] metaheuristics. Experiments in real-world domains show that it outperforms the state-of-the-art algorithm, in terms of solution quality and inference running times.

Conditional inference in graphical models involves computing the distribution of query variables given evidence and it was shown to be #P-complete. An inference algorithm for MLNs is that of [4] where the authors combine ideas from satisfiability and MCMC methods. In this paper we propose the novel algorithm SampleIRoTS based on the ILS and RoTS metaheuristics. SampleIRoTS is then plugged in the novel algorithm MC-IRoTS. Experimental evaluation shows that on a large number of inference tasks, MC-IRoTS performs faster than the state-of-the-art algorithm while maintaining the same quality of predicted probabilities.

2 Learning and inference with metaheuristics

Structure learning proceeds by adding the best clause until no improving clauses are found for two steps. The best clause is found in Algorithm 1 where an ILS is performed with strong perturbations (restarting search from unit clauses) and greedy acceptance function. Regarding MAP inference, Algorithm 2 uses RoTS as local search procedure. This algorithm is then used in the SampleIRoTS algorithm to sample from the set of satisfying assignments where a RoTS step is taken with probability p and a simulated annealing step is taken with probability 1 - p. The novel probabilistic inference algorithm MC-IRoTS uses samples from SampleIRoTS to compute the probability of a formula by sampling worlds and counting those where

Algorithm 2 Iterated Robust Tabu Search

Input: C: set of weighted clauses in CNF, BestScore: current best score)
CL_C = Random initialization of truth values for atoms in C;
CL_S = $LocalSearch_{RoTS}(\text{CL}_C)$;
BestAssignment = CL_S; BestScore = Score(CL_S);
repeat
  CL'_C = $Perturb_{RoTS}$(BestAssignment);
  CL'_S = $LocalSearch_{RoTS}(\text{CL'}_C)$;
  if Score(CL'_S) $\geq$ BestScore **then**
    BestScore = Score(CL'_S)
  end if
  BestAssignment = accept(BestAssignment,CL'_S);
until k consecutive steps have not produced improvement
Return BestAssignment

Table 1: Accuracy results and learning time for all algorithms

Algorithm	CLL	AUC	Time (minutes)
BUSL	-0.196±0.003	0.201	9350
ILS	-0.088±0.004	0.240	1160

a certain formula holds.

3 Experimental results

Table 1 presents results on structure learning in terms of conditional log-likelihood (CLL), area under curve for precision recall (AUC) and learning time in minutes. For the greedy algorithm BS [1] we were not able to report results since it did not finish in 45 days of computation. The comparison is performed with the BUSL algorithm [3]. The results show that ILS is more accurate. Moreover, it is much faster than BUSL which takes on average 7 days to complete on the CORA dataset. Tables 2 and 3 present results for MAP and conditional inference on the UW-CSE dataset. Each row represents an experiment and the results are averaged over five folds of the dataset. The experiments are performed with a different number of ground predicates and clauses in order to verify the robustness for different configurations of the inference task. For MAP inference the comparison is performed with the current best algorithms for MLNs, MaxWalkSat with tabu (MWS-T) and MWS-T with restarts. All algorithms perform the same number of flips. As can be seen our algorithm finds better solutions. Running times show that in general the proposed algorithm is faster. For conditional inference comparison is performed with MC-SAT [4]. As can be seen, results in terms of CLL are comparable, but our algorithm performs better in terms of AUC. This means that it is able to produce more accurate probability estimates in less time than MC-SAT.

Table 2: MAP inference results averaged for each experiment over 5 folds of the dataset UW-CSE.

	IROTS		MWS-T		MWS-TR			
Exp	Cost	Time	Cost	Time	Cost	Time	preds	clauses
1	**77318.7**	**45.63**	77490.1	49.32	77411.0	66.76	3454	126104
2	**60328.2**	**35.94**	60374.6	38.91	60339.7	45.23	3454	126104
3	46.58	3.14	46.58	21.01	46.58	**15.71**	3454	126104
4	**11667.11**	**46.83**	12007.90	51.36	11727.50	61.42	3932	163485
5	**10108.82**	182.26	11433.16	**178.26**	10841.27	186.11	6796	458877
6	**311071.6**	31.68	350522.44	27.21	347179.74	**26.7**	18588	1280622

Table 3: Conditional inference results averaged for each experiment over 5 folds of the dataset UW-CSE.

	MC-IRoTS			MC-SAT				
Exp	CLL	AUC	Time	CLL	AUC	Time	preds	clauses
1	-0.031±0.008	0.047	**39.18**	-0.032±0.008	0.007	47.25	3454	126104
2	-0.032±0.008	0.007	**41.05**	-0.031±0.008	0.037	48.77	3454	126104
3	-0.029±0.006	0.138	**45.69**	-0.028±0.006	0.107	51.86	3454	126104
4	-0.007±0.002	0.243	**39.93**	-0.008±0.003	0.021	48.36	3932	163485
5	-0.021±0.004	0.013	**155.07**	-0.022±0.004	0.004	165.68	6796	458877
6	-0.024±0.002	0.012	**374.14**	-0.024±0.002	0.005	463.83	18588	1280622

4 Ongoing work

Ongoing work includes the dynamic adaptation of the perturbation operator in the structure learning algorithm. The current implementation uses only strong pertubation but this often leads to long search afterwards. Another improvement being investigated is the use of probabilistic acceptance criterion. Regarding MAP inference, changing search parameters during search may help explore better solutions. Perturbation is currently fixed, but we plan to dynamically change it like in [6]. and use a probabilistic choice in *accept*. For conditional inference, current work deals with the uniformity of sampling from the space of satisfying assignments which is the main point in producing reliable probability estimates. We plan to investigate how uniformity of sampling depends on the parameters of search and how parameters can be tuned to achieve good sampling.

References

[1] S. Kok and P. Domingos. Learning the structure of markov logic networks. In *Proc. 22nd Int'l Conf. on Machine Learning*, pages 441–448, 2005.

[2] H. Loureno, O. Martin, and T. Stützle. Iterated local search. In *Handbook of Metaheuristics*, pages 321–353. Kluwer Academic Publishers,

USA, 2002.

[3] L. Mihalkova and R. J. Mooney. Bottom-up learning of markov logic network structure. In *Proc. 24th Int'l Conf. on Machine Learning*, pages 625–632, 2007.

[4] H. Poon and P. Domingos. Sound and efficient inference with probabilistic and deterministic dependencies. In *Proc. 21st AAAI*, pages 458–463, 2006.

[5] M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62:107–236, 2006.

[6] K. Smyth, H. Hoos, and T. Stützle. Iterated robust tabu search for max-sat. In *Canadian Conference on AI,*, pages 129–144, 2003.

[7] E. Taillard. Robust taboo search for the quadratic assignment problem. *Parallel Computing*, 17:443–455, 1991.

Experiments on Adaptive Heterogeneous PSO Algorithms

Paolo Spanevello[1] *and Marco A. Montes de Oca*[2]

[1]*DEI, Università degli studi di Padova, Padua, Italy*
[2]*IRIDIA, CoDE, Université Libre de Bruxelles, Brussels, Belgium*
paolospane@gmail.com, mmontes@ulb.ac.be

Abstract

Particle Swarm Optimization (PSO) algorithms are used for solving real parameter optimization problems that may be characterized by discontinuities, plateaus, multiple local optima, and other features. In most PSO variants, the swarm is homogeneous, that is, all particles use the same search mechanism; however, given the possible complex nature of the solution space, a single search mechanism can be more effective in certain areas of the search space than in others. Thus, particle homogeneity can negatively affect the performance of PSO algorithms. A possible way to tackle this issue, is to use heterogeneous PSO algorithms, which are composed of particles that use different search mechanisms. In this paper, we propose a number of different strategies to allow particles to select, from a set of two candidates, their search mechanism according to information obtained during the optimization process. We show preliminary results and outline future research.

1 Introduction

In most particle swarm optimization (PSO) [2] algorithms, simple agents (called *particles*) move in the solution space of an objective function using exactly the same set of rules. These *homogeneous* PSO algorithms perform very differently when applied to problems with different features. This can be explained by realizing that, for example, a set of rules can be good for optimizing convex objective functions, but bad for optimizing multimodal ones. Thus, particle homogeneity constrains the number of problems to which a particular PSO variant can be effectively applied to. An alternative approach that has the potential of circumventing this problem is to use *heterogeneous* particles, that is, particles that are different from each other in terms of the rules that they use to search a problem's solution space. Indeed, previous research shows that a heterogeneous PSO algorithm, with two different types of particles, will usually perform better than the worst of the two homogeneous variants [4]. However, a problem with nonadaptive

heterogeneous PSO algorithms, which are those in which particles do not change search mechanism over time, is that they cannot fully exploit the best search mechanism at any given moment during the optimization process because the number of particles using one or another search mechanism is constant over time.

In this document, we describe a number of simple adaptation mechanisms that allow particles in a heterogeneous PSO algorithm to change their search mechanism over time (Section 2). Initial results suggest that more elaborate mechanisms than the ones presented here are needed to devise competitive adaptive heterogeneous PSO algorithms (Section 3).

2 Adaptive Heterogeneous Particle Swarm Optimizers

We say that a heterogeneous PSO algorithm is adaptive if the particles it is composed of are able to change search mechanism as a result of some event triggered during the optimization process. The mechanisms used by the particles to change search mechanism are referred to as *adaptation mechanisms*. The adaptation mechanisms that are explored in this paper are detailed below.

2.1 Stagnation Threshold

This mechanism is based on the idea that each agent is self-conscious of its performance and whenever it is "unsatisfied", it changes its search mechanism. It is implemented as follows: Each particle counts the number of consecutive function evaluations in which its personal best does not improve. If the value of the counter exceeds a certain threshold (a parameter), the particle switches search mechanism and resets the counter.

2.2 Difference-proportional Probability

A good search mechanism is one that allows a particle to find good solutions; thus, it is reasonable to make a particle "copy" the search mechanism of a particle that has found a better solution. For any given particle, the probability of copying the search mechanism of its best neighbor should increase when the difference between the particles' personal best solutions is favorable to the neighbor, and decrease when the opposite is true. We implement this mechanism using an approach similar to the pairwise comparison rule used in social learning dynamics models [5]. The probability that particle i adopts the search mechanism of particle j is defined as $p_{ij} = 1/\left(1+\exp\left(-\beta\frac{b_i-b_j}{|b_j|}\right)\right)$ where b_i and b_j are the objective function values of the personal best solutions of particle i and j respectively, and β is

a parameter that controls the sensitiveness of particle i to the differences in solution quality between itself and its best neighbor. In order to avoid the swarm adopting one single search mechanism we set a number of particles as "rigid", meaning that they do not change search mechanism. There is at least one 'rigid" particle for each search mechanism.

3 Experimental Setup and Results

We carried out a series of experiments using the two adaptation mechanisms presented above in model-of-influence heterogeneous PSO algorithms [4].

3.1 Setup

The experimental design examines four main factors:

1. **Problem**. We used the same benchmark functions as in [4] except Weierstrass. In all cases, we used their 100-dimensional versions. All algorithms were run 100 times on each problem for up to 10^6 function evaluations.

2. **Particle configurations**. We used best-of-neighborhood and fully-informed particles [3]. The constriction factor as well as the acceleration coefficients were set as suggested in the literature [1]. We ran experiments with fully-connected and ring topologies.

3. **Population size**. We used swarms of 100 particles. Each particle is initialized as being of one kind with probability $p = 0.5$ producing unbiased swarms (50% best-of-neighborhood – 50% fully-informed).

4. **Adaptation mechanism**. We used the two adaptation mechanisms described in the previous section with the following parameter settings. For the stagnation threshold mechanism, we used two thresholds: $\{2, 5\}$. For the difference-proportional probability mechanism, the following values for the β parameter were used: $\{5, 10\}$. The number of "rigid" particles was set to 5 particles of each kind. As a control, we use the static heterogeneous PSO in which particles do not change search mechanism during the optimization process, and both homogeneous swarms.

3.2 Results

In Tables 1 and 2, we show the average and standard deviation of the ranking obtained by the different PSO variants across all the benchmark functions. The rankings are grouped by population topology and run length (in function evaluations). The best rankings are highlighted. The results obtained

after 10^2 function evaluations are not shown as they are the result of random fluctuations during initialization (as we worked with 10^2 particles).

Table 1: Average and standard deviation of the ranking obtained by each PSO variant (using a fully connected topology) across benchmark problems at different run lengths (in function evaluations). The best rankings are highlighted.

