J Sched (2006) 9:469–491
DOI 10.1007/s10951-006-8596-4

A case study of mutual routing-scheduling reformulation*

J. Christopher Beck · Patrick Prosser · Evgeny Selensky

Abstract Classical models of combinatorial problems, such as scheduling, play a key role in optimization research as they allow theoretical and empirical work to focus on core issues of problem solving performance. By their very nature, however, classical models are a simplification of real-world problems. We argue that the challenge now is to address those real-world features that were simplified away in the classical models, and that in order to do this we should investigate how problem features affect solution technologies. In this paper, we perform an empirical study of vehicle routing problems (VRPs) and job shop scheduling problems (JSPs) using commercially available constraint-based optimization software, ILOG Dispatcher, and ILOG Scheduler. We start with instances of the classical VRP and JSP and systematically make the two problems more realistic by removing the simplifying assumptions of the classical models. While doing so, we empirically investigate the key problem characteristics that make the problems more amenable to one solution technique or the other. We[-5pc] argue that our observations are symptomatic of the underlying technologies used in the employed software.

Keywords job shop scheduling, vehicle routing, problem reformulation, empirical analysis

* This work was supported by EPSRC research grant GR/M90641, by Science Foundation Ireland under Grant 00/PI.1/C075, and by ILOG, SA.

J. C. Beck (✉)
Department of Mechanical & Industrial Engineering, University of Toronto, Canada
e-mail: jcb@mie.utoronto.ca

P. Prosser
Department of Computing Science, University of Glasgow, Scotland
e-mail: pat@dcs.gla.ac.uk

E. Selensky
Vidus Limited, Ipswich, Suffolk, UK
e-mail: eselensky@vidus.com

1. Introduction

It is a common practice to model new problems as instances of an existing classical problem (Walsh, 2000). For example, if a problem can be modeled as a graph coloring problem or even better a linear program, there exist techniques that are very likely to be able to solve the problem reasonably well. However, there are likely to be aspects of the real problem that do not fit easily into a simplified version of a known combinatorial optimization problem. The decision must then be made as to what level of simplicity is appropriate in the model, i.e. if we should ignore or take into consideration the specific problem characteristics. The drawback of the simplification approach is that it is not clear that solutions to the simplified model are solutions to the real problem. The opposite approach, where the complicating factors are taken into consideration, also presents challenges, due to the fact that it is difficult to predict the effect of removing a simplifying assumption on the performance of standard solution methodologies. It can be that what looks like a minor assumption in the model significantly changes the performance of standard problem solving techniques.

The Artificial Intelligence (AI) and Operations Research (OR) communities have developed efficient problem solving techniques for a variety of models (e.g., job shop scheduling problem (JSP), resource constraint project scheduling problem). However, it is very rare that we can formalize practical scheduling problems as JSPs as there are typically additional constraints to deal with such as transition times between tasks, multiple (contradictory) optimization criteria, and resource alternatives. Furthermore, although a problem may look like a classical scheduling problem, perhaps some characteristics imply that vehicle routing or hybrid techniques may be a better choice of solution technology.

This paper addresses the relationship between problem characteristics and search performance by comparing the performance of scheduling and vehicle routing solution techniques in nonclassical problem models. We perform an in-depth empirical study by systematically transforming classical vehicle routing problems (VRP) such that they become increasingly similar to classical scheduling problems and, symmetrically, by transforming classical JSPs so that they become similar to VRPs. We then solve these problems using commercially available routing and scheduling software, to discover what problem characteristics most influence the search performance. Given the success of constraint programming (CP) (Baptiste, Le Pape, and Nuijten, 2001; Davenport and Beck, 1999; DeBacker et al., 2000; Kilby, Prosser, and Shaw, 2000; Laborie, 2001; Nuijten, 1994; Smith and Cheng, 1993; Voudouris and Tsang, 1998), our study focuses on constraint-based technologies for VRP (ILOG Dispatcher) and scheduling problems (ILOG Scheduler). Naive as it may at first seem, it is a crucial point in our study that we are using "out-of-the-box" commercially available technology. We use the available software products for the following reasons:

- This software represents the state-of-the-art in commercial technology from the fields of AI, CP, and OR. We are therefore conducting our study with what is available to practitioners interested in solving problems rather than in inventing a new optimization technology.
- These software packages are implementations of techniques that are well-studied in the academic literature. Therefore, the results of an empirical study using commercial software are generalizable to the extent of the type of methodology realized in those products. In particular, one could use systematic search with global constraints and powerful propagation mechanisms (as in the scheduling software) or constraint-based local search in a meta-heuristic framework (as in the VRP product). Our empirical study gives us an appreciation of the effects of problem structure on those underlying technologies.

– Commercial solutions are a natural choice for many practitioners. Ours is a pragmatic study, resulting in what we hope is practical advice for those users that intend to use a software product without deep modifications and customization.
– Despite the commercial success of CP technology, it has been pointed out in the research community that the technology is difficult for nonexperts to use for nonacademic problems (Puget, 2004). In addition to providing advice to the nonexpert, our approach also provides a case study of the difficulties that such users may experience as they use commercial software to model and solve problems that are just slightly different from classical models.

We will discuss these points and, in particular, the issue of generality of our results given the use of commercial software in Section 7.

This paper is organized as follows. In Section 2, we define vehicle routing problems and scheduling problems. In Section 3, we show how we can reformulate VRPs as scheduling problems, and vice versa. In Section 4, we identify several structural characteristics that we believe differentiate routing problems from scheduling problems. Section 5 describes the design of our experiments to investigate the effect of these problem characteristics on the relative performance of the routing and scheduling techniques. The results of these experiments are reported in Section 6, followed in Section 7 by a discussion of the generality of our study, and finally a conclusion.

2. Problem definitions: VRP & JSP

In the capacitated vehicle routing problem with time windows, m identical vehicles initially located at a depot are to deliver integer quantities of goods to n customers. Each customer has a demand for goods that must be satisfied by a single vehicle and each vehicle has a finite capacity. A vehicle can make only one tour starting at the depot, visiting a subset of customers and returning to the depot. Time windows represent an interval of time for each customer within which the visit must be made. A solution is a set of tours for a subset of vehicles such that all customers are served only once and time window and capacity constraints are respected. The objective is to minimize the distance traveled, and sometimes additionally to reduce the number of vehicles used. The problem is NP-hard (Garey and Johnson, 1979).

An $n \times m$ job shop scheduling problem consists of n jobs and m resources. Each job is a set of k completely ordered activities, where each activity has a duration and a resource on which it must be executed. The total ordering of activities in a job defines a set of precedence constraints. This implies that no activity can begin execution until the activity that immediately precedes it in the complete ordering has finished execution. Each of the k activities in a single job requires exclusive use of one of the m resources. No activities that require the same resource can overlap in their execution and once an activity is started, it must be executed for its entire duration (i.e. no preemption is allowed). The job shop scheduling decision problem is to decide if all activities can be scheduled, given for each job a release date of 0 and a due date of the desired makespan D, while respecting the resource and precedence constraints. The job shop scheduling decision problem is NP-complete (Garey and Johnson, 1979).

While not part of the basic JSP definition, transition times and resource alternatives have been studied in the scheduling literature (Focacci, Laborie, and Nuijten, 2000). In the case of a transition time, there is a temporal constraint specifying a minimum time that must pass between each pair of activities executed on the same resource. When there are alternative resources in the formulation, each activity has a set of resources on which it can be executed. This extension of the JSP is known as the flexible job shop problem.

3. Mutual reformulations

Polynomial transformations of problems are part of the foundation of the theory of NP-completeness. These transformations are used to show that a problem of interest is inherently difficult by demonstrating that an instance of a known difficult problem can be converted into an instance of the problem of interest in a number of steps polynomial in the size of instance (Garey and Johnson, 1979). Here we are performing similar steps but from a somewhat practical perspective, to show that it is possible to tackle a problem in one class with the solution technology that is used for problems in a different class (Beck, Prosser, and Selensky, 2002).

