A Reformulation of the Bridge Building Problem as Vehicle Routing*

Evgeny Selensky

Department of Computing Science, University of Glasgow, Scotland.
evgeny@dcs.gla.ac.uk

Abstract. We empirically study the effects of problem formulations and, more specifically, the influence of symmetries in the problem on algorithmic performance. We present a reformulation of the Bridge Building Problem as an instance of vehicle routing with time windows. We apply systematic and neighbourhood search to the original and reformulated problem. We also relax the original and reformulated problem by introducing additional symmetries and show how this impacts the performance of the solving techniques.

1 Introduction

Choosing a suitable problem representation has long been in the focus of AI research [13], and, more specifically, in constraint satisfaction [15], scheduling and routing [4, 5, 12]. Considerable attempts have been made to automate the process of finding "good" formulations [6]. Nonetheless, the issue of adequately formulating the given problem still seems too broad to embrace it in all entirety. E.g., it is known that symmetries in a problem formulation are undesirable for constructive search because a solver spends redundant effort in exploring symmetric states. However, recent work [10] has revealed that different neighbourhood search techniques may in contrast benefit from having symmetries in the problem representation. The intuition is that adding symmetry-breaking constraints transforms solution states into local minima. It appears that more in-depth research is needed to study possible effects of symmetries in the context of local search in particular in the scheduling and routing domains.

This paper presents an empirical study of the effects of changes in the formulation of a scheduling problem and of the influence of symmetries in the different problem formulations on algorithmic performance. In the study we use an example problem of constructing a five-segment bridge (the Bridge Building Problem, BBP). It was introduced in [3] and has since been widely accepted as a benchmark [9]. In the study, we first solve the problem by a systematic branch and bound based technique. As a second step, we reformulate the original BBP as an instance of vehicle routing and apply to the routing-reformulated instance a neighbourhood search technique specifically designed for routing. Finally, we

* This work was supported by EPSRC research grant GR/M90641 and ILOG SA.

inject various symmetries into the BBP, both direct and reformulated, and compare the performance of systematic and neighbourhood search techniques with and without additional symmetries.

The paper is organised as follows. Section 2 is a recap on the Bridge Building Problem. In section 3 we present two encodings of the BBP, the direct and reformulated as the vehicle routing problem. Section 4 explains how we relax the BBP by introducing symmetries. Section 5 presents the design and results of our experiments. Section 6 concludes the paper.

2 Bridge Building Problem

In the Bridge Building Problem and 2) we have to construct a five-segment bridge.

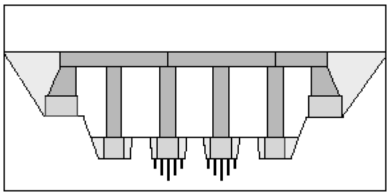

Fig. 1. The five-segment bridge (taken from the Oz online manual [9]).

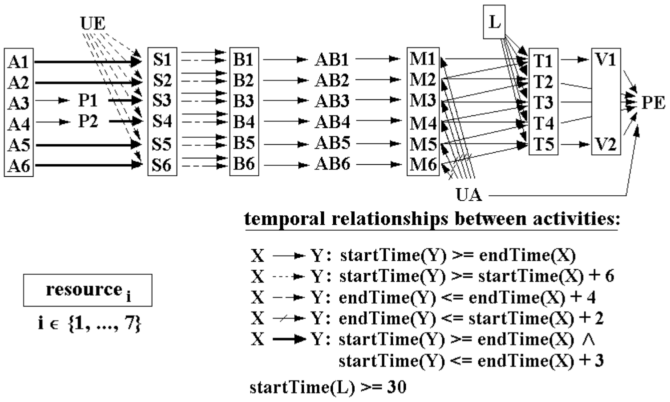

Fig. 2. The Bridge Building Problem: resources are shown as rectangles, activities as identifiers. Arrows represent temporal relationships between activities.