	10^3	10^4	10^5	10^6
$\beta = 5$	4.1 (1.524)	3.9 (0.876)	3.8 (1.549)	3.7 (0.823)
$\beta = 10$	4.3 (1.494)	4.2 (1.317)	4 (1.764)	3.6 (1.713)
Stagnation threshold = 2	2.3 (1.059)	**2.5 (2.121)**	3.5 (1.65)	3 (1.826)
Stagnation threshold = 5	3.7 (1.636)	3.3 (1.418)	5.3 (1.059)	5.6 (0.966)
Static heterogeneous PSO	4.5 (1.434)	**2.5 (1.269)**	2.8 (1.229)	3.3 (1.16)
Homogeneous PSO: Best of neighborhood	7 (0)	5.4 (1.897)	**1.7 (1.059)**	**1.9 (1.287)**
Homogeneous PSO: Fully informed	**2.1 (1.729)**	6.2 (1.932)	6.9 (0.316)	6.9 (0.316)

Table 2: Average and standard deviation of the ranking obtained by each PSO variant (using a ring topology) across benchmark problems at different run lengths (in function evaluations). The best rankings are highlighted.

	10^3	10^4	10^5	10^6
$\beta = 5$	3.6 (1.43)	2.8 (0.789)	2.7 (0.632)	3.35 (1.226)
$\beta = 10$	3 (1.633)	2.7 (0.823)	**2.6 (0.658)**	3.55 (1.039)
Stagnation threshold = 2	5.8 (0.422)	7 (0)	7 (0)	7 (0)
Stagnation threshold = 5	4.4 (1.35)	4.7 (0.949)	4.2 (1.317)	3.15 (1.811)
Static heterogeneous PSO	3.5 (1.269)	4.1 (1.197)	4.2 (1.317)	**3.05 (0.864)**
Homogeneous PSO: Best of neighborhood	5.9 (2.132)	4.9 (2.079)	4.6 (2.271)	3.85 (1.93)
Homogeneous PSO: Fully informed	**1.8 (1.874)**	**1.8 (1.687)**	2.7 (2.263)	4.05 (1.554)

In all cases, except in the case of the difference-proportional probability mechanism with a ring topology and after 10^5 function evaluations, it is either a homogeneous PSO or the static heterogeneous PSO algorithm, the one that performs best. With a fully connected topology, the difference in performance between the algorithms using an adaptation mechanism seems to be insignificant. On the contrary, when using a ring topology, the difference-proportional probability mechanism seems to outperform the stagnation threshold approach. A possible explanation of this behavior is that the adaptation mechanisms favor too strongly the search mechanism that is more exploitative very early in a run. This is especially true for the difference-proportional probability mechanism because it makes the whole swarm adopt the strategy that works best during the first iterations.[1] An-

[1]The results that support this statement are not shown for the sake of conciseness.

other possible reason for this behavior is that since the natural tendency of a swarm (especially one using a constriction factor) is to converge, the effect of changing search strategy is reduced when the inter-particle distance is small. Preliminary results obtained with variants in which the particles' positions and velocities are reset after changes in the search mechanism seem to back this statement.

4 Conclusions and Outlook

Adaptive heterogeneous PSO algorithms are, conceptually, a possible way of circumventing the problem of selecting the best suited PSO algorithm for each specific problem. We have conducted experiments with two adaptation mechanisms which were aimed at selecting the best search mechanism from a set of two candidates during an optimization run. The results indicate that adaptation mechanisms that favor exploitative search mechanisms hinder, in the long run, the algorithms' performance. Possible causes of this problem are premature convergence in the search space and in search mechanism. Preliminary results suggest that more elaborate adaptation mechanisms are needed to obtain satisfactory results.

References

[1] M. Clerc and J. Kennedy. The particle swarm–explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation*, 6(1):58–73, 2002.

[2] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings of IEEE International Conference on Neural Networks*, pages 1942–1948, Piscataway, NJ, 1995. IEEE Press.

[3] J. Kennedy and R. Mendes. Neighborhood topologies in fully informed and best-of-neighborhood particle swarms. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 36(4):515–519, 2006.

[4] M. A. Montes de Oca, J. Peña, T. Stützle, C. Pinciroli, and M. Dorigo. Heterogeneous particle swarm optimizers. In P. Haddow et al., editors, *IEEE Congress on Evolutionary Computation – CEC 2009*, pages 698–705. IEEE Press, Piscataway, NJ, 2009.

[5] F. C. Santos, J. M. Pacheco, and T. Lenaerts. Cooperation prevails when individuals adjust their social ties. *PLoS Computational Biology*, 2(10):e140, 2006.

Exploiting Constraint-Neighbourhood Interactions

Alastair Andrew
Department of Computer & Information Sciences,
University of Strathclyde, Glasgow, UK
alastair.andrew@cis.strath.ac.uk

Abstract

Local Search (LS) and its related metaheuristics have a proven track record when it comes to solving many $\mathcal{NP}$-hard constrained problems. Complete methods such as those in Constraint Programming (CP) provide alternative approaches which make explicit use of the problem structure (as defined by the constraints) to improve solution speed and quality. Previously the most successful LS algorithms made use of domain knowledge, however, unlike in CP this was typically tightly coupled to a specific problem. The advent of Constraint-Based Local Search (CBLS) with its elegant separation of Model and Search offers the chance to harness the internal structure whilst retaining a clean problem independent search. We introduce a framework for detecting the useful interactions between search neighbourhoods and problem constraints and discuss how this information can be used in a Variable Neighbourhood Search (VNS) to improve efficiency.

1 Introduction

The basic idea behind LS is that by making a series of changes to a solution the global optimum can be reached. A neighbourhood is the set of solutions which can be created from an existing solution via the application of a perturbative function. In this paper we use the term neighbourhood to refer to this generating function rather than the resultant candidate set. The neighbours are assessed by the application of a fitness criteria. A successor is chosen and the process repeats itself until none of the neighbouring solutions offer an improvement on the current solution. Unfortunately when the search terminates it is most likely only at a local (rather than global) optima.

2 Rise of the Metaheuristics

In a bid to overcome this weakness the area of metaheuristics emerged in the 80's. Foremost amongst these techniques are Simulated Annealing (SA) [5] and Tabu Search (TS) [3]. SA allows the acceptance of worse solutions to promote diversification and to prevent the search from becoming trapped

at local optima. As the search progresses the likelihood of transitioning to a worsening state decreases, the exact probability is determined by the cooling schedule. By allowing the search to explore a wider range of states the chance of reaching the global optimum increases.

The TS algorithm takes an alternative approach based upon the introduction of an adaptive memory. Unlike complete techniques TS does not try to store all the previously visited states, it retains only recently visited states and then enforces that any future solutions are not amongst this *tabu list*. In this way the search is forced away from becoming trapped at optima. The *tabu tenure* controls how long a move remains tabu for.

Whilst these metaheuristics have been successful at increasing the power of LS based algorithms this has come at the cost of added complexity in the form of parameterisation. SA requires the setting of the cooling schedule and TS is dependant on the *tabu list* length and setting moves' *tabu tenure*. The performance of algorithms can be drastically improved solely by finding good parameter settings as shown in [2, 4] where automated parameter tuning approaches are taken. This still does not address the problem that LS' are typically tightly coupled and tuned to specific problems, and in some cases even problem instances. CP operates with clearly separated models, inspired by this notion that LS could (and should) be reusable and problem independent CBLS was developed, best exemplified by the Comet language [8]. Models are stated in a simple declarative form syntactically related to CP and advanced control abstractions allow the search to be loosely coupled. This, however, raises the dilemma that previously LS algorithms had explicitly exploited features of the problem structure to boost their performance, how can this be achieved in a problem independent fashion without sacrificing the CBLS = Model + Search goal?

3 Detecting Interactions

Our work seeks to answer that question by providing a formal framework expressed in Comet with which to guide the process. The first stage is to detect the interactions between the problem constraints and the search neighbourhoods. What is meant by an *interaction*? If a neighbourhood permutes an existing solution in such a way that it can alter the violations of that constraint it is said to *interact* with that constraint. We do not try to influence whether this interaction is positive or negative as the role of the neighbourhood within the search is simply to propose candidate solutions to the acceptance function. Constraints in this sense refers to the problem constraints not the actual implementation. For example a problem constraint precluding certain assignments could be represented as a series of `disequations` or a single `alldifferent`, these modelling decisions are irrelevant to our system.

We employ two main strategies to detect whether or not there is any interplay between a neighbourhood and constraint. Firstly we compare the id's of the variables involved, if the constraint and neighbourhood operate upon disjoint sets no interaction can occur. One minor caveat is that if a constraint is defined in terms of an intermediary variable the detector must be informed. If the sets overlap then we cannot rule out an interaction. By exploring the neighbourhood and monitoring the violations we can detect if the neighbourhood causes any changes, if so then we have proof of an interaction. We fully explore a neighbourhood and if no such interactions are found we assume that no interaction exists. Discovering the interactions has a worst case complexity of $\mathcal{O}(c*n)$ where n is number of neighbourhoods and c the number of problem constraints. The average case performance is much better, if the variable id comparison has not ruled out the possibility of an interaction the first move will usually lead some change in the violation state. The results of the detection process are cached so this only needs to be performed once for a given problem and neighbourhood collection. This component operates only through Comet interfaces and has no knowledge of the underlying constraint or neighbourhood's implementation.

3.1 Exploiting Interactions

Given that we have obtained an idea of how the application of a neighbourhood will effect the state of the constraint violations, how can this information be used within a search? Understanding the effect of a neighbourhood allows a more informed decision regarding the choice of neighbourhood at any given point to be made. The VNS framework [6] offers an alternative approach to the problem of escaping local optima. VNS hinges on the observation that the local optima are just that, local to a neighbourhood, only the global optimum is an optimum in all neighbourhoods. When VNS reaches an optimum it uses this as a sign that it should change neighbourhoods, following a predefined sequence.

The idea of using multiple neighbourhoods is not revolutionary, indeed it forms the basis of most LS approaches to hard combinatorial optimisation problems such as University Course Timetabling. Typically a subset of the constraints will be optimised until a solution which is feasible with respect to these is found, subsequent phases will try to satisfy the remaining constraints whilst preserving the state of the first phase constraints. To realise this style of multi-phase algorithm the designer is exploiting their knowledge about the interactions of search neighbourhoods and constraints.

The prevailing wisdom is to create these phases based around the hard and soft constraints of the problem. Hard constraints are those which cannot be violated in any feasible solution, soft constraints may be violated but the reduction of these violations is an optimisation objective. Partitioning based upon the hard and soft constraints has the attractive property that

in the latter stages all the solutions being explored will be feasible. Does this partitioning of the search into hard and soft phases actually hinder the search process? Work into oversubscribed scheduling problems showed that there can be efficiency gains made by allowing the search to transition through the *infeasible* areas [7] and harness what are described as *shortcuts*.

We are currently investigating two strategies for determining the order in which the constraints should be optimised based upon the information from the interaction detection. The first approach selects a neighbourhood for each constraint such that this neighbourhood interacts with as few other constraints as possible, then the constraints are sorted into descending order of their selected neighbourhood's interactions (breaking ties on neighbourhood size).

The interaction information also opens another avenue, each neighbourhood now has a known set of effects making it possible to reason about its potential to alter the state of the violations. The field of Automated Planning is one where there are a series of operators which can effect the state of the world, the world begins in an initial state and there is a desired goal state. A plan is an ordered list of operator applications which transforms the initial state into the goal state. In this case the neighbourhoods are the operators, the initial state is that the constraints are unsatisfied and the goal is that they are satisfied. We generate a planning problem in the standard PDDL format and pass it to an automated planner. The resulting plan is a sequence of neighbourhood applications which is read in and used as another possible neighbourhood ordering for the VNS. It should be noted the neighbourhood ordering implicitly contains the constraint ordering.

Optimising constraint models is another potential use of the interaction information. If none of the proposed neighbourhoods effect a given constraint it becomes redundant (providing the initial solution can be constructed to satisfy it). In our system the interactions are displayed visually as a bipartite graph, this is invaluable in the neighbourhood design process where it allows the algorithm designer to see which constraints are being missed by the current set of neighbourhoods or where there are many neighbourhoods with the same effects.