Transforming vehicle routing problems into scheduling problems. Each vehicle is represented as a resource, and each customer visit as an activity. The distance between a pair of visits corresponds to a transition time between the respective activities. Each activity can be performed on any resource, and is constrained to start execution within the time window defined in the original VRP. For each resource R there are two special activities $Start_R$ and End_R as shown in Fig. 1. Activities $Start_R$ and End_R must be performed first and last, respectively, on resource R. The transition time between $Start_R$ and any other activity A_i corresponds to the distance between the depot and the ith visit. Similarly, the transition time between End_R and A_i corresponds to the distance between the depot and the ith visit. The processing time of $Start_R$ and End_R is zero. We associate a consumable secondary resource with every (primary) resource to model the capacity of vehicles. Consequently, a sequence of activities on a resource corresponds to a vehicle's tour in the VRP. In the resultant scheduling problem each job consists of only one activity, each activity can be performed on any resource, and there are transition times between each pair of activities. The problem is then to minimize the sum of transition times on all machines.

Transforming scheduling problems into vehicle routing problems. We have for each resource, a vehicle, and for each activity, a customer visit. The visits have the same duration as the corresponding activities. Each visit can be made only by the vehicles corresponding to the set of resources for the activity (more than one resource in the case of flexible JSPs, and only one in the case of pure JSPs). Any ordering between activities in a job results in precedence constraints

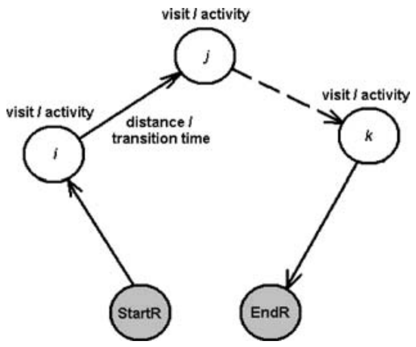


Fig. 1 Mutual VRP/JSP Transformations: Visits/activities performed by a vehicle/machine

between visits. Transition times between activities correspond to travel distances between visits. The deadline D imposes time windows on visits. Assuming we have m resources, and therefore m vehicles, we have $2m$ dummy visits corresponding to the departing and returning visits to the depot. A vehicle's tour corresponds to a schedule on a resource. For the $n \times m$ JSP we have a VRP with $m(n+2)$ visits and m vehicles. Since there are no transition times in the JSP, there are no travel distances between visits. There are precedence constraints between the visits reflecting the sequence of activities in the corresponding job. The decision problem is then to find an ordering of visits on vehicles that respects the precedence constraints and time windows.

4. Problem characteristics

We propose that the performance differences between routing and scheduling techniques are due to the characteristics of the routing and scheduling problems. In this section, we identify five such characteristics. In our study, we then generate sets of VRPs and JSPs by systematically varying the characteristics.

Alternative resources. Perhaps the most obvious difference between standard VRP and scheduling problems is the number of alternative resources that may be used for each operation. In vehicle routing, there are typically many vehicles that can be used to perform a visit. For example, the standard Solomon benchmarks (Solomon, 1987) have 25 identical vehicles. In contrast, scheduling problems tend to have very few alternative resources and, as mentioned in Section 2, the classical JSP has no alternatives. For example, Focacci, Laborie, and Nuijten (2000) experiment on problems with up to three alternatives while Davenport and Beck (1999) use problem instances with at most eight alternatives. We expect few resource alternatives to favor scheduling techniques while many should favor the VRP techniques.

Precedence relations among activities. In classical VRPs, each visit is independent, i.e. there are no constraints requiring a visit to have some temporal relation with other visits. In contrast, scheduling problems typically have long chains or directed acyclic graphs of temporal relationships among activities. As a consequence, temporal reasoning has been widely explored in the literature (Cesta, Oddi, and Smith, 2000) and is an important component of many scheduling algorithms (Laborie, 2003). We therefore predict that the performance of scheduling technology should improve as more precedence constraints are added to problems.

The ratio of operation duration to transition time. In classical VRPs, a visit has no duration but transition times can be very long. In contrast, in the JSP, the transition time between operations is zero but the operation duration can be very long. These two problem models are extremes. In both the literature and in the real world there are VRPs with visit durations and scheduling problems with transition times and costs. However, even then the ratio of operation duration to transition time tends to be substantially different. In VRPs the ratio is very small while in scheduling it is often very large. This difference is reflected in solution techniques. Many VRP techniques (e.g., the savings heuristic (Clarke and Wright, 1964)) look exclusively at transition times when searching, while many of the efficient constraint propagation (Nuijten, 1994; Le Pape, 1994) and heuristic search techniques (Beck and Fox, 2000; Smith and Cheng, 1993) in scheduling ignore the transition time. Overall, we expect that the smaller the ratio of operation duration to transition time, the better the VRP techniques should perform relative to the scheduling techniques.

Optimization criterion. The optimization criterion in the classical VRP is the minimization of the total distance traveled by all vehicles. Therefore, we expect the distance-related criterion to favor the vehicle routing technique. However, in some practical situations we may be more interested in completing visits as soon as possible, e.g. by the end of the working day. In terms of modeling, this is equivalent to minimizing the makespan: the time between the start of the first and the end of the last visit.

Contrary to vehicle routing, makespan is a common criterion in scheduling. Even though its relevance has been questioned in practical applications (Beck, Davenport, and Fox, 1997; Fox, 1983), makespan has received a great deal of attention in the literature. It is therefore reasonable to expect that most scheduling techniques are more amenable to minimizing makespan. However, there may be situations on the shop floor when we are interested in other criteria such as tardiness (Baptiste, Le Pape, and Nuijten, 2001), earliness and tardiness (Beck and Refalo, 2003), flow time or transition times (Focacci, Laborie, and Nuijten, 2000). To investigate how changing the optimization criterion influences the performance of the scheduling techniques we choose the sum of transition times as an alternative criterion in the experiments below.

Resource capacity. Resource capacity in a VRP is the maximum number of visits that a vehicle can perform in any solution. As the resource capacity is decreased, the minimum number of vehicles required in any solution increases. In a scheduling problem, resource capacity is defined as the number of activities that can simultaneously use a resource. The notion of capacity that we adopt here is that of VRP.

Reducing resource capacity in a VRP results in more tightly constrained visits since an increasing subset of visits cannot be performed by the same vehicle. We predict that such tighter side constraints will make it relatively more difficult for the VRP techniques to find good solutions as compared with the scheduling techniques which, through global constraint propagation, will exploit these constraints better during search. Reducing the VRP definition of resource capacity in a scheduling problem is meaningless for a classical JSP since all activities are assigned a resource. Instead of starting from a JSP, the base scheduling problem in this case is a flexible JSP where each activity can be executed on any resource. The reduction in resource capacity has a similar impact as in VRPs: an increasing subset of activities cannot be scheduled on the same resource. However, because we are dealing with flexible JSPs it is harder to make predictions as to the impact of changing resource capacity since the base problem is more complex than the classical JSP.

We believe that these characteristics are important in explaining the performance differences between VRP and scheduling techniques, because they are sufficient to transform one problem model to the other. Starting with a classical VRP, if we reduce the alternative resources to a singleton for each task, add chains of precedence constraints, reduce the transition time to zero while increasing the task duration, and change the optimization criterion to makespan minimization, we have a classical JSP problem.

All of the above characteristics are typical of real-world problems. For example, in VRPs the vehicle fleet is rarely homogeneous. It is typical that we have classes of vehicles that are specialized for certain tasks, such as tankers or refrigeration lorries, or drivers with specific skills. Precedence constraints are not uncommon in delivery or routing problems (Brind, Muller, and Prosser, 1995). Sometimes we are not concerned about reducing the amount of travel, but in getting all the visits done as quickly as possible.

The same holds true for scheduling problems. Changing machine configurations may be very expensive and so it might be financially justifiable to minimize the amount of time spent reconfiguring machines, thus increasing returns on investment. Therefore, the classical model that best corresponds to a real-world problem is unclear.

