An Empirical Study of Mutual Routing-Scheduling Reformulation*

J. Christopher Beck[1], Patrick Prosser[2], and Evgeny Selensky[2]

[1] Cork Constraint Computation Center, University College Cork, Ireland.
c.beck@4c.ucc.ie
[2] Department of Computing Science, University of Glasgow, Scotland.
pat/evgeny@dcs.gla.ac.uk

Abstract. Despite a number of similarities, vehicle routing problems and scheduling problems are typically solved with different techniques. In this paper, we undertake a systematic study of problem characteristics that differ between vehicle routing and scheduling problems in order to identify those that are important for the performance of typical vehicle routing and scheduling techniques. In particular, we find that the addition of temporal constraints among visits or the addition of tight vehicle specialization constraints significantly improves the performance of scheduling techniques relative to vehicle routing techniques.

1 Introduction

It is a long standing belief in AI that finding the right problem representation is a key component of solving a problem [11, 12]. The real world dictates the necessity of making compromises in formulating a problem. There will be aspects of the real problem that do not fit easily into a "pure" version of a known combinatorial optimization problem. The decision must then be made to ignore the problem components that cannot be modeled in a pure way or to add "impure" or customized model components. The drawback of the former approach is that it is not clear that solutions to the problem model are solutions to the real problem. The latter approach also presents challenges, due to the fact that we do not yet understand combinatorial problem solving well enough to be able to predict the effect of adding a customized model component. It can be that what looks like a minor addition to the model significantly changes the performance of standard problem solving techniques.

This paper directly addresses the relationship between problem characteristics and search performance. In previous work [19], it was shown that scheduling technology performs poorly on reformulated vehicle routing problems while vehicle routing technology performs poorly on reformulated scheduling problems. Here, we investigate the reasons behind this results. We perform an in-depth study by systematically transforming pure vehicle routing problems (VRP) such

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

that they become more like scheduling problems and, symmetrically, by transforming pure scheduling problems so that they look more like VRPs. We then solve these problems with routing and with scheduling technologies, to discover what problem characteristics influence the problem solving techniques.

This paper is organized as follows. We start by defining vehicle routing problems and scheduling problems, and show how we can reformulate VRPs as scheduling problems, and vice versa. We then identify five characteristics that we believe differentiate routing problems from scheduling problems. We describe the design of our experiments, to investigate the effect of these problem characteristics on the relative performance of routing and scheduling technology. The results of these experiments are reported and followed by a conclusion.

2 Problem Definitions: VRP & JSP

In the capacitated vehicle routing problems with time windows, m identical vehicles initially located at a depot are to deliver discrete quantities of goods to n customers. Each customer has a demand for goods and each vehicle has a capacity. A vehicle can make only one tour starting at the depot, visiting a subset of customers and returning to the depot. Time windows define an interval 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 distance traveled, and sometimes additionally to reduce the number of vehicles used. The problem is NP-hard [15].

An $n \times m$ job shop scheduling problem (JSP) consists of n jobs and m resources. Each job is a set of m completely ordered activities, where each activity has a duration for which it must execute and a resource which it must execute on. The total ordering defines a set of precedence constraints, meaning that no activity can begin execution until the activity that immediately precedes it in the complete ordering has finished execution. Each of the m activities in a single job requires exclusive use of one of the m resources defined in the problem. 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 [15].

While not part of the basic JSP definition, transition times and resource alternatives have been reported in the scheduling literature [13]. In the case of a transition time, there is a temporal constraint specifying a minimum time that must expire between pairs of activities executed on the same resource. When alternative resources are represented, each activity has a set of resources that it can be performed on (and this extension of the JSP is known as the flexible job shop problem).

3 Mutual Reformulations

In [5] two reformulations are presented: a reformulation of VRP as a flexible shop problem with setups, and symmetrically a reformulation of JSP as a a VRP.

We reformulate the VRP into a scheduling problem as follows. 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 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. Each activity has a demand for a secondary resource, corresponding to a visit's demand within a vehicle. For each resource R there are two special activities $Start_R$ and End_R. Activities $Start_R$ and End_R must be performed on resource R. $Start_R$ must be the first activity performed on R and End_R the last. The transition time between $Start_R$ and any other activity A_i corresponds to the distance between the depot and the i^{th} visit. Similarly the transition time between End_R and A_i corresponds to the distance between the depot and the i^{th} 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 and maybe also to minimize the number of resources used.

We reformulate a scheduling problem into a VRP as follows. We have for each resource a vehicle, and for each activity a customer visit. The visits have a duration the same as that of the corresponding activities. Each visit can be made only by the vehicles corresponding to the set of resources for the activity. Any ordering between activities in a job results in precedence constraints 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. Each visit can be performed only by one vehicle. Since there are no transition times in the JSP, there are no travel distances between visits, but visits have durations corresponding to those of the activities. There are precedence constraints between those visits corresponding to activities in a 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

The performance difference between VRP and scheduling techniques must be due to the characteristics on which the problems differ. In this section, we identify five main characteristics that we believe are sufficient to explain the performance differences between the VRP and scheduling techniques.