4 Constraint-Directed Variable Neighbourhood Search

Conventional VNS orders the neighbourhoods linearly by increasing size. Constraint-Directed Variable Neighbourhood Search (CDVNS) is our attempt to use the interaction information to make the VNS scheme more efficient. The interactions indicate when a neighbourhood will not be capable of causing a change in violations, thus during search neighbourhoods which will lead to no improvements can be omitted. The exploration of the

neighbourhoods is done much like a transitive closure, whenever an improving state is found the search returns to the starting neighbourhood, only when a neighbourhood is exhausted can the next in the list be explored.

The system which we have developed is flexible enough to allow the progression to be defined as any directed graph structure desired so, rather than just a list, it can now contain cyclic components. This allows multiphase algorithms to be succinctly captured as a configuration of transitions rather than being hard coded. CDVNS retains the structural simplicity of VNS and so can easily be hybridised with other metaheuristics.

5 Future Work

The work we have so far described is based upon existing neighbourhoods and determining their properties, an interesting future direction is to automatically create neighbourhoods with specific interaction properties. [1] introduced the concept of Constraint Oriented Neighbors, where constraints are used to explicitly create neighbourhoods. Their approach requires the constraints to expressed in Existential Monadic Second Order logic extended with counting ($\exists MSO^{+}$) and we would be looking to explore options which are independent of constraint implementation.

References

[1] M. Ågren. PhD thesis, Sweden, Dec.

[2] M. Birattari, T. Stützle, L. Paquete, and K. Varrentrapp. A Racing Algorithm for Configuring Metaheuristics. In *Proc. of GECCO 2002*, pages 11–18, July 2002.

[3] F. Glover and M. Laguna. *Tabu Search*. Kluwer Academic Publishers, Boston, MA, USA, 1997.

[4] F. Hutter, Y. Hamadi, H. H. Hoos, and K. Leyton-Brown. Proc. of cp 2006. In *Proc. of CP 2006*, volume 4204, pages 213–228, Sept. 2006.

[5] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi. Optimization by Simulated Annealing. *Science*, 220(4598):671–680, May 1983.

[6] N. Mladenović and P. Hansen. Variable Neighborhood Search. *Comput. Oper. Res.*, 24(11):1097–1100, Nov. 1997.

[7] M. F. Rogers, A. E. Howe, and L. D. Whitley. Looking for Shortcuts: Infeasible Search Analysis for Oversubscribed Scheduling Problems. In *Proc. of ICAPS 2006*, pages 314–323, June 2006.

[8] P. Van Hentenryck and L. Michel. *Constraint-Based Local Search*. The MIT Press, Aug. 2005.

Fast Search of Paths through Huge Networks

Jesica Rivero Espinosa
Computer Science Department,
Carlos III University, Madrid, Spain
jrivero@inf.uc3m.es

Abstract

There are lots of applications and systems requiring paths search across networks. Most of them have dependencies between the quality of the results and the size of the network they can manage. In addition, several of them require a fast response, despite the result would not be the optimal in quality (path cost). The target is obtaining good enough solutions in the shortest response time. There have been developed several proposals aimed to improve the performance of searches over huge networks. Crucial issues in this research are a good management of the stored data (for which making use of databases seems to be a good direction) and fast and good quality querying methods. Specifically, this work focuses on the problem of finding the path between two nodes through a huge network in the shortest response time, by applying spatio-temporal databases (DB) and metaheuristics.

1 Motivation

Systems which use graphs or networks are a useful technique to formalise and to structure certain kind of knowledge and querying it, mainly for the search of paths between two points. A typical application, for example, is to guide pedestrian and vehicles through a map. But there are also a lot of systems that can use networks for improving tasks which are not initially conceived to that formalization, by abstracting their information. For example, main concepts can be nodes and relations between concepts can be links.

Specifically, a network-based problem is found in managing circumstantial knowledge trough human-like interaction. That knowledge is handled by a Situation Model, taking into account several aspects [4], among others:

- Material aspect: encloses spatial and temporal information (passable points are nodes, and links represent connections with a cost).
- Operative aspect: regards tasks and transactions achieved through interaction (paths represent strategies or plans to attain some goal).

The work in the SOPAT project (Servicio de Orientación Personalizada y para el Turismo, financed by the Spanish Science and Innovation Ministry) realised during 2008, included, as main innovation, a basic Situation Model focused in the spatio-temporal aspect [8]. For that Interaction Domain, the scenario was small (a hotel), so Dijkstra's algorithm was found to be a good solution for finding paths through the network (stored in a Spatial DB). However, it was found that most problems would require bigger areas and lower granularity, so the network can reach giant dimensions. That increment has a very important influence in the response time, doing it inapplicable to this type of system (because of real time response requirement). Storing pre-processed paths is not a good solution because of the unthinkable amount of combinations, and also because the scenario is subject to structural changes that should be assimilated dynamically. These facts motivated the study of other kind of algorithms providing faster answers to path queries between two nodes in dynamic huge networks (hundreds of thousands of nodes) stored in a Spatio-Temporal DB.

Yet there are similar solutions, none of them suit to the posed problem, and if such algorithm is found a lot of systems would be improved, enabling them to manage big networks in real time.

2 Related Works

For many years networks have been used to represent information and to easily manage it. A classic problem is the search of a path between two points, for which resolution several algorithms have been proposed. They manage the information in main memory, obtaining fast response time, so their target is just to obtain the optimum path (minimum cost). Requirements evolution (greater domains, precision, etc) led to bigger networks, which in addition exceed the capacity of main memory and require secondary storage.

Such changes involve substantial performance loss, yet real time response is a requisite for most network-based systems. This forced a change of focus: optimal cost is desirable, but fast response time is essential. These guidelines can be seen in [1], which proposes to divide the global network in subnets and pre-process them. Queries are analysed in an abstract way, substituting subnets with direct links between their frontier nodes.

As the networks size is increased to provide more detailed and complete services, arises the need for a change in the information management. This support can be provided by a Database Management System with spatio-temporal information management. There are several DB suited proposals based on network pre-processing ([6], or [9]), given that its cost should have no incidence in the global performance. These approaches provide fast solutions, but their weakness is that they are restricted to constant information: any change in the network involves the invalidation of a great part of the

pre-processing, which updating entails to spend a lot of time, often affecting global performance. On the other hand, there are also some works [5] proposing solution quality restrictions with performance enhancement.

To reduce response time, a frequent way is to include metaheuristics. Specifically, ACO algorithms [2] are oriented to the search of paths across networks. This kind of algorithms minimize response time providing good enough solutions (close to optimal). Their main drawback is that current proposals (see [7], and [3]) are applied over small scenarios (networks of at most thousands of nodes), and need to be adapted to the massive amount of data required for huge networks.

3 Contribution

This proposal is aimed to solve the problem of fast algorithm to search paths in huge networks using DB (to store nodes and links) and metaheusristics (to search paths).

Regarding the DB, the amount of nodes and links stored may reflect the information requirements: more detailed information (or bigger problems) involve more nodes, but also more precision and service quality. Anyhow, modern systems of this kind demand to manage hundreds of thousands of nodes (even millions), and this is only practicable if supported by a DB.

Once reviewed the storage and basic management requirements, it is necessary to adapt ACO algorithms for profiting the DB facilities and for overcoming its limitations. Main issues to observe through this adaptation are summarized next:

- The time needed to update, insert, and delete information stored has to be reduced, because it affects to the time to give a solution to the query. This implies study the form in which the pheromone is going to be updated.
- Over the same network, different queries could be done in the same time. This concurrent access must be possible, and each execution must not affect to the others.
- Queries are going to be diverse (start and end nodes could not be the same). This fact must be taken into account, because the information of pheromone stored in the DB of previous searches does not have to influence over future queries if their destination nodes are not equals, but the one with different start node and equal end one could be useful (the experience of the system could be useful).
- Loops and lost ants have to be avoided, because these problems have a direct influence in the time of execution and are very normal in networks with the dimensions mentioned before.

- The time to give a first approximation of the path must be reduced, but the methods employed with this objective must not have influence in the capacity of adaptability of ants.

4 Evaluation

A spatio-temporal DB is required to evaluate the proposed algorithm. These experiments will be run over Oracle 11g, a commercial well spread DBMS that supports spatio-temporal information storage and management.

The evaluation goal is to demonstrate that the proposal works faster than other analogous algorithms with low quality loss (if any). Therefore, the evaluation methodology will simply consist in comparing the performance (and results) of the proposed extended ACO algorithm with other solutions actually used. All the algorithms will be run over the same network in the DB (randomly generated), in the same conditions (from physical resources to the programming language) and with the same querying load.

The algorithms to be compared with the proposal are the classical Dijkstra's algorithm (usually employed by commercial DBMS to search paths), a fragmented Dijkstra (as an evolved and adapted version of the previous) [1], and the basic metaheuristic ant colony algorithm (basic ACO) [2].

5 Conclusions and Future Work

There is a wide range of applications which require a fast algorithm to give a solution to a question of path between to nodes in huge networks (hundreds of thousands of nodes).

This research work aims to gather spatio-temporal DB technology and metaheuristics seeking a new search method, adapted to modern needs. Specifically, the spatio-temporal DB is used to store and to handle the network, and the search is performed applying ACO algorithms. The algorithm will suffer some transformations to exploit the potential of such technological joint. The resultant algorithm is going to be tested by comparing its performance with other currently used used algorithms. The proposal is being developed as a thesis work at the Computer Science Department of Carlos III University of Madrid.

Finally, it should be mentioned that this work is framed in THUBAN (Plataforma de Interacción Natural para Acompañamiento Virtual en Entornos Reales), a project supported by the Spanish Science and Innovation Ministry.

References

[1] E. P. F. Chan and H. Lim. Optimization and evaluation of shortest path queries. *VLDB Journal*, 16:343–369, 2007.

[2] M. Dorigo, G. Di Caro, and L. Gambardella. Ant algorithms for discrete optimization. *Artificial Life*, 5:137–172, 1999.

[3] L. Fattouh Ibrahim. Using of clustering and ant-colony algorithms cwsp-pam-ant in network planning. In *ICDT 2006*, 2006.

[4] J. Gee. *Introduction to Discurs Analysis*, volume 50. Routledge, 1999.

[5] Y. Kanza, E. Safra, Y. Sagiv, and Y. Doytsher. Heuristic algorithms for route-search queries over geographic data. In *16th ACM-GIS*, 2008.

[6] D. Kim and E. Hwang. A voronoi-based hybrid navigation system for road network database. In *ALPIT*, 2008.

[7] A. Mora, J. Merelo, J. Laredo, C. Milln, and J. Torrecillas. Chac, a moaco agorithm for compuatation of bi-criteria military unit path in the battlefield: Presenation and first results. *IJIS*, 00:1–12, 2009.

[8] J. Rivero, D. Cuadra, D. Valle, and F. Calle. Enhancing natural interaction with circumstancial knowledge. *ITSSA*, 4:122–129, 2008.

[9] E. Silva, A. Paiva, C. Baptista, and L. Menezes. Personalized path finding in road networks. In *NCM*, 2008.

Hyperheuristic as Component of a Multi-Objective Metaheuristic

Nadarajen Veerapen[1,2], *Dario Landa-Silva*[2], *Xavier Gandibleux*[1]
[1]*Département informatique, Faculté des Sciences et Techniques, Université de Nantes, Nantes, France*
[2]*ASAP Group, School of Computer Science, University of Nottingham, Nottingham, UK*
nadarajen.veerapen@etu.univ-nantes.fr,
jds@cs.nott.ac.uk, xavier.gandibleux@univ-nantes.fr

Abstract

The numerical peculiarities which inhabit the numerical instance of a MOCO problem may seriously decrease the effectiveness of an approximation method. To deal with this problem we propose a flexible two-phase method for MOCO. Phase 1 produces a good approximation of the efficient frontier. However it may not be of good enough quality in terms of density. The aim of phase 2 is to tackle this problem in a flexible way so as to deal with the potential numerical peculiarities. We test this proposition on the multi-objective Traveling Salesman Problem for which there exists a number of low-level heuristics.

1 Introduction

Multi-objective combinatorial optimization (MOCO) considers $p \geq 2$ (conflicting) objectives to find a set of efficient solutions within a set of discrete feasible solutions, see [3] for more details. It is assumed that a solution which optimizes all objectives simultaneously does not exist. A solution $\hat{x} \in X$ is said to be *efficient* if there is no $x \in X$ such that x *dominates* $\hat{x}$ ($x_k \leq \hat{x}_k, \forall k \in \{1, \ldots, p\}$ with at least one strict inequality).

Finding the whole set of efficient solutions is often not necessary. However, having a good distribution of an approximation of this set is useful for decision makers to perform informed choices.