Table 1. The Bridge Building Problem: Specification

No.	ID	Description	Duration (days)	Predecessors	Resource
1	A1	excavation (abutment 1)	4	-	excavator
2	A2	excavation (pillar 1)	2	-	excavator
3	A3	excavation (pillar 2)	2	-	excavator
4	A4	excavation (pillar 3)	2	-	excavator
5	A5	excavation (pillar 4)	2	-	excavator
6	A6	excavation (abutment 2)	5	-	excavator
7	P1	foundation piles 2	20	A3	pile driver
8	P2	foundation piles 3	13	A4	pile driver
9	UE	erection of temporary housing	10	-	-
10	S1	formwork (abutment 1)	8	A1	carpentry
11	S2	formwork (pillar 1)	4	A2	carpentry
12	S3	formwork (pillar 2)	4	P1	carpentry
13	S4	formwork (pillar 3)	4	P2	carpentry
14	S5	formwork (pillar 4)	4	A5	carpentry
15	S6	formwork (abutment 2)	10	A6	carpentry
16	B1	concrete foundation (abutment 1)	1	S1	concrete mixer
17	B2	concrete foundation (pillar 1)	1	S2	concrete mixer
18	B3	concrete foundation (pillar 2)	1	S3	concrete mixer
19	B4	concrete foundation (pillar 3)	1	S4	concrete mixer
20	B5	concrete foundation (pillar 4)	1	S5	concrete mixer
21	B6	concrete foundation (abutment 2)	1	S6	concrete mixer
22	AB1	setting time (abutment 1)	1	B1	-
23	AB2	setting time (pillar 1)	1	B2	-
24	AB3	setting time (pillar 2)	1	B3	-
25	AB4	setting time (pillar 3)	1	B4	-
26	AB5	setting time (pillar 4)	1	B5	-
27	AB6	setting time (abutment 2)	1	b6	-
28	M1	masonry (abutment 1)	16	AB1	bricklaying
29	M2	masonry (pillar 1)	8	AB2	bricklaying
30	M3	masonry (pillar 2)	8	AB3	bricklaying
31	M4	masonry (pillar 3)	8	AB4	bricklaying
32	M5	masonry (pillar 4)	8	AB5	bricklaying
33	M6	masonry (abutment 2)	20	AB6	bricklaying
34	L	delivery of preformed bearers	2	-	crane
35	T1	positioning (bearer 1)	12	M1,M2,L	crane
36	T2	positioning (bearer 2)	12	M2,M3,L	crane
37	T3	positioning (bearer 3)	12	M3,M4,L	crane
38	T4	positioning (bearer 4)	12	M4,M5,L	crane
39	T5	positioning (bearer 5)	12	M5,M6,L	crane
40	UA	removal of temporary housing	10	-	-
41	V1	filling 1	15	T1	caterpillar
42	V2	filling 2	10	T5	caterpillar
43	PE	end of project	0	T2,T3,T4,V1,V2,UA	-

Table 2. The Bridge Building Problem: Additional Constraints

No.	Additional Constraint
1	$StartTime(S_i) \geq StartTime(UE) + 6,\ i \in \{1, ..., 6\}$
2	$EndTime(B_i) \leq EndTime(S_i) + 4,\ i \in \{1, ..., 6\}$
3	$StartTime(S_i) \leq EndTime(A_i) + 3,\ i \in \{1, ..., 6\}$
4	$EndTime(M_i) \leq StartTime(UA) + 2,\ i \in \{1, ..., 6\}$
5	$StartTime(L) \geq 30$

The complete project includes excavations, making foundation piles, formwork, masonry work, positioning of the bearers, etc. as displayed in Table 1 and explained in detail in [3, 9]. Each row in Table 1 shows activity number (column 1), identifier (column 2), description (column 3), duration (column 4), a list of other activities that should precede it (column 5) and finally the required resource (column 6).

We also have the following additional constraints that cannot be represented by showing the predecessor of an activity:

– The erection of the temporary housing must begin at least 6 days before each formwork (constraint 1 in Table 2).

- The time between the completion of the formwork and the completion of the corresponding concrete foundation is at most 4 days (constraint 2).
- Between the end of a particular foundation and the beginning of the corresponding formwork can at most 3 days elapse (constraint 3).
- The removal of the temporary housing can start at most 2 days before the end of the last masonry (constraint 4).
- The delivery of the preformed bearers occurs exactly 30 days after the beginning of the project (constraint 5 in Table 2).

3 Bridge Building Problem Formulations

3.1 Direct Encoding

The BBP is clearly a scheduling problem: we have to schedule 43 activities, 34 of which are allocated to the 7 available resources while the rest require no resource. The schedule must respect the temporal constraints displayed in Tables 1 and 2.

