On Mutual Reformulation of Shop Scheduling and Vehicle Routing *

Evgeny Selensky

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

Abstract. We present results of an empirical study of the reformulation of shop scheduling problems as vehicle routing problems, and symmetrically the reformulation of vehicle routing problems as shop scheduling problems. This study aims at establishing two lower bounds of performance of conventional scheduling and routing techniques on problem from both domains.

1 Introduction

It is known that changing problem representation can dramatically improve the solution process [1] - [4]. This is particularly important when solving large and hard combinatorial problems. This paper presents the results of an empirical study of mutually changing problem formulation between shop scheduling and vehicle routing. That is, we can take a routing problem, represent it as a factory scheduling problem, and solve it as a scheduling problem. Symmetrically, we take a factory scheduling problem, reformulate it as a routing problem, and solve it using a solution procedure specialised for routing problems.

Both these problems are industrially important and computationally challenging since they are known to be NP-complete [5] - [9].

The main objective of the study is to establish two extreme cases of performance of conventional scheduling and routing techniques on problem instances from each other's domains. This, in turn, will provide a setting for future investigations into the influence of problem formulation on instances enriched by side constraints.

In the study reported here we use two commercially available toolkits, tools specialised for shop scheduling and vehicle routing. We use these tools as they are designed to be used. That is, when reformulating a routing problem into a shop scheduling problem, we do not attempt to modify the toolkit in any way.

The paper is organised as follows. In section 2, the notions of the vehicle routing problem and shop scheduling problems are given. Section 3 presents the reformulated problems. In section 4 the tools, computational models and methods of solution are described. Section 5 outlines the experiments and presents the results. Section 6 concludes the paper.

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

2 The Vehicle Routing and Shop Scheduling Problems

In the *delivery* variant of the *vehicle routing problem* (VRP), m identical vehicles located at a 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 demand for goods and each vehicle has a certain capacity, i.e. it can carry quantities less than or equal to its capacity. A vehicle can make a tour starting at the base, 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.

Required is to find such tours for a subset of vehicles such that:

- all customers are served;
- each customer is served only once;
- each vehicle makes only one tour;
- the total distance travelled by the fleet is minimised.

In the formulation of the VRP studied here, temporal constraints for the processing of customers are included i.e. *time windows*.

In the *shop scheduling* problem (SSP), one is given $m \geq 2$ machines and n jobs. Each job consists of a number of operations to be processed by a specified machine for a given amount of time. Each machine processes no more than one job at a time, and each job is processed on at most one machine at a time. No *preemption* is allowed while processing any operation. That is, when an operation on a machine has been partly processed it cannot be postponed by another operation. In general, one can have non-zero setup times between operations. Required is to schedule operations on machines such that the maximum completion time among all the jobs (the *makespan*) is minimal.

In scheduling theory three basic shop models are considered [10]: the job shop, the flow shop and the open shop. In the *job shop scheduling problem* (JSSP), each job J_j consists of a specified *chain* of m_j operations. The *flow shop scheduling problem* is a special case of the job shop where each job must pass through machines in the same order and no machine is visited by a job more than once. The *open shop scheduling problem* (OSSP) differs from the flow shop in that, for each job, the order of its operations is not fixed in advance but must be chosen, different jobs being allowed to have different orders.

3 Reformulated Problems

In this study the shop scheduling and vehicle routing problems are mutually reformulated and solved by conventional techniques (Fig 1). Informally, when reformulating a VRP as a OSSP we represent each vehicle in the VRP as a machine on the factory floor in the OSSP. Each visit in the VRP is represented as an operation in the OSSP and the distance between visits corresponds to

[1] In the *pickup and delivery* VRP, the vehicles may also be required to pick up goods.

the set up (transition) time between operations. When reformulating a JSSP as a VRP, each machine in the JSSP corresponds to a vehicle in the VRP, and each operation corresponds to a visit that can only be performed by one specified vehicle. Operations that must be performed on a specific machine then correspond to a tour. The sequencing constraints between operations in a job then correspond to sequencing constraints between visits in different tours. Since there are no transition times between operations on a resource there are no distances between visits, and we must order visits to respect time windows.

In each reformulation one comparison is made:

– the performance of solving the VRP reformulated as a special case of the OSSP is compared to that of the VRP by local search [11];
– the performance of solving the JSSP reformulated as a special case of the VRP is compared to the performance of the JSSP solved by a scheduling technique.