As resolution methods try to get better results, they often become more complicated, requiring expert knowledge to use them effectively. The hyperheuristic approach [1] provides a high-level view of the problem. Its aim is to allow the user to provide a number of heuristics, usually low-level ones, to solve a problem and then the hyperheuristic tries to find on its own which heuristics are the best online, i.e. as the search progresses. As such, a hyperheuristic does not operate in the *solution space* (this is done by the selected

heuristics) but in the *heuristic space*. This makes it different from variable neighborhood search (VNS) [7] which looks for solutions based on the systematic change of neighborhood during the search. There exists a domain barrier between the hyperheuristic and the low-level heuristics and, ideally, the hyperheuristic requires no or minimal knowledge of how the heuristics work but relies instead on the analysis of one or more objective functions (knowing whether maximization or minimization are required) and heuristic performance indicators such as running time.

This work proposes the use of a hyperheuristic to make the efficient frontier more dense by starting from a sparse approximation of the efficient frontier. The method is applied to the multi-objective Traveling Salesman Problem (moTSP). To the best of our knowledge, hyperheuristics have yet to be applied to this problem.

The objective of the TSP is to find the shortest tour passing only once through every location which has to be visited (a Hamiltonian cycle). The multi-objective version of the TSP is of interest because it represents many practical situations, for example having to make a compromise between travel time and cost of transportation.

This paper presents some initial results and discusses possible improvements and future research.

2 Overview of the Method

Our algorithm consists of two phases which are described below. We note that the key contribution here is the second phase of the proposed approach.

2.1 First Phase

The first phase computes a very good subset of efficient solutions. For the bi-ojective TSP we use the same method employed by Lust and Teghem [6]. When considering problems with more objectives, in our case three objectives, the first phase uses the algorithm proposed by Przybylski et al. [8] where the exact solver is replaced by the Lin-Kernighan heuristic [5].

2.2 Second Phase

The second phase iteratively drives a subset of the population across the potential efficient frontier with the goal of maximizing the hypervolume indicator [9]. The non-dominated solutions found in this manner are added to the population.

Components and Layout of the Method. The algorithm requires a *set of one or more heuristics* whose only requirement is to implement a simple interface so that the search mechanism can manipulate them.

Next, the algorithm needs an initial *population of solutions* P_0, $|P_0| \geq 1$. It maintains an *archive* A of all non-dominated points found and, at each iteration, only uses a running population P of maximum size S. The *hypervolume indicator* allows us to consider the movement of each solution with respect to the current running population and not as a point on its own.

At each iteration, each solution $p \in P$ is considered, a heuristic is selected and the neighborhood of p is explored to find a new point which increases the hypervolume. Any non-dominated solution found during this process is added to the archive. Should a point with a strictly improving hypervolume be found, the same heuristic is applied to this new point in a descent fashion.

If the maximum size of the running population is not reached, improving solutions are added to P as is, otherwise they replace the point they were a neighbor of. If no solutions in P were moved during the last iteration, a new running population is randomly selected from the archive. This prevents the algorithm from getting stuck in a local optimum and contributes to the diversity of the solution set. Since calculating the hypervolume and non-dominated sorting occur constantly throughout the algorithm, it is better to keep S small. We arbitrarily choose $S = 20$.

Heuristic Selection Mechanism. At the start of the search, all heuristics have the same score of 1. The performance of the heuristics is inferred through a system of reward and punishment, whereby improving heuristics obtain a higher score and non-performing heuristics a lower one. The heuristic with the best rank is selected in each iteration of the process described above.

To supplement this strategy, a tabu list is also used to prevent worse heuristics from being used during a certain amount of time (even if it was performing well previously). This strategy is inspired by the one used in Burke et al. [2]. A heuristic is included in the tabu list if it has not been able to improve the distribution by moving a solution in the given amount of time it was allowed to run. However, if an improving solution has been found, the tabu list is cleared (it is also cleared if all heuristics turn out to be tabu). No aspiration criterion is used.

Next, we present some results and name the hyperheuristic Best Rank with Tabu List (BRTL).

3 Initial Experimental Results

A population of 11 low-level classic TSP heuristics is used: subpath insertions and swaps, the 2-exchange move and a dummy heuristic (the first and second cities are inverted). Running time was set to 30 s and the neighborhood of each move was explored for a maximum of 50 ms. Averages of 10 runs are reported. We compare our method with two recent algorithms: evo-

lutionary multi-objective simulated annealing (EMOSA) [4] and two-phase Pareto local search (2PPLS) [6]. Here, BRTL and 2PPLS share the same first phase (but not EMOSA).

BRTL manages to outperform EMOSA for the hypervolume indicator in three out of the four published results for convex instances (Table 1). BRTL performs less well than our implementation of 2PPLS for instances with less objectives (Table 2). However, with more objectives, if the running time of 2PPLS is capped in the same way as for BRTL, initial results indicate that BRTL obtains better results.

Instance	BRTL	EMOSA
kroAB50	**0.3544**	0.2839
kroBC50	**0.4327**	0.2809
kroAB100	**2.1782**	1.9060
kroBC100	1.8630	**1.9392**

Table 1: Hypervolume (10^{10}) comparison of BRTL with EMOSA

Instance	Algo.	$\mathcal{H}(10^8)$	R	Time(s)
kroAB100	BRTL	225.84	0.93516	30
	2PPLS	**226.11**	**0.93526**	**13**
Cluster100	BRTL	233.12	0.94672	30
	2PPLS	**233.35**	**0.94679**	**13**
kroAB200	BRTL	835.37	0.875358	30
	2PPLS	**1076.08**	**0.94507**	**20**
kroABC50	BRTL	*4092608*	*0.98286*	**30**
	2PPLS	3454695	0.97029	**30**
	2PPLS	–	–	3600+

Table 2: Phase 2 of BRTL and 2PPLS

4 Conclusion

Our approach is a new generic resolution method for MOCO. Work is still needed to evaluate the potential advantage of the hyperheuristic with regard to its flexibility when dealing with various numerical instances and its ability to intelligently switch between heuristics. Potential avenues of investigation include a simpler first phase. The second phase could be improved in a number of ways: better indicators of the quality of the distribution of set, intelligent selection of the running population and a smarter selection mechanism. The approach also needs to be tested on more instances and other problems.

Acknowledgements. The authors are grateful for Anthony Przybylski's insights in the implementation of phase 1 and for his helpful remarks.

References

[1] E. Burke, G. Kendall, J. Newall, E. Hart, P. Ross, and S. Schulenburg. Hyper-heuristics: An emerging direction in modern search technology. In *Handbook of Metaheuristics*, pages 457–474. Kluwer Academic Publishers, 2003.

[2] E. Burke, D. Landa-Silva, and E. Soubeiga. Multi-objective hyper-heuristic approaches for space allocation and timetabling. In *Metaheuristics: Progress as Real Problem Solvers*, pages 129–158. Springer, 2005.

[3] M. Ehrgott and X. Gandibleux. A survey and annotated bibliography of multiobjective combinatorial optimization. *OR Spectrum*, 22(4):425–460, 2000.

[4] H. Li and D. Landa-Silva. Evolutionary Multi-objective Simulated Annealing with Adaptive and Competitive Search Direction. In *Proceedings of IEEE Congress on Evolutionary Computation, 2008. CEC 2008*, pages 3310–3317. Springer, 2008.

[5] S. Lin and B. Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Operations Research*, pages 498–516, 1973.

[6] T. Lust and J. Teghem. Two-phase pareto local search for the biobjective traveling salesman problem. *Journal of Heuristics*, 2009, DOI: 10.1007/s10732-009-9103-9 (in print).

[7] N. Mladenovic and P. Hansen. Variable neighborhood search. *Computers and Operations Research*, 24(11):1097–1100, 1997.

[8] A. Przybylski, X. Gandibleux, and M. Ehrgott. A two phase method for multi-objective integer programming and its application to the assignment problem with three objectives. *To appear in INFORMS Journal on Computing*, 2009.

[9] E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. *IEEE transactions on Evolutionary Computation*, 3(4):257–271, 1999.

Integrating the decision maker's preferences into Multiobjective Ant Colony Optimization

Stefan Eppe
CoDE - Computer & Decision Engineering,
Université Libre de Bruxelles, Brussels, Belgium
stefan.eppe@ulb.ac.be

Abstract

Multiobjective Ant Colony Optimization (MO-ACO) metaheuristics have shown to be very successful in addressing hard multiobjective combinatorial optimization problems. As most other multiobjective heuristics, MO-ACOs are however usually providing the decision maker with the best possible approximation of the Pareto-optimal frontier, leaving him or her with the delicate task of making a choice in an often very large set of non dominated solutions. This paper presents an approach of tackling multiobjective combinatorial optimization problems by integrating a decision maker's *a priori* preferences.

1 Introduction

Most methods used nowadays for tackling multiobjective combinatorial optimization problems (MOCOPs) are based on the best possible approximation of the Pareto-optimal frontier. While this approach has established itself and remains largely used for bi-objective problems, its applicability becomes questionable when the number of objectives grows. Indeed, not only is the meaningful and interpretable representation of the set of efficient solutions becoming a non trivial issue when dealing with more than three objectives, but also is the quantity of non-dominated solutions getting so large that a decision maker has to be provided with additional tools for selecting a solution that best fits his needs.

A common approach for addressing this issue is to integrate knowledge about a decision maker's preferences into the optimization process. This decreases the computational cost of optimization by diminishing the number of solutions to be generated. Moreover, as the number of solutions proposed to the decision maker is lower than for a whole Pareto-optimal frontier, he or she can better focus on interesting regions of the decision space.

Such approaches are usually classified into *interactive* and *a priori* methods. Interactive methods are by far the most studied ones (e.g., [1]), mainly

because they resist the argument that a decision maker almost never knows the problem at hand in such a manner that he could provide all needed parameters with sufficient accuracy.

The goal of this paper is to propose new ways of integrating a decision maker's *a priori* preferences into multiobjective combinatorial optimization heuristics. We will in particular examine possible paths of applying preference modeling techniques developed by the multicriteria decision aid (MCDA) community in a multiobjective ant colony optimization (MO-ACO) algorithm.

2 Preference modeling

An important aspect of Multicriteria Decision Aid (MCDA) consists in modeling a decision maker's preference model in a given decision context. One usually distinguishes two main approaches: First, methods based on utility functions that synthesize the performance of a solution (called "action" in MCDA) on all considered criteria into a single value, hence defining a complete preorder on the set of possible actions. The second common approach may lead to a more complex relational structure by performing pairwise comparisons of actions (see [4] for a detailed introduction).

One particular kind of pairwise comparison relation called *outranking* can informally be stated as follows: "An action a outranks another action b iff a can be considered as least as good as b on the set of objectives without comparatively performing too bad on one objective". This vague definition can be expressed in multiple mathematical ways. In the following we will use the PROMETHEE methodology ([2]) as an exemplary outranking method to illustrate how MCDA tools can possibly be integrated into multiobjective optimization ACO's.

The PROMETHEE methods introduce a preference function $P_j\left(f_j(x), f_j(y)\right)$ that expresses the degree of preference of a solution x over a solution y considered on objective j (see Fig. 1).

In order to compare two solutions the preference functions have to be aggregated over all objectives. This is done by defining the following normalized preference index:

$$\pi_{xy} = \pi(x, y) = \sum_{j=1}^{Q} w_j \cdot P_j\left(f_j(x), f_j(y)\right) \tag{1}$$

where each objective j is associated with a user defined weight w_j (with $w_j > 0$, $\forall j \in \{1, \ldots, Q\}$ and $\sum_{j=1}^{Q} w_j = 1$). The preference index is characterized by the following properties: $\pi_{xy} \geq 0$, and $\pi_{ij} + \pi_{ji} \leq 1$. Once aggregated, any two solutions can be compared in a natural way: The higher the index π_{xy}, the higher the preference of solution x over solution y.

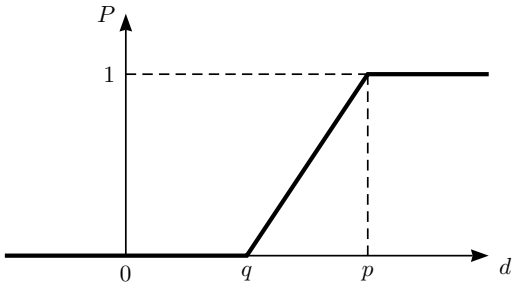

Figure 1: Example of a preference function that allows to model a decision makers preference degree P depending on the difference $d = f_j(x) - f_j(y)$ for objective $j \in \{1, \ldots, Q\}$ with two parameters: the indifference threshold q and the preference threshold p (Preference function type 5, taken from [2]).