5. Experimental design

Our goal is to empirically determine the effect of the above five problem characteristics on the relative performance of scheduling techniques and routing techniques. All experiments are run on a 1 Ghz, Pentium III, with 520 Mb RAM running Windows NT.

5.1. Solution technology

In our experiments we use the ILOG optimization suite. The scheduling library (ILOG Scheduler 5.2) has global constraint propagation (Baptiste, Le Pape, and Nuijten, 2001; Laborie, 2001; Nuijten, 1994) within constructive tree search as its core technology, while the vehicle routing library (ILOG Dispatcher 3.2) focuses on constraint-based local search (DeBacker et al., 2000; Kilby, Prosser, and Shaw, 2000).

As part of the routing technique, we apply *first accept* neighborhood search enhanced by the guided local search meta-heuristic (Voudouris and Tsang, 1998) with a penalty factor of 0.4. To construct a neighborhood of a solution, we use the standard move operators (*TwoOpt, OrOpt, Relocate, Cross*, and *Exchange*). Search starts from the first solution found by the savings heuristic or, in some cases noted below, uses the first solution found by the scheduling technique.

The scheduling technique is constructive depth first search with standard slack-based heuristics (Smith and Cheng, 1993), global constraint propagation, and edge finding (Baptiste, Le Pape, and Nuijten, 2001).

Performance of the techniques is understood in terms of quality of obtained feasible solutions and computational time. Each technology is run on an instance for the same specified CPU time and the quality of the two solutions is compared.

5.2. Experiments

We perform two symmetric sets of experiments. In the first set, we take instances of the classical VRP and examine the effect of varying one problem characteristic at a time. We solve each problem twice: first we model the problem as a VRP and solve it using routing technology; second, we model the VRP as a scheduling problem, using the transformation presented in Section 3, and solve it with scheduling technology. We then evaluate the relative performance of the technologies. We generate additional problem sets by varying the parameters one after the other. Intuitively, as the parameter value deviates more from the VRP, the problems should become increasingly similar to classical JSP.

In a symmetric set of experiments, we take classical JSP instances and again vary one problem parameter at a time. We generate JSPs, transform them, and, as above, apply both scheduling and routing technologies to each instance to evaluate the relative performance of the techniques.

5.3. Problems and their parameters

Across all problems, the working day starts at midnight and lasts 24 hours. Time is measured in minutes, consequently the origin of our schedule is 0 and the horizon is 1440. Each operation duration is 20 minutes. Each problem set in our experiments has 100 instances.

Every VRP instance has 1 depot, 25 vehicles, 100 customers. Distance is measured in decameters using the Manhattan metric. The other parameters for generating VRPs are listed below.

Each JSP instances has 10 machines and 10 jobs, i.e. 10 totally ordered sets of operations. In every instance of the initial set, a single machine is specified for each operation. There are no transition times between operations. The other parameters for generating JSPs are displayed below.

The parameters common to both problem types are as follows:

1. the number of operations, n.

   – VRP: the number of customers. In all problems $n = 100$.
   – JSP: the number of activities. In all problems $n = 100$.

2. the number of resources, m.

   – VRP: the number of vehicles. In all problems $m = 25$.
   – JSP: the number of machines. In all problems $m = 10$.

3. normalized slack, σ.

$$\sigma = \frac{le_i - \tau_i - es_i}{le_i - es_i}, \tag{1}$$

   where τ_i is the duration of operation i, and le_i, and es_i are the latest end and earliest start of operation i. It follows from this definition that $\sigma \in [0, 1]$ and that the larger σ becomes, the wider the time window is for the same τ_i. Note that this slack measure is individual on the operations in an instance.

   – VRP: In all problems $\sigma = 0.9$.
   – JSP: In all problems $\sigma = 0.986$ (time windows are as long as the working day).

4. resource specialization, $0 \leq p \leq 1$. The parameter p represents the proportion of the pool of resources (vehicles or machines) that can perform each operation. For operation i, the number of possible resources m_i is calculated as:

$$m_i = \lceil p \times m \rceil \tag{2}$$

   Small values of p correspond to few resource alternatives and therefore correspond more to classical JSPs.

   – VRP: the default value is $p = 1.0$ (any vehicle can do any visit).
   – JSP: the default value is $p = 0.1$ (each operation has a single specified machine).

5. the (integer) number of precedence constraints, pc. This parameter specifies the number of pairs of visits whose sequence is constrained such that visit i must end before the start of visit j. Note that i and j are not necessarily on the same resource. Large values of pc correspond to many precedence constraints characteristic of classical JSPs.

   – VRP: the default value is $pc = 0$.

– JSP: the default value is $pc = \frac{uv(v-1)}{2}$, where u is the number of jobs and v is the number of operations per job.

6. ρ. The ratio of visit duration to travel time for the VRP, or the ratio of activity duration to transition time for the JSP. This parameter is used to set V, the speed of the vehicles or the speed of transitions on machines:

$$V = \frac{\rho \times \bar{d}}{\bar{\tau}} \tag{3}$$

where $\bar{\tau}$ is the average duration of visits (activities), and $\bar{d}$ is the average distance (transition time) between them. Large ρ values are typical of scheduling problems while small ρ values correspond to classical VRPs.

– VRP: the default value is $\rho = 1.0$.
– JSP: the default value is $\rho = \infty$ (transition time equals zero).

7. the optimization criterion, c.

– VRP: the default value is c = *minimize total travel time*.
– JSP: the default value is c = *minimize makespan*.

8. average resource capacity, R_c.

$$R_c = \frac{\sum_{i=1}^{m} r_{c_i}}{n}. \tag{4}$$

Here r_{c_i} is the capacity of vehicle (machine) i measured as the number of operations (customer visits or activities) that resource i can perform, m is the number of resources, n is the overall number of operations.

– VRP: the default value is $R_c = 25.0$ (r_{c_i} is set constant such that each one of $m = 25$ vehicles can perform $n = 100$ visits).
– JSP: the default value is $R_c = 1.0$ (r_{c_i} is set constant such that each one of $m = 10$ machines can perform $n = 10$ operations).

5.4. Problem generation

Instances of classical VRPs and JSPs are generated with the default parameters as specified above. The generated problem instances (http://www.dcs.gla.ac.uk/pras/resources/) are filtered by attempting to solve them. If no feasible solution could be found by the scheduling algorithm within five CPU minutes, the instance was not included in the set. This filtering is done on all problem sets, both the pure JSPs and VRPs and the sets with different values of the independent variables. A discussion of the implications of this filtering is provided in Section 5.7.

5.4.1. Generating VRPs

To create the initial set of classical VRP instances, the geographic coordinates of each visit are randomly drawn from all postal codes of locations in the city of Glasgow, within a five km radius of the city center. The data is drawn such that more than one visit could be generated using the same geographical location. The postal codes can be directly translated to coordinates with an accuracy of ten meters. Manhattan metric is assumed in all experiments.

We have also generated additional problem sets varying one parameter at a time while all the remaining parameters take their default values.

- p Resource specialization is varied such that $p \in \{0.04, 0.1, 0.23, 0.5, 1.0\}$. With $m = 25$ this corresponds to VRPs with 1, 3, 6, 13, and 25 possible vehicles per visit, respectively. For a given p, to get the corresponding number of vehicle alternatives, we randomly select vehicles for each visit.
- R_c Vehicle capacities take on the following values: $R_c \in \{1.25, 2.50, 6.25, 25.00\}$. This corresponds to resources capable of performing 5, 10, 25, and 100 customer visits.
- ρ The visit duration to travel time ratios are as follows: $\rho \in \{0.25, 0.5, 1.0, 2.0, 2.25, 3.0\}$. All visit durations were set at 20 minutes, consequently problems have a vehicle speed ranging from 12 km/h to 72 km/h.
- pc The different numbers of precedence constraints are 0, 50, 450, 1200. Each precedence constraint imposes an ordering for two given visits, without forcing these two visits to use the same vehicle. This corresponds to varying numbers and lengths of totally ordered precedence chains based on the number of jobs and operations in a job as explained above.
- c In the 100 classical VRPs we change the optimization criterion from *total travel time* to *makespan*.