In the direct encoding there are 43 activities and 7 unary resources. Each unary resource can be occupied by only one activity at any instant. We post all the resource allocation and temporal constraints shown in Tables 1 and 2. For example, we state that activity $P1$ requires unary resource *pile driver* for its entire duration (20 days) and that *pile driver* can only start execution of $P1$ after *excavator* has finished execution of $A3$. While building a schedule we minimise the finish time of the latest activity PE, known as the makespan in the scheduling literature [1].

3.2 BBP as Vehicle Routing

Now we represent the Bridge Building Problem as an instance of the Vehicle Routing Problem.

In the *delivery* variant of the *vehicle routing problem* (VRP), m identical vehicles initially located at the base are to deliver discrete quantities of goods to n customers[1]. The locations of the base and customers are known. Each customer has a certain discrete demand for goods and each vehicle has a discrete capacity, i.e., it can carry quantities less than or equal to its capacity. A vehicle can make only one tour starting at the base (or the depot), visiting a subset of customers and getting back to the base. It is also assumed that travel time equals travel distance computed using the Euclidean metric. There are also temporal constraints known as *time windows* for servicing customers. Required is to find tours for a subset of vehicles such that (i) all customers are served, (ii) each customer is served only once and (iii) the total distance travelled by the fleet is minimum.

In the BBP reformulated as a routing problem, an activity is a customer visit, a resource is a vehicle. E.g., there is a vehicle corresponding to the *caterpillar*.

[1] In the *pickup and delivery* version, customers may require to pick up goods as well as to deliver.

This vehicle can only visit customers $V1$ and $V2$ and these visits should satisfy all precedence constraints involving $V1$, $V2$ and other visits. Clearly, the time window and duration of an activity is the time window and duration of the respective customer visit. Apart from that, we have to introduce two additional dummy visits for each vehicle to mark the start and end of its tour. Consequently, if in a solution a vehicle is not used, its empty tour will contain only two dummy visits. If a vehicle is used in a solution, any customer visit occurs after the start dummy visit and before the end dummy visit.

Travel between the depot and customers in the VRP corresponds to transition (setup, teardown) times between activities. Because we have zero transition times between activities in the BBP, all travel is also zero. So we can think of the BBP as a strange routing problem where everything occurs in one place! This representation of a scheduling problem is analogous to [5].

The end of project PE is a unique activity in this encoding. We do not represent it as a customer visit, but model it using a deadline variable D. To minimise the deadline, we iteratively reduce D during search. The cost function is the sum of tardiness on each vehicle, not the total distance as in the original VRP formulation.

4 BBP with Symmetries

As shown in [5], two important problem characteristics responsible for the performance of typical scheduling and routing techniques are the number of alternative resources in a problem and the number of temporal relationships between operations. Our long-term goal is to develop a deeper understanding of the influence of these parameters on solving techniques.

In this study, we can think of introducing two kinds of symmetries to the Bridge Building Problem, i.e., symmetries with respect to ordering activities and with respect to allocating resources. The first type of symmetry in the extreme turns a conventional scheduling problem into a variation of the open shop problem [8], where any order of jobs is satisfactory. The resource allocation symmetry amounts to having alternative resources. Examples of symmetries are given below in sect. 5.

In the next section, we first solve the original BBP as a scheduling problem and as a routing problem. Then we systematically vary the original Bridge Building Problem by adding symmetries in the problem formulation and observe how these variations in the problem influence algorithmic behaviour.

5 Experiments

5.1 No additional symmetries

The first set of experiments is to run the direct and routing-reformulated encodings of the Bridge Building Problem and compare the performance of solving

Table 3. Experiments with Scheduling Techniques. CPU time limit: 1 second

Techn.	Description	#Sols.	First Solution			Best Solution		
			Cost	Nodes	Backtr.	Cost	Nodes	Backtr.
ST1	Branch & bound + restart resource selection: min global slack activity selection: min earliest start time	4	120	27	1	104	90	2
ST2	Branch & bound without restart, the rest is as ST1	1	104	102	87	104	102	87

Table 4. Comparison of Routing Techniques. CPU time limit: 60 seconds

Technique	Description	Cost	
		First Sol.	**Best Sol.**
RT1	First sols. by savings, first-accept local search guided local search metaheuristic	123	120
RT2	First sols. by ST1, first-accept local search guided local search metaheuristic	120	116

techniques. The software we use throughout the paper is the ILOG optimisation suite (in particular, ILOG Scheduler 5.2 and Dispatcher 3.2).

