by Evgeny Selensky August 31, 2003

Problem Reformulation and Search. Project History

Goals

The aim of the project is to investigate what happens when we change the representation of problems. For example, we can represent a vehicle routing problem (routing vehicles to pick up goods) as a factory scheduling problem: i.e. we represent vehicles as machines on the shop floor, visits as operations to be performed, and the travel distances as transition (setup) times between operations. What representation will be best, and under what circumstances? This change of representation is a *change of representation in the large.*

We can also have *changes of representation in the small.* For example, we can use logically equivalent encodings of a problem instance.

There are some obvious reformulations, such as graph colouring as a problem of partitioning into independent sets. And there are some less obvious reformulations, such as the execution of a program (e.g. Gale-Shapley) into applying arc consistency to a suitable representation.

Reformulation in the large: vehicle routing vs. shop scheduling

A big part of the project [15] concerned changing representations of *vehicle routing* and *job shop scheduling* problems with the goal of being able to give practical recommendations as to what solving technology is best to use when dealing with real-world problems having characteristics of both problem classes.

First, we observed that standard solving technologies in one domain (e.g. systematic branch and bound based search with powerful constraint propagation in the shop scheduling domain and local search techniques for routing problems) had noticeably different performance when applied to instances from the other problem class [16].

As a second step, we tried to develop an understanding of why this happens. We *a priori* identified some problem parameters that might be responsible for the performance difference [2]. These parameters are: the number of alternative resources (big for VRP and small for JSP), the number of complex temporal constraints between operations (small for VRP and big for JSP), the optimization criterion (minimize travel for VRP vs. minimize makespan for JSP), the ratio of operation duration to transition time (small for VRP, big for JSP) and the amount of temporal slack (i.e. how tight time window constraints are in an instance).

We took a standard benchmark library (due to M. Solomon), extended it and performed a series of experiments, to start with, specifically focused on the ratio of operation duration to transition time. The results of our experiments suggested that distance was a crucial factor when VRPs were solved with a standard VRP technology (guided local search) and that when reformulated as shop scheduling and solved with a standard scheduling technology (systematic branch and bound search with domain specific consistency techniques) distance was more important for problems with structure and relaxed time window and capacity constraints.

Having analysed the results we came to the conclusion that to develop a deeper understanding of the influence of various problem characteristics on algorithmic behaviour we needed more control over the parameter values. In other words, we needed to produce our own data set and perform a thorough empirical study.

To this end, we developed a problem generator capable of producing mixed JSP-VRP instances with desired characteristics.

Having produced the testbed, we performed a large-scale experimental study on the response of algorithmic behaviour to variation of parameter values [4, 3]. The results showed that the most tangible response in solution quality by both solving technologies was brought about by change *in optimization criterion*, *the number of alternative resources* and *the number of complex temporal constraints between operations*. For example, we find that the removal of alternative resources in VRPs favours the scheduling technology; an increase in temporal constraints between operation in VRPs favours the scheduling technique and, symmetrically, a respective decrease in JSPs improves the performance of the routing technique; finally, changing the optimization criterion from total travel minimization to makespan minimization in VRPs notably favours the scheduling technology and changing the criterion from makespan minimization to total processing time minimization in JSPs favours the routng technology.

The remaining factors, such as the ratio of operation duration to transition time or temporal slack, influenced solution quality less than those above.

Other problems addressed

VRP vs. JSP: Theoretical Time Complexity Transitions from NP through P to NP

Varying the key problem parameters, we can explore entire problem space between vehicle routing and shop scheduling. Surprisingly, while moving in the VRP/JSP spectrum of problems from one extreme to the other we can have transitions from NP-hard to polynomially solvable problems and then back to NP. For example, if we remove precedence constraints from the job shop scheduling problem we enter the class P, given all the resource allocation constraints in the original JSP and no temporal transitions between operations. If we then remove resource allocation constraints we will be back in NP because it will be equivalent to partitioning operations on resources. Similarly, if we remove the resource allocation constraints from the JSP allowing an operation to be performed on any resource, we will have a polynomially solvable problem. If we further remove precedence constraints, we are back in NP since the resultant problem can be polynomially reduced to multidimensional bin-packing.

VRP vs. JSP: Transformations of associated graphs

Having noticed that specific problem transformations with respect to operation duration vs. transition time can influence solution quality in VRP and JSP, as an aside we introduced and theoretically analysed transformations of graphs associated with the core subproblems of vehicle routing and job shop scheduling [1, 2]. We believe this area of research may be interesting in its own right and deserves attention in the future, in particular, because transforming the graphs associated with the problem we can control the behaviour of the solving techniques.

Symmetry and Local Search

Another interesting implication of the reformulation research that is worth investigating is the influence of symmetry in resource allocation on search performance. Following [12], where a local search algorithm is presented for balanced incomplete block designs, we can, for example, expect symmetry to improve the performance of local search for the VRP.