5.4.2. Generating JSPs

Each classical JSP generated is a 10×10 instance. The default parameters of generated JSPs are given above.

We have also generated additional problem sets varying one parameter at a time. Unless otherwise stated, all the remaining parameters take their default values.

- p Resource specialization $p \in \{0.1, 0.25, 0.5, 1.0\}$. This corresponds to each operation having 1, 3, 5, and 10 alternative machines. Instance generation is analogous to VRPs.
- R_c Because varying resource capacity in classical JSPs does not have an influence on the quality of solutions, we experiment with flexible job shop instances. We generate additional problem sets such that in each 10×10 instance every activity has 10 alternative resources ($p = 1.0$). All the remaining parameters take their default values. Resource capacity varies such that $R_c \in \{1.3, 2.5, 6.3, 10.0\}$. This corresponds to each machine being capable of performing 13, 25, 63, and 100 activities.
- ρ The activity duration to transition times ratio is chosen from the following set: $\rho \in \{0.5, 1.0, 3.0, 10.0, \infty\}$. A ρ of ∞ corresponds to classical JSPs with infinitely fast transitions between operations on a machine.
- pc We vary the number of precedence constraints such that $pc \in \{0, 40, 126, 450\}$. With respect to pc in isolation, removing precedence constraints corresponds to making the instances similar to the classical VRPs.
- c In the 100 classical JSPs we change the optimization criterion from *minimize makespan* to *minimize total processing time*. Total processing time minimization can be thought of as minimizing the sum of makespans over all machines.

5.5. Evaluation criteria

Our primary evaluation criterion is λ, the ratio of the cost of the best solution found by the scheduling technique to that found by the VRP technique in a given CPU time limit.

Specifically,

$$\lambda = \frac{C_{sched}}{C_{rout}}, \tag{5}$$

where C_{sched} and C_{rout} are the cost of the best solutions found by the scheduling and routing techniques in the given amount of CPU time. Because we are minimizing, routing technology outperforms scheduling technology when $\lambda > 1$. When $\lambda < 1$, scheduling technology dominates.

5.6. Choosing the CPU time limit

To choose the CPU time bound for our empirical study we ran preliminary experiments. In each experiment we randomly took 20 instances out of 100 problem sets of VRPs with varying operation duration to travel time ratios ρ. Each problem was solved using different CPU time limits up to 50 minutes as displayed in Fig. 2. Each point in Fig. 2 is a mean λ over 20 test instances displayed against ρ and CPU time limit. We see that there is no significant change in λ after 10 minutes. An analysis of the results shows that neither solution technology makes significant improvements in the solution quality. As a result, the CPU time bound is chosen to be 10 minutes.

5.7. Discussion

Two points regarding the problem generation require further discussion. First, as briefly noted in Section 5.1, different starting solutions are used with the VRP technology. Specifically, when solving variations of routing problems, the VRP technology starts from a solution found by the savings heuristic. However, when solving variations of JSP problems, the VRP technology starts from the first solution found by the *scheduling* technology. This approach is followed because the routing technology was not able to find a first solution. The first solution construction heuristic is sensitive to nonclassical VRP characteristics such as resource allocation constraints and temporal relationships between activities (Beck, Prosser, and Selensky, 2003). Starting the VRP technology from a scheduling solution may result in an unfair advantage for the VRP technology. We believe

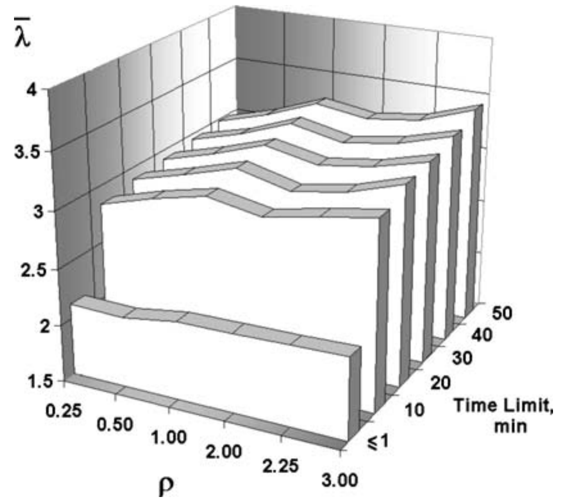


Fig. 2 Influence of CPU time on λ for different values of ρ. Each value of λ is an average of 20 classical VRPs with the other parameters having the default values

that any such advantage is negligible as, with no upper bound on the cost function, it is very easy for the scheduling technology to find a feasible solution. However, as a consequence the starting solutions tend to be of low quality: it is only when significant constraint propagation occurs via bounding of the optimization function that the scheduling technology is able to focus on high quality areas of the search space. Starting VRP solving using the scheduling solution may weaken conclusions that directly compare VRP to JSP technology. However, we believe that the low quality of the starting solutions used mitigates this weakness as a simple heuristic technique could be designed to find solutions of equal quality.

Secondly, the problem sets are preprocessed to remove instances for which a feasible solution could not be found by the scheduling technology within five CPU minutes. It is, therefore, relatively easy to find a *feasible* solution to our problem instances. The scheduling technology was used because preliminary experiments showed that it was more robust in finding feasible solutions. This is to be expected as the routing technology uses the one-pass savings heuristic to find a first solution while the scheduling technology conducts a backtracking search. There were a number of instances in our experiments where the scheduling algorithm was able to find a feasible solution while the routing technology was not. The inverse situation was never observed.

Our main objective in performing the problem filtering was to remove instances with no feasible solutions. Having such instances would skew the problem instances in favor of the scheduling algorithm since, as a constructive search technique, it is complete. For practical reasons, we adopted the five minute time limit. Unfortunately, such a design may benefit one solution technology over the other. For example, since the feasible solutions are easy to find, one might argue that the local search approach of the routing technology is favored as it is based on examining a neighborhood for locally improving moves. Conversely, because we are using the scheduling technology to filter the problem instances, the scheduling technology may be favored: it is possible, though we believe unlikely for the reasons given above, that the routing technology could have found a feasible solution to some of the instances that were discarded. Overall, because our empirical results can be reasonably understood by the specific impact of the manipulation of the independent variables on the solution technologies, we do not believe that the filtering has a significant impact on our conclusions. Nonetheless, problem filtering represents a weakness is our experimental design that should be avoided in future work.

6. Empirical results

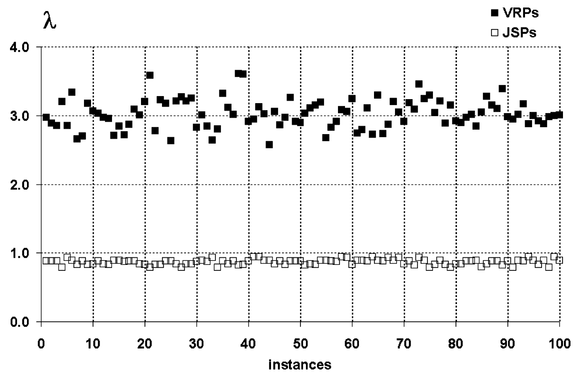
6.1. Base cases: Instances of classical VRPs and JSPs

In this section, we present the experimental results obtained by scheduling and routing technologies on two base problem sets, classical VRPs and classical JSPs.

In Fig. 3 each point represents one problem instance. For each of the VRP and JSP instances we ran the two techniques and computed the value of λ. For each JSP instance, search starts from common initial solutions found by the scheduling technique.

Clearly, the routing technology dominates for the VRP instances ($\lambda > 1$, with an average of 3.0) and the scheduling technology dominates for the JSPs ($\lambda < 1$, with an average of 0.9). The results on the routing problems also appear significantly more variable than those on the scheduling problems. We believe that this is a consequence of the fact that our measurement, λ, is the ratio of JSP algorithm performance to VRP algorithm performance. Differences in absolute performance for $\lambda < 1$ (i.e. the scheduling problems) will appear as much smaller differences in λ than when $\lambda > 1$ as for the routing problems.