3 Integration of a preference model into MO-ACO

A decision maker often has a more or less rough idea on values of the preference modeling parameters presented in the previous section. Allowing him to express his (partial) knowledge thus narrows the problem, hence allowing him a deeper and more focused exploration of the regions of the decision space that are of interest to him. Although applicable to virtually any MOCOP, we will describe our integration approach on the example of a multiobjective TSP (with n cities) in order to allow a better grip on the presented ideas.

At its different stages, an ACO algorithm (we refer to [3] for a thorough introduction to the ant colony optimization metaheurstic) needs to compare solutions either at component or solution level: The component level comparison is required to state - under consideration of multiple objectives - which city, denoted j, is the closest to another given city i (this is needed to compute heuristic information, build candidate lists of nearest neighbours, etc.). At solution level, the comparison of two generated solutions mainly allows to determine which ants will be allowed to update the pheromone trail(s).

Using the notations defined in section 2, we will now take a closer look at both levels:

Solution Level Let $F_i(X_k)$ be the evaluation in respect to criteria i of the solution $X_k = \left(x_k^{(1)}, \ldots, x_k^{(n)}\right)$ built by ant k. The preference degree of this solution over a solution built by ant l will be given by $\pi(X_k, X_l)$. The aggregation of all pairwise preference degrees yields the positive and negative outranking flows of each solution X_k, respectively denoted by Φ_k^+

and Φ_k^-, and defined by the following equations:

$$\begin{cases} \Phi_k^+ = \Phi^+(X_k) & = \frac{1}{m-1} \sum_{i=1}^{m} \pi(X_k, X_i) \\ \Phi_k^- = \Phi^-(X_k) & = \frac{1}{m-1} \sum_{i=1}^{m} \pi(X_i, X_k) \end{cases} \tag{2}$$

This definition is based on Eq. 1 and follows the PROMETHEE I method, explained in detail in [2]. Notice that $\Phi_k^+ \in [0,1]$ and $\Phi_k^- \in [0,1]$. Intuitively, the positive flow Φ_k^+ expresses the outranking *power* of the solution of ant k compared to all other ants. On the other hand, Φ_k^- quantifies its outranking *weakness*. Based on the preceding, an intuitively quite forward manner of implementing these flows would be to directly use Φ_k^+ to reinforce the trail associated to solution k, while using Φ_k^- to weaken (like evaporation) that same trail in the pheromone matrix. The resulting variation of pheromone could thus be defined as $\Delta\tau = \mu \cdot \Phi_k^+ - \rho \cdot \Phi_k^-$, where μ and ρ are two parameters to be chosen.

Component Level Let $x_k^{(l)}$ be the l-th component of the solution X_k; in the case of a TSP, $x_k^{(l)} = \left(a_k^{(l)}, b_k^{(l)}\right)$ represents a pair of connected nodes (i.e., cities) belonging to a constructed tour. In order to determine the nearest neighbours of a given node a, we need to define a way of comparing two solution components. We propose to define the *component net outranking flow* $\phi\,(a,b)$ for two nodes a and $b \in C$:

$$\phi_{ab} = \phi\,(a,b) = \frac{1}{m-1} \sum_{c \in C} \left[\pi\,(x_{ab}, x_{ac}) - \pi\,(x_{ac}, x_{ab})\right] \tag{3}$$

where $x_{ab} = (a,b)$ and $x_{ac} = (a,c)$ are respectively the connections between nodes a and b, and nodes a and c. In this equation C represents the set of all nodes of the construction graph. This definition is strongly inspired by the net flow of the PROMETHEE II method (that provides a complete ranking on a set of actions).

In this way, the nearest neighbours of any node a can be determined by decreasingly sorting the set of component net outranking flows $\phi(a,b)$, $\forall b \in C$. Any pair of connections (a,b) and (c,d) can also easily be compared by comparing their respective component net outranking flows ϕ_{ab} and ϕ_{cd}.

Since the evaluation functions f_i and F_i are used at different scales (f_i scales a component, while F_i scales a solution, i.e., an aggregation - often a sum, $F_i(X_k) = \sum_{x \in X_k} f_i(x)$ of component evaluations), it is useful to take a *relativistic* variation of the preference functions P, that is using relative thresholds in stead of absolute once.

4 Conclusion

This paper opens some promising perspectives on how a finer preference model could be used in conjunction with a MO-ACO for tackling multiobjective hard combinatorial optimization problems. We expect interesting results to show up thanks to the flexible representation of the decision maker's preference structure.

References

[1] J. Branke, K. Deb, K. Miettinen, and R. Slowinski, editors. *Multiobjective Optimization: Interactive and Evolutionary Approaches*, volume 5252 of *Lecture Notes in Computer Science*. Springer, 2008.

[2] J.-P. Brans and B. Mareschal. *Promethee Methods*, chapter 5, pages 163–195. Multiple Criteria Decision Analysis, State of the Art Surveys. Springer Heidelberg, 2005.

[3] M. Dorigo and T. Stuetzle. *Ant Colony Optimization*. The MIT Press, 2004.

[4] M. Oeztuerk and A. Tsoukias. *Preference Modelling*, chapter 2, pages 27–71. Multiple Criteria Decision Analysis, State of the Art Surveys. Springer Heidelberg, 2005.

Stochastic Local Search Strategies for Reverse Engineering of Gene Regulatory Networks

Francesco Sambo
Ph. D. School in Information Engineering,
Department of Information Engineering,
University of Padova, Padua, Italy
francesco.sambo@dei.unipd.it

Abstract

The problem of Reverse Engineering of Gene Regulatory Networks consists of the inference of causal relations between genes from the observation of gene temporal profiles. If causal relations are modelled by a system of equations, fitting the system parameters can be seen as an optimization problem, where the function to be minimized is the error between the real and the estimated temporal profiles. We present here a study on the application of a mixed optimization approach to this problem, with a discrete search step in the space of network structures and a continuous search step in the space of system parameters.

1 The biological problem

All the information necessary for an organism to live is coded in the genes of its DNA, and almost every biological function in living organisms is carried out by proteins; DNA molecules are *transcribed* into mRNA molecules, which, in turn, direct chemical machinery which *translates* the nucleic acid message into a protein [4].

Some proteins have the role to activate or inhibit the transcription of genes and to control the translation of mRNA into new proteins; the process by which genes, through the proteins they code, control the expression (*i.e.* the mRNA transcription rate) of other genes is known as *genetic regulation*.

The expression rate of the whole genome can be monitored through the new technology of DNA microarray [7]: each microarray experiment is a snapshot of the transcription level of every single gene, and different experiments can be replicated under various genetic, chemical and environmental conditions or in subsequent temporal instants.

One of the goals of microarray experiments is to infer regulatory relations between genes from the analysis of gene expression profiles, gaining finally

a *Gene Regulatory Network* (GRN), which is a graph in which nodes represent genes or proteins and two or more nodes are connected if a regulatory relation exists between them.

The inference of GRNs is intrinsically difficult for a number of reasons: first, the number of genes in an organism is $O(10^3 \sim 10^4)$, whereas the number of observations for each gene is usually limited to $O(10^1 \sim 10^2)$, thus making the problem underdimensioned. Moreover, regulatory relations are usually differential and highly nonlinear, and DNA microarray experiments are affected by a rather large amount of noise.

2 State-of-the-art

The inference of a GRN from a set of gene expression time series consists basically of two steps: choice of a model to describe temporal data and fit of the model to data. In the literature, a wide spectrum of different models have been proposed to describe gene regulation (for a survey see [2]): Relevance Networks, Boolean Networks, Dynamic Bayesian Networks and systems of additive or differential equations, being them linear, ordinary nonlinear or S-systems.

For Relevance Networks, Boolean Networks and Dynamic Bayesian Networks there is no straightforward way to define an error measure for a particular fit with respect to expression data, since no mathematical description of model variables is considered; systems of equations, on the contrary, allow one to analytically describe gene profiles. Thus, error measures, such as Relative Squared Error between real and estimated profiles, can be defined to assess the fitness of a particular assignment of values to the parameters of the model. The problem is then mapped to an optimization problem, in which the model's parameters form the search space and the error is the cost function to be minimized.

For all the aforementioned reasons, the Stochastic Local Search community has focused mostly on systems of equations to model the GRN: in [5, 16, 17], mixed discrete and continuous optimization techniques are exploited to fit data to a Recurrent Neural Network, which basically consists of a system of nonlinear differential equations; [6, 8] model the network with an S-system and exploit evolutionary algorithms for the inference; [15] is a comparison of a set of evolutionary algorithms on the inference of S-systems.

All the cited works have two major limitations: first, they estimate derivatives of the profiles directly from time series data, either with interpolation or with finite difference approximation, rather than numerically solving the whole system of equations. Such an approach, though computationally faster, amplifies the effects of noise and requires a large amount of data points. Moreover, trials are usually run on one or two simulated networks of small size (5 to 10 genes), preventing the results from being a

general assessment of average algorithmic performance.

3 Main contributions

Our three main contributions to the solution of the Reverse Engineering problem are the following.

CNET

We designed *CNET* [11, 12], an algorithm for the inference of causal relations between genes from gene temporal profiles. For each gene, the algorithm searches, among all the possible sets of causes, for the set of causes that minimizes a heuristic scoring function. The function exploits information-theoretic criteria and is carefully designed to take into account real features of gene profiles, such as noise and variable delays in the regulatory effects. CNET, tested both on simulated and real data, exhibits a performance comparable to the state of the art.

Topological properties of Gene Regulatory Network

We studied the relations between the performance of two Reverse Engineering algorithms (CNET and Dynamic Bayesian Networks) and the structural and topological properties of the GRN to be inferred [1, 10, 13]. Results show that performance of the algorithms is higher when the biological system is externally stimulated and that the edges of the regulatory network which are closer to the stimulated gene are easier to be inferred. Moreover, we observed that relations with a single regulator are identified with higher accuracy than relations with a combination of effects from different regulators, and that there are network *motifs*, *i.e.* recurrent substructures, which are significantly easier to be identified than others.

Fitness distance correlation analysis

Using Recurrent Neural Networks to model regulatory relations, we systematically studied the fitness landscape induced by minimizing the Relative Squared Error between real and estimated temporal profiles [14]. Results indicate that the generated landscapes have a positive fitness-distance correlation, but the error values span several orders of magnitude over very short distance variations. This suggests that the fitness landscape has extremely deep valleys, which can make general-purpose state-of-the-art continuous optimization algorithms exhibit poor performance.

An analysis based on perturbations of the optimal network topology showed that the portions of the search space that correspond to networks structurally close to the optimum present more organization in the fitness

landscape. The region close to the optimum can thus be a local basin of attraction for an algorithm which searches in the discrete space of network structures.

Finally, we run two state-of-the-art continuous optimization algorithms, NEWUOA [9] and CMA-ES [3], on a set of simulated test instances and in two different experimental conditions: first without a priori information and then with full information on network structure (*i.e.* configuration of zero and nonzero parameters) but no information on the values of nonzero parameters. Without a priori information, none of the algorithms is able to converge to the global optimum; with structural information, CMA-ES converges in almost every run, whereas NEWUOA's behaviour strongly depends on the particular network to infer. It converges easily in the majority of cases but is unable to escape from local optima for some problem instances.

Given all the previous observations, we concluded that a two-phase mixed optimization algorithm, which alternates between a search step in the discrete space of network structures and a search step in the continuous space of nonzero system parameters, has the potential of reaching high-quality solutions for the problem of network inference.

4 Future directions

As far as future directions, we plan to design, implement and test the two-phases mixed optimization algorithm, exploiting CMA-ES for the continuous search phase, given its good and stable behaviour, and a standard local search strategy with best improvement rule and multiple restarts for the discrete phase. We will focus on exploiting locality when evaluating the discrete neighbourhood of a network structure, thus avoiding re-optimizing unchanged system parameters in the subsequent continuous search step. Moreover, we will study how to efficiently plug structural a priori information, such as sparsity of the network or knowledge about certain regulatory relations, into the algorithm. Finally, we will consider how to merge the results of multiple runs of the same stochastic algorithm.

References

[1] S. Badaloni, M. Falda, and F. Sambo. Scale-free structure and topological properties in reverse engineering of gene regulatory networks. In *Proc. of WIVACE2008*, Venezia, Italy, 2008.