Because every resource in the problem is unary and, consequently, at any instant it is occupied by no more than one activity, for any pair of activities A and B on the same resource, we have to decide if A goes before B or vice versa. Therefore as a scheduling technique, ST, for the direct encoding we use the approach of disjunctive scheduling [11], which is known to be efficient for this type of problem. At each step while we have activities to sequence, we dynamically choose the most critical resource to schedule activities on. Resource criticality is calculated in terms of global slack [2] as the difference between the demand and supply over a given time interval. Among the unscheduled activities on the selected resource, scheduled first is the activity that has the minimum earliest start time, breaking ties with the minimum latest start time. $ST1$ in Table 3 searches for solutions iteratively, posting constraints on the quality of each next solution and restarting search after each solution found. On the contrary, $ST2$ adds a new constraint on the quality of solutions at a leaf node and backtrack from there instead of starting search from scratch.

We see from Table 3 that $ST1$ improves on the overall number of search states explored (90 nodes and 2 backtracks) compared to $ST2$ (102 nodes and 87 backtracks) even though the very first solution obtained by $ST2$ is optimum (with a cost of 104 days). It means that propagation on the optimisation criterion helps the scheduling heuristics make better decisions than by $ST2$. Using either technique, the optimum solution was found within 0.1s CPU time (Windows NT workstation, 1 Gb RAM, 933 MHz Pentium III processor).

As a routing technique, RT, we apply a neighbourhood search based procedure to the BBP reformulated as vehicle routing. In $RT1$ (Table 4), the first solution is obtained by the savings heuristic [11]. In $RT2$, the first solution is imported from $ST1$. To improve solutions locally we use first-accept local search enhanced by the guided local search metaheuristic [14] to drive search out of local minima.

It has been found in [5] that impurities of the vehicle routing problem such as precedence constraints between visits or resource allocation constraints between visits and vehicles may deteriorate the performance of first solution construction heuristics (such as savings). It appears that it would have been the case for this problem as well if we had had more complex temporal relationships between activities, e.g., between activities on the same resource. When a routing technique cannot come up with a first solution, an alternative approach is to start search from solutions found by scheduling technology as is done by $RT2$.

It is clear from Tables 3 and 4 that the scheduling technique outperforms the routing technique for the Bridge Building Problem encoded as a VRP instance. This agrees with previous research in routing-scheduling reformulation. In [4, 5, 12] several problem parameters were identified that were responsible for the observed difference in algorithmic performance. Problems with a "scheduling" optimisation criterion (for example, makespan), many precedence constraints between operations and high specialisation of resources are amenable to typical scheduling techniques (for example, systematic branch and bound search, powerful global constraint propagation mechanisms and sophisticated heuristics). On the contrary, instances with a "routing" optimisation criterion (e.g., travel time), very few or no precedences between operations and many alternative resources for an operation benefit from using conventional routing techniques (fast and efficient first solution construction methods and local search in a metaheuristic framework).

5.2 Introducing symmetries to the BBP

Now we are transforming the BBP by relaxing temporal constraints between activities or adding alternative resources. Solutions to the transformed problems are therefore no longer solutions to the original BBP.

Precedence Symmetries In this section we systematically transform precedence constraints (Fig. 2) such that if in the original problem activity B could start only after activity A has finished execution ($A \rightarrow B$), now either A can precede B or vice versa, $A \leftrightarrow B$. In Table 5 we present the results of several experiments:

- PS_0: no additional symmetries;
- PS_1: precedence symmetries between operations B_i, AB_i, M_i, $i \in \{1, ..., 6\}$ to say that the order of these operations is no longer significant. Assuming

Table 5. Influence of Precedence Symmetries on Performance of Scheduling and Routing Techniques. CPU time limit: 60 seconds

Exp.	Technique	# added symms.	Cost		Cost Difference,%	
			First Sol.	Best Sol.	First Sol.	Best Sol.
PS_0	ST1	0	120	104	0	0
	RT1	0	123	120	2.5	15.4
PS_1	ST1	36	104	92	0	0
	RT1	36	107	95	2.9	3.3
PS_2	ST1	66	104	90	0	0
	RT1	66	95	95	-8.7	5.6
PS_3	ST1	126	-	-	-	-
	RT1	126	91	76	-	-

the operations do not overlap, it is equivalent to having $6 \cdot 3! = 36$ additional symmetries.