Fig. 3 Base case: λ for 100 instances of classical JSPs and 100 instances of classical VRPs



We now study the influence of the following: variation in the number of alternative resources, resource capacity, the number of precedence constraints, the influence of operation duration versus transition times, and, finally, optimization criteria.

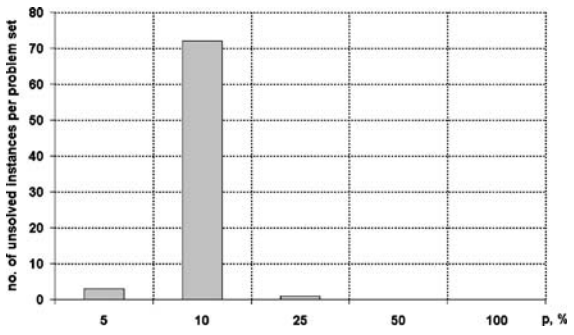
6.2. Alternative resources (p)

6.2.1. VRP instances

To investigate the influence of resource alternatives on the performance of the solving technologies, we ran a series of experiments using instances with varying p that were generated as explained in Section 5.4. The routing technology could not solve some of the instances with a small number of alternative resources. Figure 4 shows how many instances per problem set could not be solved by the routing technology. We observe an insolubility peak at $p = 0.1$. We conjecture that at $p = 0.1$ problems are critically constrained such that there are very few feasible vehicle assignments. Since the savings heuristic is a single pass heuristic, it does no search for such assignments and therefore fails when they are difficult to construct. With $p < 0.1$, no vehicle choices are necessary since only one vehicle can perform each visit. At $p > 0.1$, many vehicles are possible, increasing the number of feasible vehicle assignments.

In Fig. 5 we present λ with respect to p, expressed as a percentage, for first solutions (λ_{first}) and for best solutions (λ_{best}) found by both techniques. Both contours in Fig. 5 exclude any instance for which the routing technology could not construct a first solution. Search starts from independently found first solutions. We see that as the number of vehicles (machines) per visit (activity) increases, VRP technology improves relative to scheduling technology.

Fig. 4 The number of instances per problem set that could not be solved by the routing technique vs. the number of alternative resources



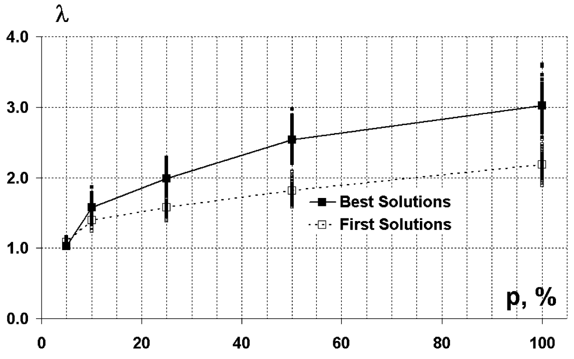


Fig. 5 VRPs: Effects of varying vehicle specialization

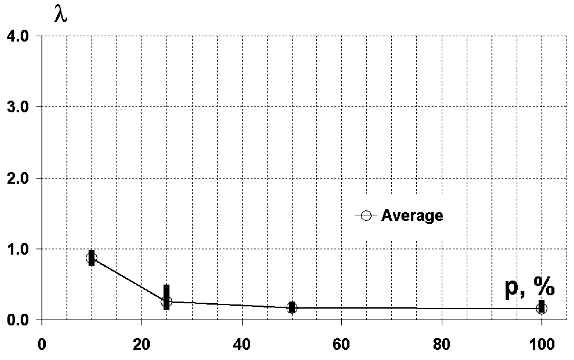


Fig. 6 JSPs: Effects of varying machine specialization

6.2.2. JSP instances

In Fig. 6 we show how λ depends on the number of alternative resources allowed for an operation. In the graph, search starts from common first scheduling solutions. We could expect that as p grows and, consequently, problems become more similar to classical VRPs, the routing technology should be favored. However, we see an initial drop in λ followed by a relatively flat curve indicating that the scheduling algorithm is outperforming the routing algorithm. The increase in alternative resources does not make the problems easier for the routing technology because the precedence constraints continue to make many neighbors infeasible. The additional alternative resources serve to significantly increase the size of the neighborhood that must be examined at each move.

6.3. Resource capacity (R_c)

6.3.1. VRP instances

Figure 7 shows that across the range of varied resource capacity the scheduling technique is dominated by the routing technique. In Fig. 7, search starts from independently found first solutions. Recall that varying resource capacity (slack) does not necessarily make a problem more or less similar to the classical job shop or VRP.

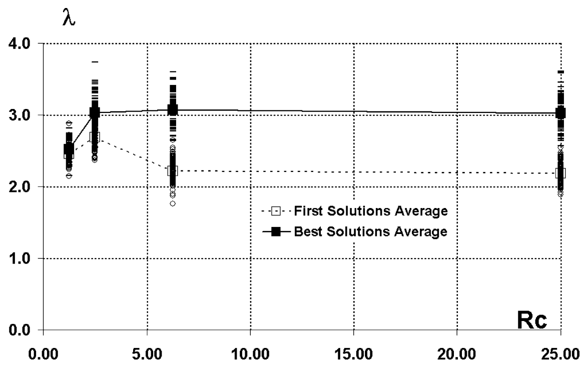


Fig. 7 VRPs: Varying vehicle capacity

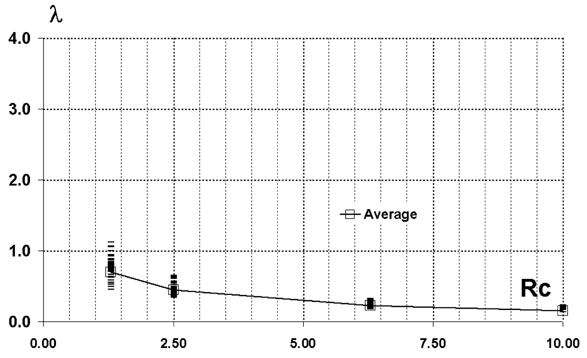


Fig. 8 JSPs: Varying machine capacity

6.3.2. JSP instances

Experimenting with varying resource capacity in classical JSPs is not meaningful because changing machine capacity does not have an influence on the quality of solutions. Consequently, we experiment with flexible job shop instances, i.e. instances where an activity can be performed by any machine.

Figure 8 shows that as we increase resource capacity the routing technology is dominated by the scheduling technology. Search starts here from common first scheduling solutions.

The picture is analogous to the results of the experiments with alternative resources (Section 6.2): the precedence constraints make finding feasible moves difficult for the routing technology and the alternative resources increase the size of the neighborhood without providing more feasible moves.

6.4. Precedence constraints (pc)

6.4.1. VRP instances

To study the influence of precedence constraints we vary pc. Experiments show that as we introduce precedence constraints the VRP technology again fails to produce initial solutions, but the scheduling technology appears to be robust. If both techniques start with a scheduling solution (Fig. 9), the VRP technology dominates the scheduling technology (when $pc = 0$, $\lambda \approx 1.2$) but this becomes marginally less dominant as we increase the number of precedence constraints (when $pc = 1200$, $\lambda \approx 1.1$).