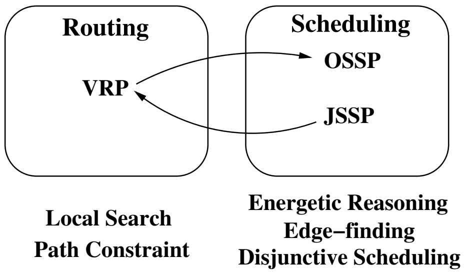


Fig. 1. Scheme of Reformulation

3.1 The VRP as the OSSP

In terms of the OSSP, the original VRP can be described as follows. Given are the number of machines m (vehicles) and the number of jobs n, each of which consists of only one operation (visit to a customer). Between operations on a machine there exist non-zero transition costs (distances between the respective customer visits). Each operation can be processed on any available machine available. That is, all the machines are *alternative* (the machines are said to be *multipurpose*). Required is to obtain a schedule with the minimal sum of transition costs on all machines. The order of operations on machines is immaterial. Unlike the ordinary open shop this problem has non-zero transition times, one operation per job (each operation has a unit duration) and alternative machines.

3.2 The JSSP as the VRP

The job shop scheduling problem represented as an instance of the vehicle routing problem is the following. Find such tours for each of the m vehicles that the following conditions are met:

- It is known beforehand which of the n customer visits are served by the given vehicles;
- There are predefined sequences of visits (jobs) performed by different vehicles; there are no limitations on the orders of visits in a tour;
- Distances between customers are all zeroes (as in the original JSSP there are no setups);
- The deadline for all tours should be minimal (minimise makespan)

4 Computational Models and Methods of Solution

It is the principle of this study that, while obtaining solutions to the given problem, the tools and methods available in an industrial setting are dealt with. For this reason commercial C++ libraries ILOG Solver 5.0, ILOG Scheduler 5.0 and ILOG Dispatcher 3.0 have been chosen [12].

Solver is a general and flexible constraint programming tool. Scheduler and Dispatcher are build on top of Solver and have a wide functionality in creating respectively scheduling and routing applications.

4.1 An OSSP model of the VRP

In the Scheduler encoding of the VRP, there are:

- m identical pairs of resources; each pair represents a machine/vehicle and consists of a unary resource (i.e., a resource whose capacity equals unit) and a discrete resource (with capacity greater than unit). Unit resources are present due to the fact that a machine can process only one operation at a time. Discrete resources are present to take into account capacity constraints;
- $n + 2m$ operations of unit duration: n visits to customers and $2m$ dummy operations in the beginning and end of each tour;
- $n + 2m$ resource constraints saying that whenever an operation requires a unary resource it should also require the respective discrete resource;
- setups between operations specified in an n x n square matrix;
- m precedence constraints saying that the start of each tour should precede all the visits in the tour;
- m precedence constraints saying that the end of each tour should be preceded by all the visits in the tour;
- $n + 2m$ resource constraints saying that the $2m$ dummy operations require specific resources (each such operation requires only one known pair of resources) and that n actual operations require any one of the m pairs of resources available.
- the objective is to minimise the sum of transition costs on all machines.

The search consists of the following steps:

1. For each pair of activities impose disjunctive constraints ranking them on alternative resources;
2. Select a resource for each activity from among alternative resources;
3. Assign start times for each activity in the schedule.

For this encoding limited discrepancy search (LDS) with a time limit is used [13].

4.2 The Default VRP Encoding

For comparison, the VRP has been solved by local search using the path constraints that accumulate cargo, time and distance [12], [14], encoded in Dispatcher. The Dispatcher model has as many vehicles and customers as there are in the problem. The cost function is Euclidean distance between customers.

The first solution is constructed using the savings heuristic and then improved by guided local search (GLS) [9].

4.3 A VRP Model of the JSSP

A machine here is represented as a vehicle. An operation is a visit to a customer. Unlike the ordinary VRP, now it is known beforehand which customers are served by a particular vehicle, because in the JSSP it is known which machine is required by an operation. The duration of an operation is the service time of a vehicle visiting the respective customer.

There are no distances between customers. The cost function is time instead of distance in the original VRP. To minimise the makespan in the JSSP, an additional variable, deadline d, is introduced to constrain the completion time of each tour. The first solution is generated by the savings heuristic and then improved by guided local search.

4.4 The Default JSSP Encoding

A encoding of the JSSP in Scheduler applies the following technique:

– A lower bound of the makespan is obtained by means of arbitrating potential resource conflicts; this is performed with an efficient heuristic which chooses resource constraints for ranking based on the least amount of propagation; the concept of disjunctive scheduling [15] is involved here;
– The upper bound is obtained by means of propagation;
– Binary search is performed with a slack-based heuristic starting from the previously found bounds [12], [16].