To further develop an understanding of how local search quality can be affected by symmetry in the problem, we took another benchmark problem (the bridge building problem [11]), represented it as a routing problem and solved it using local and systematic search [6]. Then we gradually transformed the original problem by adding more resource allocation and operation precedence symmetries and compared the performance of local and systematic search techniques. It turns out that on the test problems induced by the original example problem, local search algorithms (guided local search) were favoured by the addition of operation precedence symmetries and at some point outperformed systematic search. In contrast, the addition of alternative resources (with or without operation precedence symmetries) proved more amenable to systematic search.

Logically Equivalent Encodings and Algorithmic Behaviour

We have also looked at how changes in problem representation in the small may influence algorithmic performance. We considered the maximal independent set problem and used logically equivalent encodings of the problem constraints and different software tools [13, 14]. We find first that different encodings of the same constraints have an impact on runtime across the tools applied. Secondly, the best encoding with respect to runtime for one tool is not necessarily the best for another tool. So before using “the best” algorithm for a given problem, we should always compare possible equivalent encodings of it; we should be even more cautious when we have to use off-the-shelf solving technologies and empirically test how small changes in the representation of a problem influence algorithmic behaviour.

Arc Consistency vs. Stable Marriage

As noted above, one can have algorithms in different problem classes that work similarly and produce similar results. As an example of reformulations of executions of a program two problem classes were considered, the stable marriage problem (SM) and the constraint satisfaction problem (CSP). It turns out [7] that the application of the extended Gale-Shapley algorithm [10] to the SM achieves the same result as the application of arc consistency [5] to the CSP which is a reformulation of the original stable marriage problem. In particular, the domains of the CSP variables coincide with Gale-Shapley lists that contain all stable matchings. [7] also demonstrates that all SM solutions can be obtained in a failure-free manner.

Finally, empirical studies [8, 9] present the first complete algorithm for the stable marriage with ties and incomplete lists formulated as CSP and the results of applying SAT techniques to stable marriage. The chosen SAT encodings of stable marriage prove amenable to standard SAT technology (Chaff).

References

1. J.C. Beck, P. Prosser, and E. Selensky. Graph transformations for the vehicle routing and job shop scheduling problems. In *1st Int. Conference on Graph Transformation (ICGT 2002), Barcelona, Spain*, 2002.
2. J.C. Beck, P. Prosser, and E. Selensky. On the reformulation of vehicle routing problems and scheduling problems. In *LNAI 2371, Proceedings of the Symposium on Abstraction, Reformulation and Approximation (SARA 2002)*, pages 282–289, 2002.
3. J.C. Beck, P. Prosser, and E. Selensky. An empirical study of mutual routing-scheduling reformulation. In *Proceedings of the post-CP03 Modelling and Reformulation Workshop*, 2003.
4. J.C. Beck, P. Prosser, and E. Selensky. Vehicle routing and job shop scheduling: What's the difference? In *Proceedings of the 13th International Conference on Artificial Intelligence Planning and Scheduling*, 2003.
5. E. Bessiere and Regin J.-C. Arc concistency for general constraint networks: Preliminary results. In *International Joint Conference on Artificial Intelligence*, 1997.
6. Selensky E. A reformulation of the bridge building problem as vehicle routing. In *Post-CP03 Modelling and Reformulation Workshop*, 2003.
7. I.P. Gent, R.W. Irving, D. Manlove, P. Prosser, and B.M. Smith. A constraint programming approach to the stable marriage problem. In *International Conference on Principles and Practice of Constraint Programming*, 2001.
8. I.P. Gent and P. Prosser. An empirical study of the stable marriage problem with ties and incomplete lists. In *15th European Conference on Artificial Intelligence*, 2002.

9. I.P. Gent and P. Prosser. Sat encodings of the stable marriage problem with ties and incomplete lists. In *5th International Symposium on Theory and Applications of Satisfiability Testing (SAT2002)*, 2002.
10. D. Gusfield and Irving R.W. *The Stable Marriage Problem: Structure and Algorithms.* MIT Press, 1989.
11. The Oz Constraint Programming Language Online Manual. http://www.mozart-oz.org/documentation/fdt/node48.html.
12. S. Prestwich. A local search algorithm for balanced incomplete block designs. In *Proceedings of the Joint Workshop of the ERCIM Working Group on Constraints and the CologNet area on Constrain and Logic Programming*, pages 110–122, 2002.
13. P. Prosser and E. Selensky. On the encoding of constraint satisfaction problems with 0/1 variables. In *Post-CP01 Modelling and Reformulation Workshop*, 2001.
14. P. Prosser and E. Selensky. A study of encodings of constraint satisfaction problems with 0/1 variables. In *Lecture Notes in Computer Science: Recent Advances in Constraints*, volume 2627, pages 121–131, 2003.
15. Problem Reformulation and Combinatorial Search Project Website. http://www.dcs.gla.ac.uk/pras/.
16. E. Selensky. On mutual reformulation of shop scheduling and vehicle routing. In *Proceedings of the 20th UK PLANSIG*, 2001.