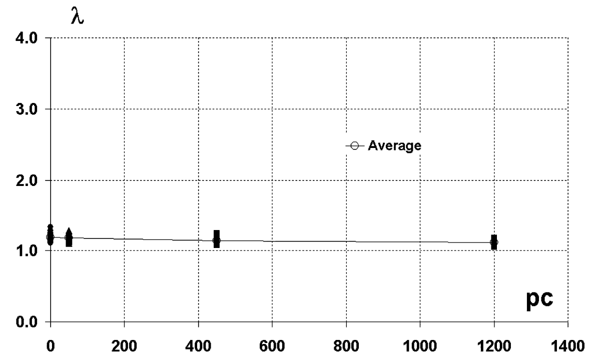


Fig. 9 VRPs: Influence of the number of precedence constraints

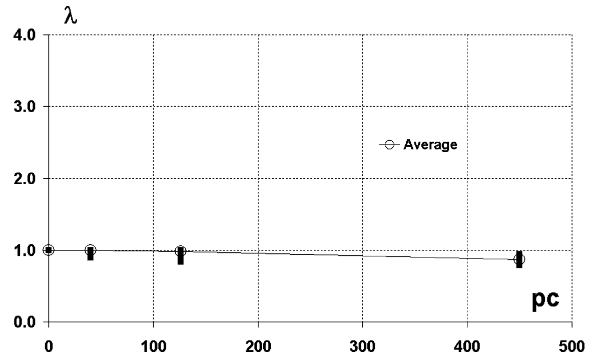


Fig. 10 JSPs: Influence of the number of precedence constraints

6.4.2. JSP instances

In Fig. 10 we display how relative solution quality λ is affected by the number of precedence constraints in JSPs. We see that as we remove precedence constraints and both technologies start from common initial scheduling solutions, they tend to perform equally ($\lambda \approx 1$). It is an expected result because if we remove *all* precedences from a JSP, the problem becomes tractable. Indeed, subject to no precedence constraints, no alternative resources and zero transition times, we can construct an unsorted list of n operations on each of the m resources in $O(nm)$ time. The length D of the longest list of operations on some resource r_i gives us the makespan. It is optimal because when we have zero transition times D equals the sum of durations of operations on r_i and these operations cannot be relocated to any other resource. Consequently, when $pc = 0$, the first solution found by the scheduling technique is an optimal solution and finding those solutions does not require any search.

6.5. Operation duration to transition time ratio (ρ)

6.5.1. VRP instances

In Beck, Prosser, and Selensky (2002) it was argued that higher ρ's will tend to favor scheduling techniques because large values of ρ correspond to small transition times compared to operation durations, typical of scheduling problems. Our results suggest, however, that in isolation this parameter does not bring about such a performance difference and the routing technology continues

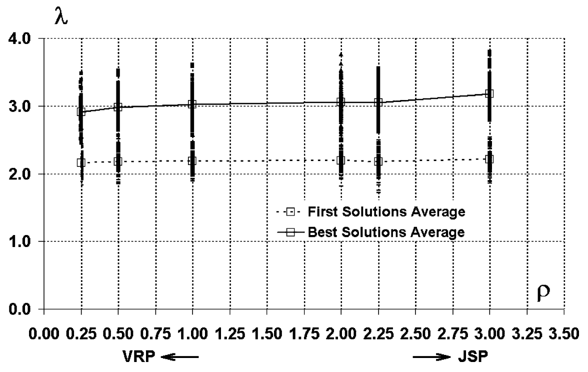


Fig. 11 VRPs: Effects of varying the ratio of operation duration to transition time

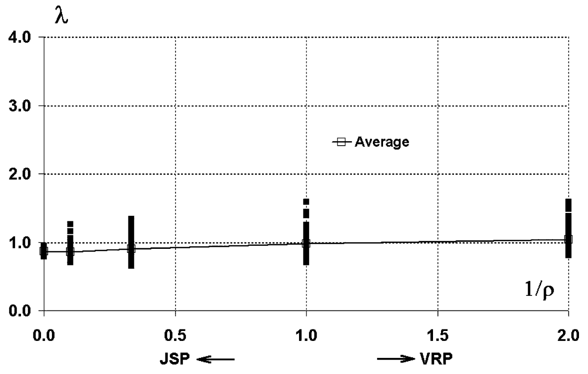


Fig. 12 JSPs: Effects of varying the ratio of operation duration to transition time

to outperform the scheduling technique as can be seen in Fig. 11. Search starts here from independently found initial solutions.

6.5.2. JSP instances

In Fig. 12 we observe that increasing ρ (i.e. decreasing $1/\rho$ in the picture) for the JSPs improves the relative performance of the routing technique (when $\rho = \infty$ and $1/\rho = 0$, $\lambda \approx 0.87$ whereas when $\rho = 0.5$, $\lambda \approx 1.04$). In Fig. 12, search starts from common initial scheduling solutions.

6.6. Optimization criterion (c)

6.6.1. VRP instances

In Fig. 13 we present a scatter plot, on the left λ_{best} and λ_{first} using the minimization of travel, and on the right λ_{best} and λ_{first} where the criterion is minimizing makespan. We see a large difference in the behavior of the solving technologies. When minimizing travel (scatter on the left) the routing technology is consistently 2 to 3 times better than the scheduling technology. When we switch to minimizing makespan (scatter on the right) the scheduling and routing technologies perform equally (i.e. λ_{best} and λ_{first} are around 1). These results are particularly dramatic given that all the other parameters are that of a classical VRP: the only difference is the optimization criterion.

We attribute the observed performance difference when we change optimization criterion to a difference in the suitability of the underlying techniques. Minimizing the makespan results

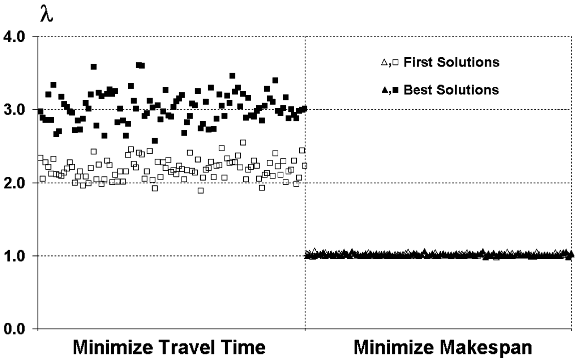


Fig. 13 VRPs: Swapping optimization criterion for classical VRPs

in significantly more constraint propagation than the sum of travel times. When minimizing a maximum function (such as makespan), the upper bound can be directly propagated on the completion time of each operation/visit, and the domains of possible start times can be effectively tightened. However, when a sum function is being optimized, propagation is weaker and, in some circumstances, may not result in any reduction in the possible start times (Baptiste, Le Pape, and Nuijten, 2001). The scheduling technology is built to exploit the powerful constraint propagation that is possible using such criterion as makespan, whereas the routing technology is not.

6.6.2. *JSP instances*

We now experiment with two different optimization criteria for classical JSPs, makespan and total processing time. In Fig. 14 we have a scatter plot with the minimization of the makespan on the left and the minimization of total processing time on the right. The difference in mean λ between these two sets of data is statistically significant with $p \leq 0.05$. When minimizing makespan (scatter on the left) the routing technology is consistently dominated by the scheduling technology. When we switch to minimizing total time (scatter on the right) the routing technology starts to compete with the scheduling technology. The observed change in solution quality in favor of the routing technique occurs in the area of zero transition cost (i.e. where arcs in the routing graph have a zero cost). Because the cost function computation in the routing technique is entirely arc-based, we expect this change to be even greater for JSPs with nonzero transition costs.

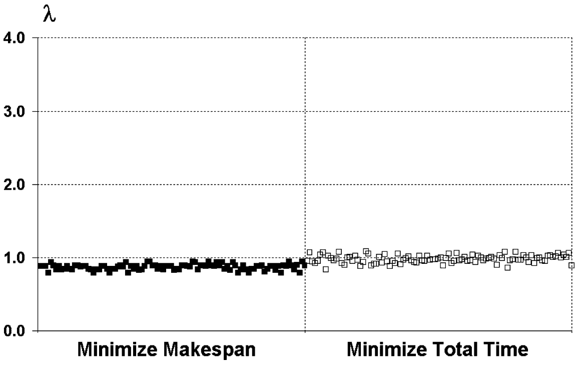


Fig. 14 JSPs: Swapping optimization criterion for classical JSPs

6.7. Summary of results

We have varied the number of alternative resources, the number of precedence constraints, the operation duration to transition time ratio, and the optimization criterion. Each parameter was varied in isolation, in two symmetric sets of experiments. In the first series of experiments, we started from classical VRPs and transformed them systematically so that they became increasingly similar to classical JSPs. In the second series, we started from classical JSPs and moved toward VRPs. We have also varied the resource capacity. However, with this parameter to have a meaningful comparison of technique performance, we used classical VRPs in the first set of experiments and flexible job shop in the second.