5 Numerical Results and Discussion

For experiments, a Windows NT workstation with an Intel Pentium III 933MHz processor and 1Gb RAM was used. The empirical study comprised two parts: experiments with the encodings of the VRP and JSSP encodings, respectively.

5.1 Experiments With the Encoding of VRP's as OSSP's

The testbed for these experiments is the one due to M. Solomon [17].

In Table 1 the results of quasi-complete depth-first backtracking search with an energetic reasoning based constraint propagation algorithm in Scheduler are shown. The duration of each experiment was equal to 3 hours. The discrepancy count was set to 5. During search, a constraint was posted that in each next solution the cost function value should be at least one unit less than in the previous one (minimisation is considered).

Experiments show that for VRP instances with time windows it is critical to rank operations imposing the disjunctive constraints [16] prior to assigning a resource from available alternatives. While in experiments whose results are presented below, such ranking has been introduced, it was not possible to obtain a single solution in the same time interval without the disjunctive ranking of operations.

Experiments also reveal that for the chosen benchmarks the critical heuristic was that for selecting a resource from alternative resources. The best solutions were obtained over the same time interval when choosing a resource with the minimal capacity available (which is in agreement with the general *fail first* principle).

In Table 1, the results of experiments are compared against those obtained using the Dispatcher encoding using guided local search with penalty factor 0.4. The duration of each experiment for the other VRP encoding was 3 hours as well. The first solution is constructed using the savings heuristic. The neighbourhood of a solution is the product of the following move operators. *TwoOpt*, *OrOpt*, *Relocate*, *Exchange* and *Cross* operators are applied to the current solution [9] in the order they are mentioned. In Table 1 the best known results are also shown [18].

Typical tours (for benchmarks R103 and R104) are presented in Fig. 2.

From Table 1 and Fig. 2, it can be seen that the Scheduler solutions obtained by quasi-complete search using the approach of section 4.1 do not compare very well against the best known results obtained by local search. The possible reasons are poor performance of search for this formulation of the problem (this is known as thrashing [8]) and lack of optimality reasoning in the used domain filtering algorithms. The problem is that when dealing with optimisation problems, conventional constraint logic programming techniques efficient for feasibility domain filtering improve only the upper bound of the cost function. A better strategy would be updating both bounds [19], [14].

5.2 Experiments With the JSSP's Encoded as VRP's

The data set is job shop problem instances of sizes 6x6 to 15x15 that can be found in [20] and [21].

The results are presented in Table 2. The same benchmark problems were solved using Dispatcher with GLS (section 4.3) and the scheduling algorithm described in section 4.4. The GLS parameters used for the first encoding are the following.

The first solution is built by the savings heuristic and then improved by GLS with penalty factor 0.4. The neighbourhood of a solution explored to improve it is the product of the following move operators: *TwoOpt*, *OrOpt*, *Relocate* and *Exchange*.

In the experiments with those benchmarks for which the scheduling algorithm described in section 4.4 obtained optimal solutions, the time limit was set to 60 seconds, however the actual amount of CPU time necessary for it to obtain them was much less (see values in parentheses in Table 2). For the rest benchmarks the time limit was 180 seconds.

Experiments show that the solutions obtained by Dispatcher with GLS compare poorly against the results by the Scheduler toolkit. The comparison is even worse as the problem size grows. It is also a natural result that the solution improvement by guided local search for the 'rectangular' instances, i.e. where the number of jobs exceeds the number of machines (Lawrence10x5 and Lawrence15x10), is less than that of 'square' instances because 'rectangular' problems are harder to solve [5].

6 Conclusion

In this paper, the results of the mutual reformulation of the vehicle routing problem and the shop scheduling problems are presented. The purpose of this empirical study is to show the two extreme cases of performance of the conventional routing and scheduling techniques on problem instances from each other's domains.

Future studies will be devoted to investigating what happens when we enrich the VRP such that it has characteristics more like the JSSP, and symmetrically when the JSSP is gradually relaxed such that it becomes increasingly *open* and more like the VRP. Will we reach a class of scheduling (routing) problem that is amenable to VRP (scheduling) technology? We expect this will be the case.

References

[1] Bernard A. Nadel, Representation Selection for Constraint Satisfaction: A Case Study Using n-Queens, IEEE Expert, pages 16-23, June 1990.

[2] Eugene Fink, Systematic Approach to the Design of Representation-Changing Algorithms, In Proceedings of the Symposium on Abstraction, Reformulation, and Approximation, pages 54-61, 1995.