[2] H. de Jong. Modeling and simulation of genetic regulatory systems: A literature review. *Journal of Computational Biology*, 9(1):67–103, 2002.

[3] N. Hansen. The CMA evolution strategy: a comparing review. In *Towards a new evolutionary computation. Advances on estimation of distribution algorithms*, pages 75–102. Springer, 2006.

[4] L. Hunter. Life and its molecules: A brief introduction. *AI Magazine - Special issue on AI and Bioinformatics*, 25(1):9–22, 2004.

[5] K. Kentzoglanakis, M. Poole, and C. Adams. Incorporating heuristics in a swarm intelligence framework for inferring gene regulatory networks from gene expression time series. In *Proc. of ANTS2008*, pages 323–330, 2008.

[6] S. Kimura, K. Ide, A. Kashihara, M. Kano, M. Hatakeyama, R. Masui, N. Nakagawa, S. Yokoyama, S. Kuramitsu, and A. Konagaya. Inference of S-system models of genetic networks using a cooperative coevolutionary algorithm. *Bioinformatics*, 21(7):1154–1163, 2005.

[7] M. Molla, M. Waddell, D. Page, and J. Shavlik. Using machine learning to design and interpret gene-expression microarrays. *AI Magazine - Special issue on AI and Bioinformatics*, 25(1):23–44, 2004.

[8] N. Noman and I. Iba. Reverse engineering genetic networks using evolutionary computation. *Genome Informatics*, 16(2):205–214, 2005.

[9] M. J. D. Powell. *Large-Scale Nonlinear Optimization*, volume 83 of *Nonconvex Optimization and Its Applications*, chapter The NEWUOA software for unconstrained optimization, pages 255–297. Springer-Verlag, 2006.

[10] F. Sambo, B. Di Camillo, M. Falda, G. Toffolo, and S. Badaloni. Evaluation of local reliability of gene networks inferred from time series expression data. In *RECOMB Satellite on Regulatory Genomics and Systems Biology - Abstract Book.*, page 121, Boston, MA., 2008.

[11] F. Sambo, B. Di Camillo, M. Falda, G. Toffolo, and S. Badaloni. CNET: an algorithm for the inference of gene regulatory interactions from gene expression time series. In *Proc. of IDAMAP09*, Verona, Italy, 2009.

[12] F. Sambo, B. Di Camillo, and G. Toffolo. CNET: an algorithm for reverse engineering of causal gene networks. In *Proc. of NETTAB2008.*, pages 134–136, Varenna, Italy, 2008.

[13] F. Sambo, B. Di Camillo, and G. Toffolo. Role of network structure and experimental design on the performance of two reverse engineering methods. In *Proc. of ECCB2008*, Cagliari, Italy, 2008.

[14] F. Sambo, M. A. Montes de Oca, B. Di Camillo, and T. Stützle. On the difficulty of inferring gene regulatory networks: A study of the fitness landscape generated by relative squared error. In *Proc. of AE09*, Lecture Notes in Computer Science, Strasbourg, France, 2009.

[15] C. Spieth, R. Worzischek, F. Streichert, J. Supper, N. Speer, and A. Zell. Comparing evolutionary algorithms on the problem of network inference. In *Proc. of GECCO2006*, pages 305–306, 2006.

[16] R. Xu, G. K. Venayagamoorthy, and D. C. Wunsch, II. Modeling of gene regulatory networks with hybrid differential evolution and particle swarm optimization. *Neural Networks*, 20(8):917–927, 2007.

[17] R. Xu, D. Wunsch II, and R. Frank. Inference of genetic regulatory networks with recurrent neural network models using particle swarm optimization. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 4(4):681–692, 2007.

Tabu Search and Simulated Annealing for Tackling Large QAP Instances

Mohamed Saifullah Hussin and Thomas Stützle
IRIDIA, CoDE, Université Libre de Bruxelles, Brussels, Belgium
{mbinhuss, stuetzle}@ulb.ac.be

1 Introduction

The Quadratic Assignment Problem (QAP) is considered to be one of the hardest combinatorial optimization problems. The largest instance ever solved from QAPLIB, a benchmark library for the QAP, is of size 36. In this article, we compare the performance of Simulated Annealing (SA), Robust Tabu Search (RoTS), and Reactive Tabu Search (RTS) on solving large QAP instances. Several comparisons between SA and Tabu Search (TS) are reported [2, 3], but none on tackling large instances.

2 Algorithms

In this article, we compare the performance of SA and two TS variants; RoTS and RTS. SA has been applied to various difficult optimisation problems since its introduction in the early 1980's, including the QAP. In this article, we adopt a basic SA to generally justify the potential of SA in solving problems of large dimension. As for TS, we apply RoTS by Taillard [5] which was reported to achieve very good performance on the QAP, and RTS by Battiti and Tecchiolli [1]. TS was reported to perform better than SA [2] in terms of CPU time needed to reach a solution quality which is 1% from the best known solutions. The instances considered then, however are rather small instances.

This article is structured as follows. In the next section, we introduce the benchmark instances we use in this article. Experimental results with the comparison between RoTS, RTS, and SA are given in Section 3 and we conclude in Section 4.

3 Experimental results

3.1 Experimental setup

The experiments reported in this article have been run on Intel Xeon 2.4 Ghz quad-core CPUs with 6MB cache and 8 GB of RAM running under

Cluster Rocks Linux. Only a single core is used for computations, due to the sequential implementation of the algorithms. The stopping criteria for the experiments were based on the CPU time taken by our re-implementation of Taillard's RoTS [5] to perform $10000 \cdot n$ iterations, where n is the instance size. Results are collected at CPU time, $t_1 = 1000 \cdot n$ iterations, and $t_2 = 10000 \cdot n$ iterations. Algorithms studied in this article are coded in `C` and compiled with `gcc` version 3.4.6.

Studies of algorithms for the QAP are normally done on instances from QAPLIB. However, the number of large instances (size 100 and above) in QAPLIB is limited, which prevents a thorough analysis of solver performance on large instances. Therefore we generated a new set of QAP instances which, comparable to those used in the research of the Metaheuristics Network [4]. Generating our own instances allows us to generate a systematically varied set of instances, where each set is of acceptably similar characteristics.

We have generated four classes of instances with Euclidean distance matrices (ES) and four with Grid distance matrices (GS), where both come with structured flow matrices. These four instance classes differ mainly in the sparsity of the flow matrices (defined as the percentage of zero entries) and the distance or flow dominance values (variation coefficient of the distance and flow matrix entries multiplied by 100). Instances are named based on the type of distance matrix and the sparsity of flow matrix, e.g., ES.25 refers to instance with Euclidean distance matrix and structured flow matrix with sparsity 0.25. We have generated for each instance class 100 instances.

3.2 Comparison of RoTS, RTS, and SA

In Table 1, we present the comparison of RoTS, RTS, and SA on instances of size 50 and 100 at t_1 and t_2. For each instance class, we compute the average solution quality across 100 instances, and we normalize the results w.r.t. the best performing algorithm. We used the pairwise Wilcoxon test with Holm corrections for multiple comparisons to check the statistical significance of our results. The difference between the corresponding algorithms is significant if the p-value for the comparison is less than $\alpha = 0.05$. On instance size 50 and 100, SA is clearly the best performing algorithm at t_1 for all instance classes, except on very sparse instance size 50. Similar results are observed on instance size 50 at t_2, except on instance class GS.67, where RoTS performs the best. At t_2, RoTS shows a better average result on dense instance size 100, while on other instance classes, SA remains as the best performing algorithm.

We then compared the performance of RoTS and SA on larger instances, and present the results in Table 2. For this experiment, 30 randomly selected instances from each class are considered to reduce the total computation time for running the experiments. Statistically significant differences in the table

Class	Size	t_1	RoTS	RTS	SA	t_2	RoTS	RTS	SA
ES.25	50	4	*0.20*	*1.36*	**0.00**	40	0.03	*0.95*	**0.00**
ES.67	50	4	*0.70*	*2.72*	**0.00**	40	0.14	*2.24*	**0.00**
ES.90	50	4	0.72	1.22	**0.00**	40	0.16	*2.49*	**0.00**
ES.97	50	4	*6.48*	**0.00**	*30.70*	40	*2.10*	**0.00**	*22.60*
GS.25	50	4	*0.54*	*1.40*	**0.00**	40	*0.04*	*0.56*	**0.00**
GS.67	50	4	*1.08*	*2.75*	**0.00**	40	**0.00**	*1.20*	*0.08*
GS.90	50	4	*4.48*	*3.97*	**0.00**	40	*2.33*	*3.92*	**0.00**
GS.97	50	4	*0.26*	**0.00**	0.03	40	*0.01*	**0.00**	*0.01*
ES.25	100	33	*0.42*	*0.78*	**0.00**	300	**0.00**	*0.67*	*0.19*
ES.67	100	33	*1.63*	*2.42*	**0.00**	300	*0.27*	*1.22*	**0.00**
ES.90	100	33	*9.35*	*13.04*	**0.00**	300	*4.76*	*11.12*	**0.00**
ES.97	100	33	*15.81*	*8.27*	**0.00**	300	*8.69*	*8.98*	**0.00**
GS.25	100	33	*1.47*	*0.64*	**0.00**	300	*0.05*	*0.35*	**0.00**
GS.67	100	33	*2.35*	*1.07*	**0.00**	300	*0.08*	*0.45*	**0.00**
GS.90	100	33	*31.70*	*23.80*	**0.00**	300	*5.94*	*15.28*	**0.00**
GS.97	100	33	*27.62*	*9.67*	**0.00**	300	6.62	7.78	**0.00**

Table 1: Comparison of *RoTS*, *RTS*, and *SA* on instance size 50 and 100. Statistically signicant differences are indicated in italics font, while the best solutions are in bold font

are indicated in italics font, while the best solutions are in bold font. SA is again the best performing algorithm at t_1 on all instance classes considered. Different results are observed at t_2, where RoTS generally performs better than SA on dense instances.

Finally, we compared the performance of RoTS, RTS, and SA on several QAPLIB instances. Similar stopping criteria as the previous experiments are used. For each instance, we conducted 30 independent trials, and the solutions obtained were recorded for comparison. The results of the comparison are given in Table 3. We normalize the results obtained w.r.t. the results published at QAPLIB website (http://www.opt.math.tugraz.ac.at/qaplib/inst.html). SA is the best performing algorithm only on two smallest QAPLIB instances considered, tai50b and tai60b, while for larger instances, RoTS and RTS clearly show better performance.

4 Conclusions

In this paper, we have reported the comparison of selected metaheuristics on large generated QAP instances and selected QAPLIB instances, where we have obtained interesting results from our experiments. The results obtained suggest that SA surpasses RoTS and RTS when a small number of iterations are executed. However, when the number of iterations increased, RoTS and

Class	Size	t_1	RoTS	SA	t_2	RoTS	SA
ES.25	200	265	*0.51*	**0.00**	2650	**0.00**	*0.02*
ES.67	200	265	*1.05*	**0.00**	2650	0.12	**0.00**
ES.90	200	265	*16.65*	**0.00**	2650	*0.46*	**0.00**
ES.97	200	265	*24.24*	**0.00**	2650	*6.37*	**0.00**
GS.25	200	265	*1.10*	**0.00**	2650	**0.00**	*0.05*
GS.67	200	265	*1.57*	**0.00**	2650	**0.00**	*0.13*
GS.90	200	265	*23.37*	**0.00**	2650	**0.00**	*0.01*
GS.97	200	265	*67.90*	**0.00**	2650	*9.79*	**0.00**
ES.25	300	900	*0.93*	**0.00**	9000	**0.00**	*0.11*
ES.67	300	900	*2.03*	**0.00**	9000	0.08	**0.00**
ES.90	300	900	*34.52*	**0.00**	9000	*1.26*	**0.00**
ES.97	300	900	*60.44*	**0.00**	9000	*12.52*	**0.00**
GS.25	300	900	*1.80*	**0.00**	9000	**0.00**	*0.04*
GS.67	300	900	*1.84*	**0.00**	9000	**0.00**	*0.08*
GS.90	300	900	*44.18*	**0.00**	9000	*0.93*	**0.00**
GS.97	300	900	*159.41*	**0.00**	9000	*24.56*	**0.00**
ES.25	500	4220	*0.57*	**0.00**	42200	**0.00**	*0.07*
ES.67	500	4220	*1.84*	**0.00**	42200	*0.06*	**0.00**
ES.90	500	4220	*24.42*	**0.00**	42200	*0.85*	**0.00**
ES.97	500	4220	*68.25*	**0.00**	42200	*7.95*	**0.00**
GS.25	500	4220	*1.13*	**0.00**	42200	**0.00**	*0.05*
GS.67	500	4220	*2.10*	**0.00**	42200	**0.00**	*0.10*
GS.90	500	4220	*18.75*	**0.00**	42200	**0.00**	0.26
GS.97	500	4220	*173.65*	**0.00**	42200	*21.91*	**0.00**

Table 2: Comparison of *RoTS* and *SA* on instance size 200, 300, and 500

RTS beat SA, particularly on dense instances.