Figure 15 pictorially summarizes our study. The domains of classical VRP and classical JSP are shown as circles. Problem transformations are depicted by means of arrows labeled by the corresponding problem parameters. We also show the response to each parameter change in the relative quality λ of the solving techniques.

The most prominent effects on the performance of the solution techniques were brought about by varying

– *The number of alternative resources p*. The removal of alternative vehicles in the VRPs favors the scheduling technology but the addition of alternative machines to the JSPs proves even less amenable to the routing technique than classical JSPs.
– *Resource capacity R_c*. The effects of varying the resource capacity are analogous to the effects of changing the number of alternative resources. Decreasing capacity in VRPs favors the scheduling technology although it remains dominated by the routing technique. In contrast, increasing resource capacity in flexible JSPs leads to even more pronounced dominance of the scheduling technique.
– *The number of temporal constraints between activities pc*. An increase in *pc* in VRPs favors the scheduling technique, and, symmetrically, a decrease in *pc* in the JSP improves the performance of the routing technique.

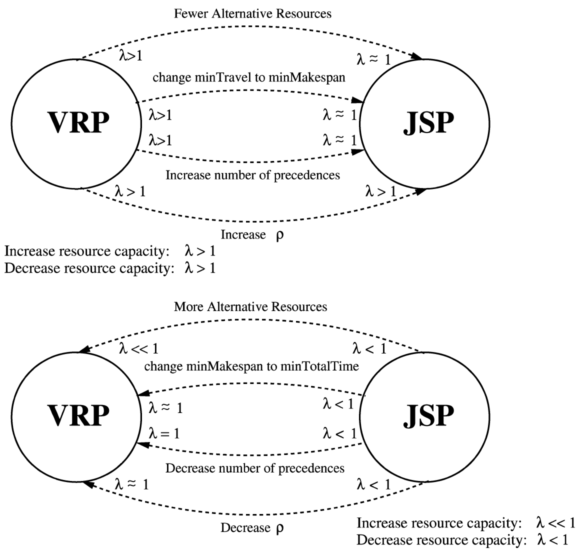

Fewer Alternative Resources
λ>1
λ ≈ 1
change minTravel to minMakespan
VRP
λ>1
λ ≈ 1
λ>1
λ ≈ 1
JSP
Increase number of precedences
λ > 1
λ > 1
Increase ρ
Increase resource capacity: λ > 1
Decrease resource capacity: λ > 1
More Alternative Resources
λ << 1
λ < 1
change minMakespan to minTotalTime
VRP
λ ≈ 1
λ < 1
λ = 1
λ < 1
JSP
Decrease number of precedences
λ ≈ 1
λ < 1
Decrease ρ
Increase resource capacity: λ << 1
Decrease resource capacity: λ < 1

Fig. 15 Influence of various problem characteristics on the performance of VRP and JSP technologies. Arrows represent reformulation direction, arrow labels show the varied parameter

– *Optimization criterion c*. Changing c from total travel minimization to makespan minimization in VRPs notably favors the scheduling technology. Symmetrically, changing c from makespan minimization to total processing time minimization in JSPs favors the routing technique.

7. Perspectives and contributions

In many ways, the work described in this paper is nontraditional: it was not our goal to demonstrate a new optimization algorithm, to solve a new problem, or to perform detailed empirical analysis of the low level components of existing algorithms. Rather, motivated by the fact that real-world problems are seldom a perfect match for a classical, mathematical optimization problems studied in the literature, we performed a systematic empirical study to analyze the robustness of two optimization technologies to problem complications common in real-world problems. The contributions of this work are

– We have identified the problem characteristics that represent important differences in the definition of job shop scheduling and vehicle routing problems. By manipulation of these characteristics, researchers can generate hybrid problem instances that may be more representative of real-world problems than the classical problem definitions.
– We have empirically analyzed the effect of changing each of the problem characteristics.
– The identification of the impact of each problem characteristic serves as a foundation for practitioners who have a real problem to solve. For high performance, customization of commercial software is necessary. However, depending on the important characteristics of the real problem, the practitioner can use our analysis as a basis to decide which technology should be the starting point for the customization effort.
– The identification of the impact of each problem characteristic also serves as a starting point for researchers interested in making constraint technology easier to use. If we are to achieve a sufficient level of performance with "out of the box" optimization technology without detailed customization by the user, we need to address the variations in problem characteristics that have the most impact on current technology.

7.1. Generality

There are a number of issues with regard to the generality of our results and the extent to which our findings can be directly used to answer questions beyond those that we asked above. In this section, we address these issues.

7.1.1. Using commercial software

Given that we have used ILOG Scheduler and ILOG Dispatcher without significant customization, will the results observed in our empirical studies be the same with different scheduling and VRP technology or even with the next version of the commercial software?

Both software products are implementations of technology that has been published in the academic literature (Clarke and Wright, 1964; Smith and Cheng, 1993; Le Pape, 1994; Nuijten, 1994; Baptiste and Le Pape, 1996; Prosser, Kilby, and Shaw, 1997; Voudouris and Tsang, 1998; Focacci, Laborie, and Nuijten, 2000; DeBacker et al., 2000; Beck and Perron, 2000; Kilby, Prosser, and Shaw, 2000; Baptiste, Le Pape, and Nuijten, 2001; Laborie, 2003). While a practitioner can use them as a blackbox, they are not blackboxes to the research community. Therefore, the question of whether a study using independent implementations of these techniques will produce consistent results is the same question that accompanies any empirical study of algorithm performance. The use of commercial implementations of algorithms that have been *published in the literature* does not reduce the generality or potential reproducibility of our results. In fact,using commercial software, with its high level of software engineering and optimization, rather than "research code" makes it less likely that a reimplementation using clever data structures would produce substantially different results or that the results presented here are due to a programing error. Furthermore, the usefulness of our results for the practitioner is increased as the software packages are commercially available.

There is an another sense in which our experiments are generalizable in a way that some standard studies in the optimization literature are not. It is not uncommon in studies that compare competing approaches to a problem, that the winning technology has a strong correlation with the background of the authors. This is because developing techniques for solving optimization problems is a challenging and creative endeavor and existing expertise in the technology allows the researcher to concentrate on core problem solving. The outcome may, therefore, be due more to the time, effort, and quality of the researcher rather than inherent characteristics in the optimization technique. By using existing technology, our experimental design controls for our levels of expertise with the different technologies. Had we proceeded differently and modified the underlying technology to deal with the specific changes in problem characteristics, not only would our study be significantly less useful to practitioners but it would also have been more a test of our creativity and expertise than of the robustness of the technology.

7.1.2. Generalizing to other algorithms

Our experiments used only two algorithms, one scheduling algorithm and one routing algorithm. Of obvious interest is the question of whether our findings would generalize to other scheduling and routing algorithms. If we have identified fundamental problem characteristics that are important in solving scheduling and routing problems, then we would expect to see similar results with other algorithms. However, this work, as a "case study" of two specific algorithms does not provide any basis on which to make any such claims. Further study using different algorithms with the set of parameters used here is needed to assess the generality of our results in this direction.

7.1.3. Changing more than one parameter at a time

We did not conduct a fully crossed experiment but rather only assessed the impact of changing one parameter at a time. Real problems are unlikely to be so well-behaved, so it is also of interest to understand algorithm performance when multiple parameters have changed from their default values. Our experimental design provides some evidence as we varied parameters starting from both routing and scheduling problems. Therefore, for example, we have evaluated the effect of changing the optimization criterion in the presence and absence of precedence constraints among the activities/visits in those problem instances created by modifying classical JSP and VRP problems, respectively. While we have undertaken no direct experiments, we can speculate based on the results that we have observed. The magnitude of the performance changes with the addition of precedence constraints and with the change in the optimization criterion suggesting that these changes would dominate "opposite" changes in the number of alternative resources and resource capacity. As indicated by the JSP results with modifying the optimization criterion (Section 6.6), the presence of precedence constraints appears to slightly dominate the optimization criterion parameter.