[3] Rainer Weigel, Christian Bliek and Boi V. Faltings, On Reformulation of Constraint Satisfaction Problems (Extended Version), SARA98, 1998.

[4] Eugene C. Freuder and Daniel Sabin, Interchangeability Supports Abstraction and Reformulation for Constraint Satisfaction, SARA95, 1995.

[5] J. Blazewicz, W. Domschke, E. Perch, The job shop scheduling problem: Conventional and new solution techniques, Invited Review, EJOR 93 (1993) 1-33.

[6] D. Applegate, W. Cook. A Computational Study of the Job-Shop Scheduling Problem. ORSA Journal on Computing, 3(2):149-156, 1991.

[7] W.P.M.Nuijten, Time and Resource Constrained Scheduling, A Constraint Satisfaction Approach, Thesis, Tilburg, the Netherlands, 1994.

[8] Edward Tsang, Foundations of Constraint Programming, Academic Press, 1993.

[9] P.Kilby, P.Prosser, P.Shaw, Guided Local Search for the Vehicle Routing Problem, in *Proceedings of the 2nd International Conference on Metaheuristics*, July 1997.

[10] Bo Chen, Vitaly A. Strusevich, Approximation Algorithms for Three-Machine Open Shop Scheduling, ORSA Journal on Computing, 5(3): 321-326, 1993.

[11] Meta-Heuristics. Advances and Trends in Local Search Paradigms for Optimization. Stefan Voss, Silvano Martello, Ibrahim H. Osman, Catherine Roucairol (Ed.), Kluwer Academic Publishers.

[12] ILOG SCHEDULER 5.0 Manuals. July 2000, ILOG S.A., Gentilly, Cedex, France.

[13] William D. Harvey, Matthew L. Ginsberg, Limited Discrepancy Search. IJCAI, 1995.

[14] Filippo Focacci, Solving Combinatorial Optimization Problems in Constraint Programming, PhD thesis, Università di Ferrara, 2000.

[15] E. Neron, P. Baptiste, J. Gupta, Solving Hybrid Flow Shop Problem Using Energetic Reasoning and Global Operations, March 29, 2000.

[16] P. Baptiste, C. Le Pape, A Theoretical and Experimental Comparison of Constraint Propagation Techniques for Disjunctive Scheduling.

[17] M. Solomon, Benchmarks for the vehicle routing problem with time windows, 1987. *www.cs.strath.ac.uk/research/GreenTrip/NewBest/SolProblems/*

[18] The Green Trip Project, *www.cs.strath.ac.uk/research/GreenTrip/*

[19] F. Bosi and M. Milano, Enhancing CLP Branch and Bound Techniques, in: *Software: Practice and Experience*, **Vol.31**, No.1, January 2001, pp.17-42.

[20] Job shop problem instances, *http://www.ms.ic.ac.uk/jeb/pub/jobshop1.txt*

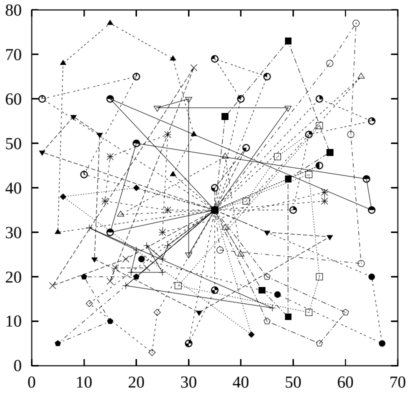
[21] Job shop problem instances, *http://www.dcs.gla.ac.uk/pras/resources.html*

Benchmark	LDS		GLS		Best Known Solutions	
	Cost	Number of Vehicles	Cost	Number of Vehicles	Cost	Number of Vehicles
R103	2445	22	1216.59	14	1292.68	13
R104	2664	20	995.04	11	1007.31	9
R107	2523	20	1080.43	11	1104.66	10
R108	2767	20	947.493	10	963.99	9
R109	2211	16	1156.32	12	1197.42	11
R110	2381	18	1091.6	12	1135.07	10
R111	2496	19	1062.03	12	1096.73	10
R201	3067	8	1162.76	9	1254.09	4
R204	2778	8	856.103	8	867.33	2
R205	3128	8	1146.54	8	998.72	3
R209	2997	8	934.05	8	923.96	3
R210	3041	8	940.675	8	963.37	3
RC103	2646	18	1329.21	12	1261.67	11
RC104	2418	18	1174.87	11	1135.48	10
RC107	2551	17	1262.33	13	1230.48	11
RC201	2814	9	1324.69	10	1406.94	4
RC203	2777	9	1092.67	9	1068.07	3
RC204	2622	9	1014.27	9	803.90	3
RC205	2925	9	1241.89	9	1302.42	4
RC206	2928	9	1174.00	9	1156.26	3
RC207	2758	9	1122.60	9	1074.10	3