SA which introduces a single neighborhood exchange for every iteration provides a very fast and cursory search mechanism that benefits from having a large number of zero entries in the flow matrices, thus obtains better results than TS when solving sparse instances.

References

[1] R. Battiti and G. Tecchiolli. The reactive tabu search. *ORSA Journal on Computing*, 6(2):126–140, 1994.

[2] R. Battiti and G. Tecchiolli. Simulated annealing and tabu search in the long run: A comparison on QAP tasks. *Computer and Mathematics with Applications*, 28(6):1–8, 1994.

Class	Size	t	RoTS	RTS	SA
tai50b	50	40	0.18	0.14	**0.03**
tai60b	60	71	0.11	0.05	**0.00**
sko72	72	123	**0.03**	0.42	0.09
tai80a	80	166	0.55	**0.14**	1.62
sko81	81	177	**0.02**	0.27	0.09
sko100a	100	328	**0.03**	0.19	0.12
sko100b	100	328	**0.01**	0.26	0.08
tai100a	100	328	0.57	**0.11**	1.64
wil100	100	328	**0.01**	0.18	0.05

Table 3: Comparison between *RoTS*, *RTS*, and *SA* on selected QAPLIB instances. Best found results are indicated in bold font

[3] J. K. Hao and J. Pannier. Simulated annealing and tabu search for constraint solving. In E. Boros and R. Greiner, editors, *Electronic Proceedings of the Fifth International Symposium on Artificial Intelligence and Mathematics, Fort Lauderdale, Florida, USA*, 1998.

[4] T. Stützle and S. Fernandes. New benchmark instances for the QAP and the experimental analysis of algorithms. In J. Gottlieb and G. Raidl, editors, *EvoCOP 2004*, volume 3004 of *LNCS*, pages 199–209, Berlin, Germany, 2004. Springer.

[5] É. D. Taillard. Robust taboo search for the quadratic assignment problem. *Parallel Computing*, 17(4–5):443–455, 1991.

The Effect of Filtering on the Uniform Random 3-SAT Distribution

Andrew M. Sutton, Adele E. Howe, and L. Darrell Whitley
Department of Computer Science,
Colorado State University, Fort Collins, CO, USA
{sutton,howe,whitley}@cs.colostate.edu

Abstract

Stochastic local search algorithms are among the state-of-the-art for solving 3-SAT problems. Computational studies for 3-SAT typically involve examining the behavior of solvers on instances drawn from a uniform random problem distribution as the *constrainedness* (measured as the ratio of clauses to variables) is varied. However, most studies for local search are performed on *filtered* problem distributions in which unsatisfiable instances have been removed. We show that by filtering the problem distribution, statistical trends are introduced at the critical point which subsequently intensify into the overconstrained phase. The presence of these biases implies that filtered problem distributions have significant statistical deviations from uniform random problem distributions. We conjecture these biases may be partially responsible for the observed local search easy-hard-easy pattern which is distinct from the analogous phenomenon detected in complete solvers.

1 Introduction

The Boolean 3-Satisfiability Problem (3-SAT) is a fundamental problem in the theory of computation. Generating hard benchmarks is of interest to both the experimental evaluation of solvers and the theoretical computer science community.

A 3-SAT formula F consists of the logical conjunction of a set of clauses, each containing exactly 3 literals in disjunction. The objective is to determine whether or not the formula is *satisfiable*, that is, ascertaining whether there exists an assignment to variables such that all clauses in the formula are simultaneously satisfied.

A 3-SAT formula with n variables and m clauses is generated uniformly at random by selecting, for each of the m clauses, exactly 3 distinct elements out of the set of n variables and negating each with probability 1/2. Let $\Xi(n, m)$ denote the set of all 3-SAT formulas with n variables and m clauses.

This uniform random generation process is equivalent to choosing m clauses uniformly at random (with replacement) from the $8\binom{n}{3}$ possible

clause configurations. This in turn is equivalent to choosing $F \in \Xi(n, m)$ uniformly at random [3]. In other words, generating an instance uniformly at random is identical to drawing an unbiased sample from $\Xi(n, m)$.

The constrainedness of an instance belonging to $\Xi(n, m)$ is given by a control parameter $\alpha = \frac{m}{n}$. This is the average number of constraints that each variable must satisfy. Low α values indicate underconstrained instances where high α values indicate overconstrained instances. If F is drawn uniformly at random from $\Xi(n, m)$ then the probability that F is satisfiable is a function of α. There is a *phase transition* between underconstrained and overconstrained phases where the probability that a formula is satisfiable drops from one to zero. This transition occurs around a critical value which has been empirically determined to be $\alpha \approx 4.26$. The hardest problems occur around this critically constrained value.

In many studies, probabilistic analysis can be employed if the assumption holds that F has been drawn uniformly from $\Xi(n, m)$. For example, an exact expression for the expected number of solutions can be derived if we assume the expectation is taken over all $\Xi(n, m)$.

Complete solvers (if given sufficient computation time) will halt for both satisfiable and unsatisfiable instances. However, a stochastic local search algorithm cannot, in principle, determine that an instance is unsatisfiable. Thus local search algorithms are typically evaluated using instances that are guaranteed to be satisfiable.

2 The filtering process

Forcing is one way to generate an instance that is guaranteed to be satisfiable. First a variable assignment is randomly selected at the beginning of the generation process. Then, clauses are subsequently generated and tested for compatibility with the selected assignment. A clause is accepted only if it is compatible with the assignment. After all m clauses are generated, the formula is guaranteed to have at least one satisfying assignment.

Forced instances tend to be much easier to solve by stochastic local search algorithms [1]. The conjecture is that the forced clauses possess some structural regularity that guides stochastic local search toward solutions.

To eliminate this bias, the *filtering* process attempts to select satisfiable instances at the instance level rather than the individual clause level. An instance is generated uniformly at random and tested for satisfiability using a complete solver. If the instance is unsatisfiable it is discarded and a new instance is generated.

Strictly speaking, the filtering process becomes unusable above the critical value as n approaches infinity and the satisfiability probability function approaches a step function. In this limit, instances above the critical value have a vanishing probability of being satisfiable and the filtered distribution

becomes essentially truncated. Thus filtering, as Achlioptas et al. [1] have pointed out, becomes limited for larger instances (to address this, they introduce a structured distribution which is significantly distinct from $\Xi(n, m)$).

This filtering process has been employed to generate standard benchmarks (e.g. many SATLIB benchmarks) as well as to identify and study an easy-hard-easy pattern for stochastic local search algorithms [2, 6, 4].

3 Trends that emerge at the critical point

Beyond the critical point, the number of "free" variables available for clauses reduces and more variable inter-dependencies arise. To produce satisfiable problems in this overconstrained phase, the increasing constraints must be offset by some structure of the interactions.

Due to the random uniform generation process, a variable should occur negated roughly the same number of times it appears unnegated. We can calculate the expectation of the absolute difference Δ between the number of times a variable appears negated and the number of times it appears unnegated. Let ω denote the number of times a particular variable occurs in the formula. Fix $\omega = i$. The *conditional expectation* of the absolute difference Δ between negated and unnegated occurrences given i total occurrences is

$$\mathrm{E}[\Delta|\omega = i] = \sum_{j=0}^{\lfloor i/2 \rfloor} \binom{i}{j} p^j (1-p)^{i-j} 2(i-2j) \tag{1}$$

Since p is the probability a variable appears negated, uniform random problems should have $p = \frac{1}{2}$. Now we characterize ω as a random variable. Over uniform random problems, ω is distributed binomially:

$$\Pr\{\omega = i\} = \binom{m}{i} \left[\frac{3}{n}\right]^i \left[1 - \frac{3}{n}\right]^{m-i} \tag{2}$$

since the probability of an arbitrary variable occurring in a clause is $\frac{3}{n}$. Thus the theoretical expectation of Δ taken over all possible values of ω is

$$\mathrm{E}[\Delta] = \sum_{i=0}^{m} \mathrm{E}[\Delta|\omega = i] \Pr\{\omega = i\} \tag{3}$$

We have discovered on filtered problems a significant divergence in a key coefficient that appears in a decomposition of the 3-SAT objective function [5]. This deviation begins at the critical point and suggests that the negation balance is changing in filtered instances.

To test this, we generated two benchmark sets. The *unfiltered* set consists of instances of 100 variables each sampled uniformly at random from $\Xi(100, m)$ varying m in such a way to obtain 391 values with α ranging

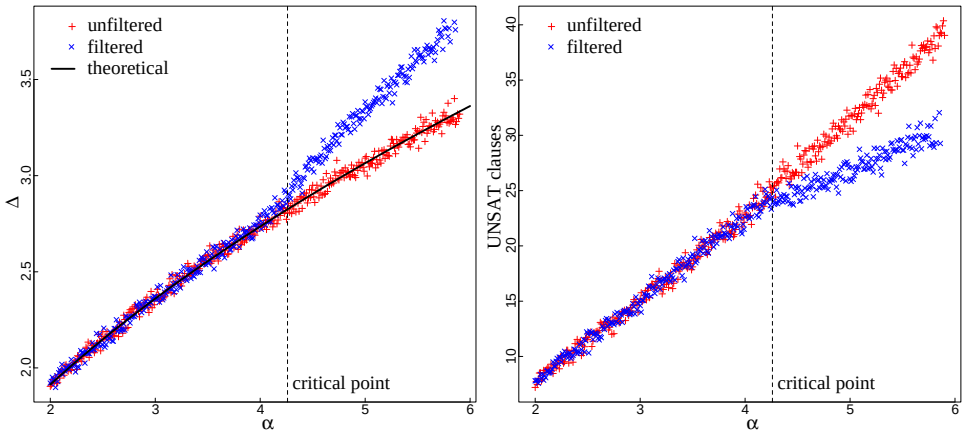

Figure 1: Unfiltered vs filtered sets: polarity delta [left] and `POLARITY-HEURISTIC` value [right].

from 2.00 to 5.90 in increments of 0.01. The *filtered* set consists of the same number of instances with the same α values but the generate-and-test procedure was employed to filter out unsatisfiable instances. We generated 50 instances for each value of α.

We report the average difference in positive vs. negative occurrence over all variables for each instance on each set in Figure 1. We compare this with the expectation computed in Equation (3). We see a deviation in the filtered problems emerge at the critical point. In other words, the filtering process is selecting instances that have variable negation ratios that are fluctuating away from expectation.

This structural regularity might be exploited by a stochastic local search algorithm if the polarity of a given variable tends to satisfy more clauses than if it were negated. The `POLARITY-HEURISTIC` is computed by simply setting each variable to true (false) if it appears more often unnegated (negated). We plot the resulting number of clauses satisfied for each instance under the `POLARITY-HEURISTIC` on the right in Figure 1.

4 Summary

We conjecture that filtering instances for satisfiability has the side-effect of selecting instances that have certain structural characteristics that cause significant statistical deviations from $\Xi(n, m)$. Stochastic local search algorithms may be able to exploit these characteristics, perhaps partially explaining the reduced computational effort noticed by others [2, 4] in the filtered overconstrained phase.

References

[1] D. Achlioptas, C. Gomes, H. Kautz, and B. Selman. Generating satisfiable instances. In *Proc. of AAAI 2000*, Austin, TX, 2000.

[2] D. A. Clark, J. Frank, I. P. Gent, E. MacIntyre, N. Tomov, and T. Walsh. Local search and the number of solutions. In *CP 1996*, pages 119–133, 1996.

[3] D. Coppersmith, D. Gamarnik, M. T. Hajiaghayi, and G. B. Sorkin. Random MAX SAT, random MAX CUT, and their phase transitions. *Random Structures and Algorithms*, 24(4):502–545, 2004.

[4] J. Singer, I. P. Gent, and A. Smaill. Backbone fragility causes the local search cost peak. *JAIR*, 12:235–270, 2000.