7.1.4. Solving larger problems

By the standards of academic problems, the problems instances used in our study are small (i.e. standard constraint programing algorithms can generally solve 10×10 JSPs to optimality in a few seconds). A final question of generality of this work, then, is the application of our findings to larger problems. We would expect that algorithm performance on larger problems would reflect a combination of the results observed here and the scaling behavior of the underlying algorithms. As it is generally believed that local search algorithms scale better than constructive algorithms, one could argue that for larger problems, the routing algorithm would start to dominate as the scaling effects became more important than the effects of changes in the problem parameters. However, the impact of precedence constraints on the routing algorithm will not be reduced with larger problems and therefore we expect that the scheduling technology would still dominate such problems. More generally, constraint programing techniques are most successful in highly constrained problems. As some of the parameter manipulations can generate highly constrained problems, we do not believe that the effects of the scaling of problem size are easily predicted. Further empirical work is necessary to elucidate this impact.

8. Conclusions

We have considered the vehicle routing and job shop scheduling problems and carried out an empirical study to develop an understanding of how their characteristics contribute to the performance of standard solution technologies. We have identified five such parameters: vehicle specialization, complex temporal relationships between operations, the optimization criterion, operation duration to travel time ratio, and resource capacity. In the experiments, we started from classical VRPs and, separately, from classical JSPs and varied these parameters, each in isolation, to assess their influence on the performance of the solution techniques.

In our study, we used routing and scheduling technology found in commercially available software. We believe that this is a crucial feature of our study. The current research literature boasts a number of powerful approaches for dealing with combinatorial optimization problems (e.g., constraint programing, mixed integer programing, SAT solvers, etc.). These techniques have been developed for classical models. For problems that do not fall directly under a classical model, however, these solution techniques require significant skill and experience to apply to their full potential. In this study, we were not interested in the algorithmic customizations we could make to scheduling or routing technology to allow it to tackle more realistic problems more efficiently. Rather, we were interested in changing the problem characteristics of classical models and studying the impact on existing solution techniques. We hope that our findings might then be of real value to practitioners who use commercial software to address real-world problems.

Understanding how changes in classical models effect solution techniques is critical in extending the scope of optimization technology to more realistic problems. Furthermore, one of our long term goals is to decrease the expertise required to customize algorithms for real-world problems. We plan to investigate the automation of problem analysis to be able to suggest appropriate solution technology. The identification of the relationship between problem characteristics and search performance is an important step in this direction.

Acknowledgements The work in this paper was conducted while the first author was employed by ILOG, S.A. and, subsequently, the Cork Constraint Computation Centre, University College Cork and while the third author was employed by the Department of Computing Science, University of Glasgow and, subsequently, the Cork Constraint Computation Centre, University College Cork.

References

Baptiste, Ph. and C. Le Pape, “Edge finding constraint propagation algorithms for disjunctive and cumulative scheduling,” in *Proceedings of the Fifteenth Workshop of the U.K. Planning Special Interest Group*, 1996.

Baptiste, Ph., C. Le Pape, and W. Nuijten, *Constraint-based scheduling: Applying constraint programming to scheduling problems*, Kluwer Academic Publishers, 2001.

Beck, J. C. and M. S. Fox, “Dynamic problem structure analysis as a basis for constraint-directed scheduling heuristics,” *Artificial Intelligence*, **117**(1), 31–81 (2000).

Beck, J. C. and P. Refalo, “A hybrid approach to scheduling with earliness and tardiness costs,” *Annals of Operations Research*, **118**, 49–71 (2003).

Beck, J. C., A. J. Davenport, and M. S. Fox, “Five pitfalls of empirical scheduling research,” in *3rd Int. Conference on Principles and Practice of Constraint Programming (CP’97)*, 1997.

Beck, J. C. and L. Perron, “Discrepancy-bounded depth first search,” in *Proceedings of the Second International Workshop on Integration of AI and OR Technologies for Combinatorial Optimization Problems (CPAIOR’00)*, 2000.

Beck, J. C., 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)*, 2002, pp. 282–289.

Beck, J. C., 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.

Brind, C., C. Muller, and P. Prosser, “Stochastic techniques for resource management,” *BT Technology Journal*, **13**, 55–65 (1995).

Cesta, A., A. Oddi, and S. F. Smith, “A constraint-based method for project scheduling with time windows,” *Journal of Heuristics*, **8**(1), 109–136 (2000).

Clarke, G. and J. W. Wright, “Scheduling of vehicles from a central depot to a number of delivery points,” *Operations Research*, **12**(4), 568–581 (1964).

Davenport, A. J. and J. C. Beck, “An investigation into two approaches for constraint directed resource allocation and scheduling,” In *INFORMS*, 1999.

DeBacker, B., V. Furnon, P. Shaw, P. Kilby, and P. Prosser, “Solving vehicle routing problems using constraint programming and metaheuritics,” *Journal of Heuristics*, **6**, 5001–5523 (2000).

Focacci, F., P. Laborie, and W. Nuijten, “Solving scheduling problems with setup times and alternative resources,” in *Proceedings of the 5th International Conference on Artificial Intelligence Planning and Scheduling*, 2000.

Fox, M. S., *Constraint-Directed Search: A Case Study of Job Shop Scheduling*. PhD thesis, Carnegy Mellon University, Intelligent Systems Laboratory. The Robotics Institute, Pittsburgh, PA, 1983.

Garey, M. R. and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York, 1979.

Kilby, P., P. Prosser, and P. Shaw, “A comparison of traditional and constraint-based heuristics methods for vehicle routing problems with side constraints,” *Constraints*, **5**(4), 389–414 (2000).

Laborie, P., “Algorithms for propagating resource constraints in AI planning and scheduling: Existing approaches and new results,” in *Proceedings of the 6th European Conference on Planning (ECP01)*, 2001.

Laborie, P., “Algorithms for propagating resource constraints in AI planning and scheduling: Existing approaches and new results,” *Artificial Intelligence*, **143**, 151–188 (2003).

Le Pape, C., “Implementation of resource constraints in ILOG SCHEDULE: A library for the development of constraint-based scheduling systems,” *Intelligent Systems Engineering*, **3**(2), 55–66 (1994).

Nuijten, W. P. M., “Time and resource constrained scheduling: a constraint satisfaction approach,” PhD thesis, Department of Mathematics and Computing Science, Eindhoven University of Technology, 1994.

Prosser, P., P. Kilby, and P. Shaw, “Guided local search for the vehicle routing problem,” in *Proceedings of the 2nd International Conference on Metaheuristics*, 1997.

Puget, J. F., “Constraint programming next challenge: Simplicity of use,” in M. Wallace, (ed.), *Proceedings of the Tenth International Conference on Principles and Practice of Constraint Programming (CP’04)*, 2004 pp. 5–8.

Smith, S. and C. Cheng, “Slack based heuristics for constraint satisfaction scheduling,” in *Proceedings of the Eleventh National Conference on Artificial Intelligence (AAAI–93)*, 1993 pp. 139–144.

Solomon, M., “Algorithms for the vehicle routing and scheduling problem with time window constraints,” *Operations Research*, **35**, 254–365 (1987).

Voudouris, C., and E. P. K. Tsang, “Guided local search,” *European Journal of Operational Research*, **113**(2), 80–110 (1998).

Walsh, T., “Reformulating propositional satisfiability as constraint satisfaction,” in *Symposium on Abstraction, Reformulation and Approximation (SARA)*, 2000.

PRAS vehicle routing and shop scheduling problem instances. http://www.dcs.gla.ac.uk/pras/resources/.