- PS_2: as in PS_1 plus symmetries between L and T_i, $i \in \{1, ..., 5\}$. The number of symmetries is $3! \cdot 2! \cdot 5 + 3! = 66$ because for each $i \in \{1, .., 5\}$ we have $3!$ permutations of B_i, AB_i, M_i, $2!$ permutations of L, T_i, and for $i = 6$ we have $3!$ permutations of B_6, AB_6 and M_6;
- PS_3: in addition to PS_2, symmetries between M_i and T_i, $i \in \{1, ..., 5\}$. The number of symmetries is $3! \cdot 2! \cdot 2! \cdot 5 + 3! = 126$.

From Table 5 we see that the performance of routing technology is improving relative to scheduling technology as we inject more symmetries. In the last experiment, the routing technology successfully found solutions while $ST1$ did not find any over the allocated CPU time.

Resource Allocation Symmetries Now we inject resource allocation symmetries into the original BBP to allow alternative resources (Table 6):

- RAS_0: no resource allocation symmetries (the same as PS_0);
- RAS_1: we introduce six sets of alternative resources such that B_i, AB_i and M_i, $i \in \{1, ..., 6\}$ are processed by any one resource in the i-th set[2]. There are $3^3 = 27$ different variants for allocating 3 activities on 3 alternative resources (not taking into account the fact that there may be other constraints in the problem that disallow a subset of these allocations). Because we have six sets of alternative resources, this gives 162 symmetries.
- RAS_2: in addition to RAS_1 we have resource allocation symmetries with respect to L and each of T_i, $i \in \{1, ..., 5\}$. This gives $6^6 = 46656$ symmetries in addition to RAS_1 (46818 in total).

As displayed in Table 6, the performance of the routing technique gets worse as we add in more resource allocation symmetries. More specifically, $RT1$ failed

[2] To do this we have to introduce extra resources.

Table 6. Influence of Resource Allocation Symmetries on Performance of Scheduling and Routing Techniques. CPU time limit: 60 seconds

Exp.	Technique	# added symms.	Cost		Cost Difference,%	
			First Sol.	Best Sol.	First Sol.	Best Sol.
RAS_0	ST1	0	120	104	0	0
	RT1	0	123	120	2.5	15.4
	RT2	0	120	116	0	3.3
RAS_1	ST1	162	98	92	0	0
	RT1	162	-	-	-	-
	RT2	162	98	98	0	6.5
RAS_2	ST1	46818	75	72	0	0
	RT1	46818	-	-	-	-
	RT2	46818	75	75	0	4.2

to produce first solutions in the experiments RAS_1 and RAS_2. That is why we had to resort to $RT2$ that imported first scheduler solutions. This is because the introduction of some resource allocation symmetries increases the possibility of the wrong resource choice by the first solution heuristic and, consequently, the possibility of subsequent failure. Finally, when we have substantially many alternatives the performance of the heuristic gets improved since increasingly more resource allocations will lead to solutions. Similar effects were noticed in [5].

Second, even $RT2$ could not find any improvements to the injected first solutions. One possible explanation is that in the presence of many temporal constraints a lot of moves in the neighbourhood are discarded. On the contrary, scheduling technology actually benefited from having many temporal constraints even with additions of resource allocation symmetries. Despite extra variants of operation placement introduced, temporal and resource constraint propagation was able to detect inconsistencies early on.

Precedence and Resource Allocation Symmetries We have also performed tests to assess the response of solution quality to the introduction of both types of symmetry at the same time. In total, we generated six different problems based on problems PS_1,PS_2,PS_3 from Table 5, and RAS_1 and RAS_2 from Table 6. As before, we have run branch and bound based systematic search algorithm ST_1 and, separately, local search algorithms RT_1 and RT_2 on each of the six problems. The experiments showed that RT_1 was not able to construct a first solution for any instance. Moreover, for all but one problems the first solution ST_1 found was optimal. For the remaining instance with precedence symmetries taken from PS_3 and resource allocation symmetries taken from RAS_2, even ST_1 could not find a solution over the allotted time. This remaining instance had the biggest number of symmetries of all the six problems.

5.3 Summary of Experimental Results

We observe that the introduction of precedence symmetries to the original problem instance leads to increasingly better performance of the local search technique as compared to the systematic search technique. Resource allocation symmetries, in contrast, have a negative influence on the performance of the local search technique because they make it harder to construct first solutions. These observations also hold when we introduce both types of symmetry at the same time.