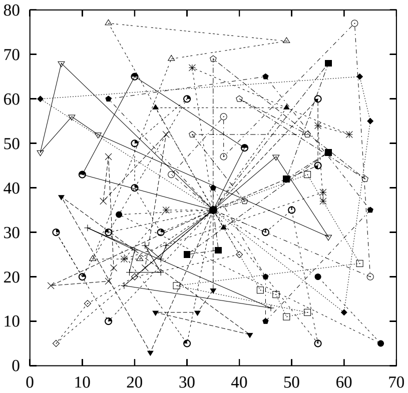
Table 1. Comparison of results obtained for the reformulated VRP model by Scheduler using LDS and those for the Dispatcher model using GLS to the best known results

Benchmark	Scheduling technique			Routing technique		
	Cost of First Solution	Cost of Best Solution	Time limit,s	Cost of First Solution	Cost of Best Solution	Time limit, s
Thompson 6x6	57	55*	60 ($\leq$1)	165	114	60
APES group 7x6	58	57*	60 ($\leq$1)	183	126	60
Lawrence 10x5	676	665*	60 ($\leq$1)	2306	2306	60
Adams 10x10	1321	1234*	60 (24)	6363	6055	60
Lawrence 15x10	1257	1121	180	6067	5854	180
Lawrence 15x15	1367	1287	180	9084	8589	180

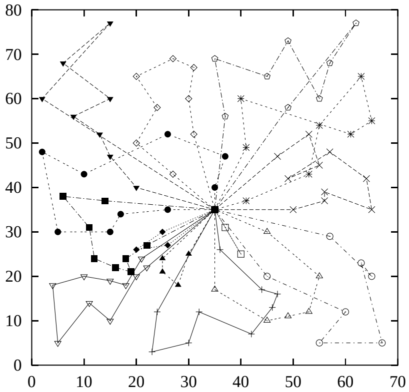
Table 2. Results of experiments with the JSSP encodings. An asterisk (*) after the cost value means the solution is optimal, values in parentheses are the actual CPU times to obtain the optimal solutions



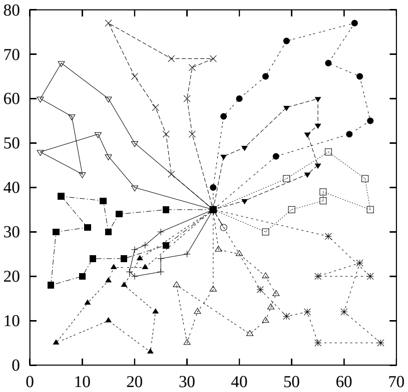
(a) Benchmark r103, 22 vehicles used



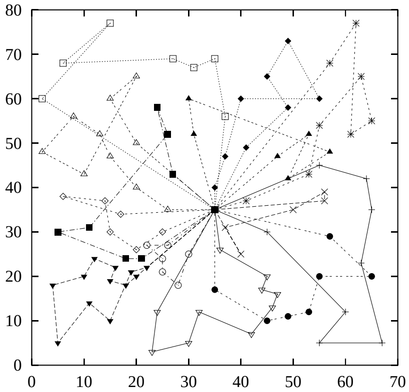
(b) Benchmark r104, 20 vehicles used



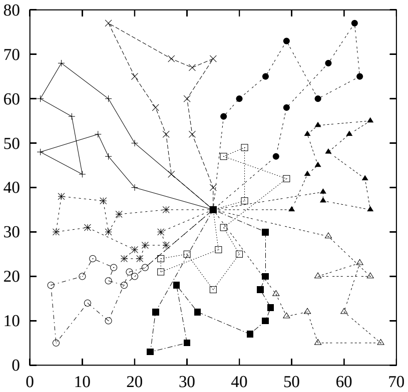
(c) Benchmark r103, 14 vehicles used



(d) Benchmark r104, 11 vehicles used



(e) Benchmark r103, 13 vehicles used



(f) Benchmark r104, 9 vehicles used

Fig. 2. The best tours obtained in 3 hours by Limited Discrepancy Search with discrepancy count equal to 5 (a, b), Guided Local Search with penalty factor set to 0.4 (c, d), and the best known tours from Green Trip project (e, f). Points represent locations of customers on the (x,y)-plane, lines parts of respective tours