[5] A. M. Sutton, L. D. Whitley, and A. E. Howe. A polynomial time computation of the exact correlation structure of k-satisfiability landscapes. In *Proc. of GECCO 2009*, Montréal, Canada, 2009.

[6] M. Yokoo. Why adding more constraints makes a problem easier for hill-climbing algorithms: Analyzing landscapes of CSPs. In *CP 1997*, pages 356–370, 1997.

Weight Setting Strategies for Two-Phase Local Search: A Study on Biobjective Permutation Flowshop Scheduling

Jérémie Dubois-Lacoste
IRIDIA, CoDE, Université Libre de Bruxelles, Brussels, Belgium
jeremie.dl@gmail.com

Abstract

Two-Phase Local Search (TPLS) is a general algorithmic framework for multi-objective optimization. TPLS transforms the multi-objective problem into a sequence of single-objective ones by means of weighted sum aggregations. This paper studies different sequences of weights for defining the aggregated problems, in the bi-objective case. In particular, we propose two weight setting strategies that show better anytime search characteristics than the original weight setting strategy used in the TPLS algorithm.

1 Introduction

In this work we focus on the study of weight setting strategies for the Two-Phase Local Search (TPLS). Two-Phase Local Search has been shown to be an important part (with Pareto Local Search) of state-of-the-art algorithms for the bi-objective traveling salesman problem [5] and the bi-objective permutation flowshop scheduling problem (PFSP) [3]. We use the latter as our test problem for our experimental study. This extended abstract is structured as follows. We first introduce the TPLS, the bi-objective PFSP, and the performance assessment tools used to analyse the results. Then we present the different weight setting strategies.

Two-Phase Local Search. TPLS [6] is a general algorithmic framework for multi-objective optimization which consists in two main phases. The first phase uses an effective single-objective algorithm to find a good solution according to one objective. We slightly modified this first phase by using two initial solutions, for each one of the two objectives. The second phase solves a sequence of *scalarizations*, that is, weighted sum aggregations of the multiple objectives into a single scalar function. In TPLS, the best solution found by the previous scalarization is used as the initial solution for the next scalarization. The motivation for TPLS is to exploit the effectiveness of the underlying single-objective algorithm.

Bi-objective Permutation Flowshop Scheduling. The Permutation Flowshop Scheduling Problem (PFSP) is a well-known problem among the scheduling class. Our study here is based on previous work [3, 2] and the reader should refer to these for a complete description of the PFSP. We show here some results for the bi-objective PFSP with the objectives of makespan minimization and weighted tardiness minimization. However, the experimental results are also representative for those obtained on two other combinations: makespan and sum of flowtimes, and sum of flowtimes and weighted tardiness.

Performance assessment and experimental setup. Results are analyzed by examining the differences between the empirical attainment function (EAF), in order to identify in which region and by how much an algorithm performs better relatively to another one [4]. Given a pair of algorithms, the differences in favor of each algorithm are plotted side-by-side and the magnitude of the difference is encoded in grey levels. The bi-objective PFSP instances were generated following a similar procedure as for existing benchmarks [2, 3]. The experiments have been carried out using 50 scalarizations of 500 iterations each. All single-objective solutions have been obtained with the Iterated Greedy (IG) algorithm; initial solutions were obtained running IG for 1 000 iterations. EAFs have been determined using 100 independent runs of each algorithm.

2 Weight Setting Strategies

Single direction (*1to2* or *2to1*). The simplest way to define a sequence of scalarizations is to use a regular distribution of the weights from $(0, 1)$ to $(1, 0)$ in a given direction, either from the first objective to the other or vice versa. We call these alternatives *1to2* or *2to1*, depending on the direction followed. There are two major drawbacks of this strategy. First, the direction chosen gives a clear advantage to the starting objective, that is, the Pareto front approximation will be better on the starting side. This fact explains why these strategies are always the best ones for their starting sides. Second, to allow a fair computation effort in the different regions of the Pareto front, one needs to know in advance the computation time which is available to define appropriately the number of scalarizations.

Double strategy. This is a combination of *1to2* and *2to1*, where half of the scalarizations are defined sequentially from one objective to the other one, and other half in the other direction. This approach, proposed in [6], tries to avoid giving advantage to the starting objective, but it still requires to define the number of weights, and hence, the computation time, in advance.

Anytime strategy. We propose the *anytime* strategy, which does not require to define the number of scalarizations in advance. This strategy defines a new weight in the middle of the interval for two previous scalarizations.

Then for each of the two solutions obtained with the previous scalarizations that form the interval, the weighted sum value for the new weight is computed. The solution which is better for this new weight is taken to be the initial solution of the scalarization for the new weight. Given a weight vector $\vec{\lambda} = (\lambda_1, \lambda_2)$, where $\lambda_2 = 1 - \lambda_1$, the sequence of weights produced by the *anytime* strategy is $\lambda_1 = (1, 0, 0.5, 0.25, 0.75, 0.125, 0.375, 0.625, 0.875, \dots)$. Moreover, for each level of depth, the scalarizations can be solved in a random order, for instance one can have half a chance to start with 0.25 or with 0.75. This strategy allows to stop the search at any time and insures that the computation effort has been fairly distributed along the Pareto front. This strategy appears to have a different behavior of *1to2*, *2to1*, and *double* by performing differently in different regions of the Pareto front. Hence, it is not easy to compare the respective quality of the outcomes. However, one can say that the quality of *anytime* strategy is roughly similar to the other ones.

Dichotomic scheme. A dichotomic scheme does not define the weights sequence in advance, but determines the weights according to the solutions already found. The aim is to fit automatically the resolution to the Pareto front shape (more precisely, the approximation of it which is known at a given time).

The dichotomic scheme for exact resolutions was proposed by Aneja and Nair [1] and was adapted recently for the approximate case by Lust and Teghem [5]. However we found that, at least for the bi-objective PFSP, the acceptance criterion used in this latter paper is not the best one. Indeed in their criterion a solution is accepted only if it is inside the triangle formed by the two initial solutions and their ideal point, and some non-dominated solutions can be rejected (for instance, a very good new solution which dominates the ideal point). We found that another criterion, more permissive, is more effective. We call s_1 the initial solution which have the higher value for the second objective and s_2 the other one. With our permissive criterion, the new solution is accepted if at least one of these conditions is true:

1. $Objective_1(newSol) < Objective_1(s_1)$
2. $Objective_2(newSol) < Objective_2(s_2)$
3. $newSol$ is on the left side of the segment $[s_2, s_1]$

Moreover we propose a new way to define the scalarizations, by solving two scalarizations for each step instead of one. Both scalarizations are defined by the same weight, but each one uses a different initial solution as a seed for the underlying single-objective algorithm. That is, for each recursive step, two new solutions are found (if both are accepted), then three new recursive steps are defined instead of two in the original version. During the resolution, a solution that appears to be dominated is replace by the better solution.

This scheme is proposed specifically for approximate problems for which use one of the previously found solutions to seed the underlying single-objective algorithm leads better results than use an heuristic or random seed. This scheme appears to improve the effectiveness of TPLS for the PFSP. We give in Figure 1 a comparison of this dichotomic scheme with the other strategies.

3 Conclusion

We presented two weight setting strategies that do not require defining an a priori number of weights. This feature is useful in practical situations when the available computation time is not known in advance. The *anytime* strategy utilizes a regular distribution of weights, whereas the *dichotomic* strategy tries to adapt the set of weights to the shape of the Pareto frontier. The latter is shown to be more effective, at least for the PFSP, than the other strategies examined. Further research should test whether this result generalizes to other multi-objective problems.

References

[1] Y. P. Aneja and K. P. K. Nair. Bicriteria transportation problem. *Management Science*, 25(1):73–78, 1979.

[2] J. Dubois-Lacoste. A study of pareto and two-phase local search algorithms for biobjective permutation flowshop scheduling. Master's thesis, IRIDIA, Université Libre de Bruxelles, Brussels, Belgium, 2009.

[3] J. Dubois-Lacoste, M. López-Ibáñez, and T. Stützle. Effetive hybrid stochastic local search algorithms for biobjective permutation flowshop scheduling. In *Proceedings of HM2009*. Springer-Verlag, 2009. to appear.

[4] M. López-Ibáñez, L. Paquete, and T. Stützle. Exploratory analysis of stochastic local search algorithms in biobjective optimization. Technical Report TR/IRIDIA/2009-015, IRIDIA, Université Libre de Bruxelles, Brussels, Belgium, May 2009.

[5] T. Lust and J. Teghem. Two-phase pareto local search for the biobjective traveling salesman problem. *Journal of Heuristics*, 2009. To appear.

[6] L. Paquete and T. Stützle. A two-phase local search for the biobjective traveling salesman problem. In C. M. Fonseca et al., editors, *Proceedings of EMO 2003*, volume 2632 of *LNCS*, pages 479–493. Springer Verlag, 2003.

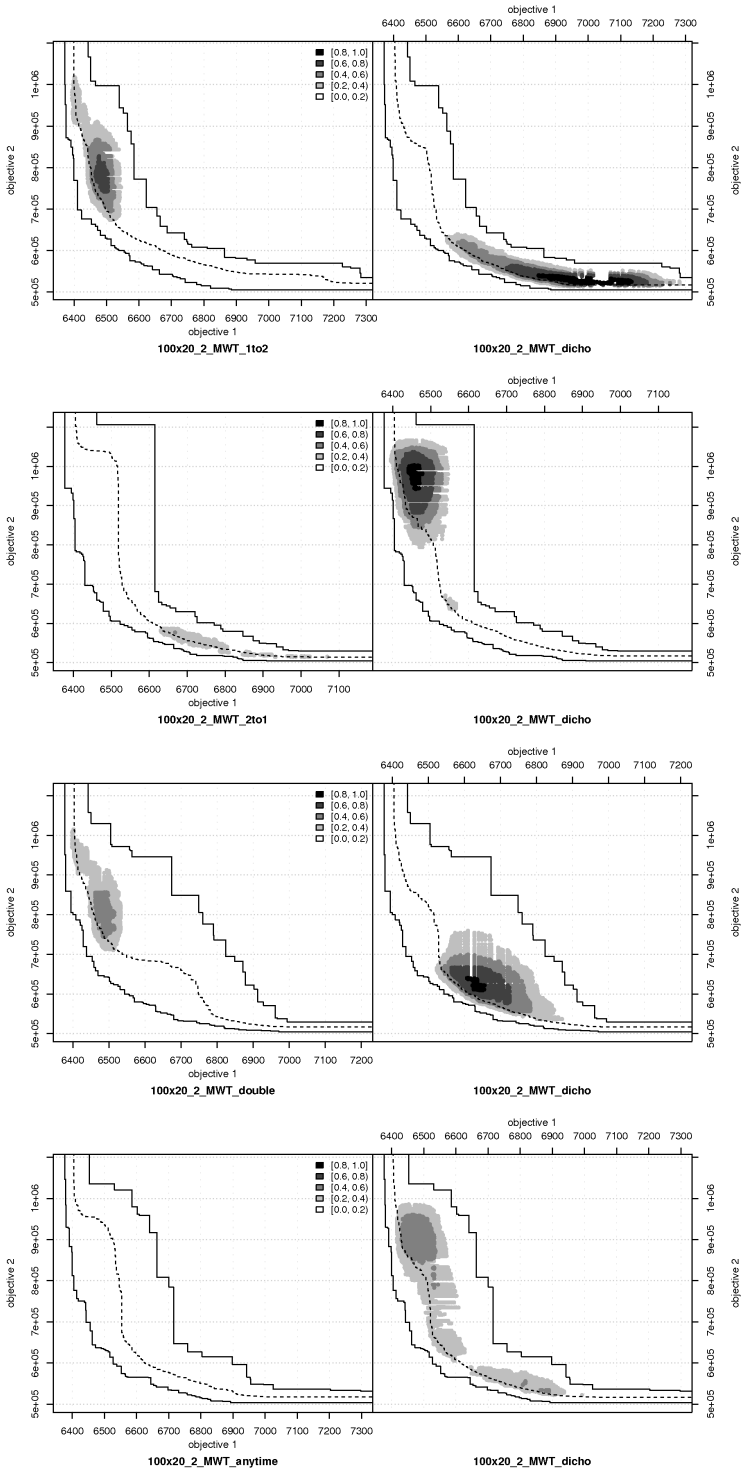

Figure 1: Dichotomic strategy (on the right) against (from top to bottom): *1to2*, *2to1*, *double*, *anytime*. Objective 1 is makespan minimization.

Authors Index