6 Conclusion

This paper provides empirical evidence to emphasise the importance of problem formulation in the process of problem solving. More specifically, the study is focused on two aspects of problem reformulation: mapping an instance from one problem class to another and having symmetries in a problem representation.

Using a simple scheduling problem, we have seen that mapping to a different problem class can have a profound effect (positive or negative) on algorithmic performance. In particular, we have observed that a typical local search based routing technique performs worse on the Bridge Building Problem than a conventional systematic search based scheduling technique does. As [5] points out, it is important to realise what features a problem has as this can give an indication of what solving technique is best to use.

We could also view the results in this paper the other way around. Suppose that we had a "routing" problem with many predefined vehicle specialisations, precedences between customer visits and that we were interested in minimising the finish time of the latest tour. After all, we can have it in practice, for example, when certain trucks can only carry certain types of goods, trucks are loaded in a way that dictates some order of visits to minimise unloading delays, visits are time-critical and the only concern is to do all of them as soon as possible regardless of travel. This study provides evidence that the scheduling technique would be a much better alternative to the routing technology in this scenario.

We have also demonstrated that symmetries may have a dramatic impact on the performance of solving techniques. It is no surprise that, in general, symmetries are not desirable for systematic search [7]. However, local search techniques can benefit from having symmetries, as noted in [10]. In our study, when we added operation precedence symmetries, we observed a notable increase in the performance of the routing technique even for a problem of modest size such as the Bridge Building Problem. The introduction of resource allocation symmetries in isolation had a negative effect on the performance of the routing technique because it was not amenable to first solution construction and also because it proved hard for guided local search to even improve scheduling solutions in the presence of many temporal and resource constraints.

Based on a small set of test problems generated using a known scheduling benchmark, this study gives preliminary indications of the importance of issues of problem representation and symmetries for efficient problem solving. To develop a deeper understanding of their influence on search performance, in particular for scheduling problems, experiments need to be carried out systematically on larger sets of instances.

Acknowledgement

The author wishes to thank Patrick Prosser (Glasgow University, Scotland) and Paul Shaw (ILOG, France) for their ideas and comments.

References

1. Kenneth R. Baker. *Introduction to Sequencing and Scheduling*. John Wiley and Sons, 1974.
2. Ph. Baptiste, C. Le Pape, and W. Nuijten. *Constraint-Based Scheduling: Applying Constraint Programming to Scheduling Problems*. Kluwer Academic Publishers, 2001.
3. M. Bartusch. *Optimierung von Netsplanen mit Anordnungsbeiziehungen bei knappen Betriebsmitteln*. PhD thesis, Universitat Passau, Fakultat fur Matematik und Informatik, 1983.
4. 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.
5. 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 (ICAPS 2003), to appear*, 2003.
6. E. Fink. *Changes of Problem Representation: Theory and Experiments*. Physica-Verlag, 2003.
7. I.P. Gent, W. Harvey, and T. Kelsey. Groups and constraints: Symmetry breaking during search. In *Principles and Practice of Constraint Programming (CP'02)*, 2002.
8. T. Gonsalez and S. Sahni. Open Shop Scheduling To Minimize Finish Time. *Journal of the Association for Computing Machinery*, 23(4):665–679, 1976.
9. The Oz Constraint Programming Language Online Manual. http://www.mozart-oz.org/documentation/fdt/node48.html.
10. S. Prestwich. Negative Effects of Modeling Techniques on Search Performance. *Annals of Operations Research*, 18:137–150, 2003.
11. ILOG SA. ILOG Scheduler 5.1 User Manual, France, 2001.
12. E. Selensky. On mutual reformulation of shop scheduling and vehicle routing. In *Proceedings of the 20th UK PLANSIG*, 2001.
13. Yu. Smirnov and M. Veloso. Efficiency competition through representation changes: Pigeonhole principle versus linear programming relaxation. In *5th Int. Conference on Principles of Knowledge Representation and Reasoning (KR'96)*, 1996.
14. C. Voudouris and E.P.K. Tsang. Guided Local Search. *European Journal of Operational Research*, 113(2):80–110, 1998.
15. T. Walsh. Reformulating propositional satisfiability as constraint satisfaction. In *Symposium on Abstraction, Reformulation and Approximation (SARA)*, 2000.