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 [21] have twenty five identical vehicles. In contrast, scheduling problems tend to have very few alternative resources and, as noted, in the pure JSP there are no resource alternatives. This has been reflected in the scheduling literature. For example, Focacci et al. [13] experiment on problems with up to three alternatives while Davenport & Beck [9] use problem instances with at most eight alternatives. We expect few resource alternatives to favour scheduling techniques while many should improve the performance of VRP techniques.

Temporal Constraints In pure 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 complex directed acyclic graphs of temporal relationships among activities. As a consequence temporal reasoning has been widely explored in the literature [7] and is an extremely important component of many scheduling algorithms [16]. We therefore predict that scheduling technology should improve on problems where there are complex temporal constraints.

The Ratio of Operation Duration to Transition Time In pure VRPs a visit has no duration. In contrast, in pure JSP the transition time between operations is zero. The modeling abstraction for scheduling is obviously opposite to the VRP: the transition time between activities on the same resource is so small that it can be ignored. These two problem models are extremes. In both the literature and in the real world there are VRPs with visit duration and scheduling problems with transition times and costs. However, even then the ratio of operation duration to transition time tends to be 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 [8]) look exclusively at transition times when searching, while many of the strong constraint propagation [18, 17] and heuristic search techniques [20, 2] in scheduling ignore transition time. Overall, we can 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 In a VRP the standard optimization criterion is to minimize the total distance traveled by each vehicle. In scheduling, a common criterion is the minimization of makespan: the time between the start of the first operation and the end of the last operation. While many criteria have been studied (e.g., tardiness [1], earliness and tardiness [3], flow time, transition times [13]) makespan has received a great deal of attention in the literature even though its relevance has been questioned in practical applications [4], [14]. We can expect the performance of the scheduling techniques should be favoured when using the minimization of makespan as the criterion.

Temporal Slack Although it seems difficult to make specific predictions as to how temporal slack, i.e., the difference between the time window for an operation and the operation's duration, can affect the performance of the search techniques in both routing and scheduling, we think it worthwhile to experiment with it as well. Slack can be important while solving both routing and scheduling problems, e.g., there are state-of-the-art global constraint propagation algorithms and efficient search heuristics based on temporal slack [20].

These characteristics may be sufficient to explain the performance differences between VRP and scheduling techniques, because they are sufficient to transform one problem model to the other. E.g., starting with a pure VRP, if we reduce the alternative resources to a singleton for each task, add chains of temporal constraints, reduce the transition time to zero while increasing task duration, and change the optimization criterion to makespan minimization, we have a pure JSP problem.

5 Experimental Design

Our goal is to empirically determine which of the above five problem characteristics actually have an effect on the relative performance of scheduling techniques compared to VRP techniques.

We perform two symmetric sets of experiments. In the first set of experiments, we take pure VRP instances and examine the effect of varying one problem characteristic at a time. We generate instances of VRPs. We then 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 (as described in sect. 2) and solve it with scheduling technology. We then measure the relative performance of both technologies. For each problem characteristic, therefore, we begin with a set (or ensemble) of pure VRP instances. After that we generate additional problem sets by varying the parameter in question. Intuitively, as the parameter value deviates more from the VRP, the ensemble should look increasingly like scheduling problems.

In a symmetric set of experiments, we take pure JSP instances and again vary one problem parameter at a time. We generate pure and transformed JSPs and, as for the VRPs above, apply both scheduling and routing technologies to each instance and measure their relative performance.

In both series of experiments, the size of an ensemble of instances is 100.

5.1 Problems and Their Parameters

VRP instances To create the initial set of pure VRP instances, the geographic coordinates of each visit are randomly drawn with replacement from a data file containing all postal codes of locations in the city of Glasgow, within a 5 km radius of the city centre. These postal codes can be directly translated to coordinates with an accuracy of ten meters. Because we are using city coordinates, all distances are Manhattan distance. The other parameters for generating VRPs are listed below.

JSP instances Each of the 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 on a given machine. The other parameters for generating JSPs are displayed below.

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

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

p resource specialization. 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:

$$m_i = \lceil p.m \rceil \tag{1}$$

Small values of p correspond to few resource alternatives and therefore correspond more to scheduling problems.
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).

pc The (integer) number of precedence constraints. 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 scheduling problems.
VRP: the default value is $pc = 0$.
JSP: the default value is $pc = 450$ (10 jobs with 10 operations each).

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:

$$V = \frac{\rho.\overline{d}}{\overline{\tau}} \tag{2}$$

where $\overline{\tau}$ is the average duration of visits/activities, and $\overline{d}$ is the average distance/transition between visits/activities. Large ρ values are typical of scheduling problems while small ρ values correspond to VRPs.
VRP: the default value is $\rho = 1.0$.
JSP: the default value is $\rho = \infty$ (transition time equals zero).

σ Normalized slack.

$$\sigma = \frac{le_i - \tau_i - es_i}{le_i - es_i}, \tag{3}$$

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. Operation durations are measured in minutes. The working day is 24 hours.
VRP: the default value is $\rho = 0.9$.
JSP: the default value is $\rho = 0.986$ (time windows are as long as the working day).

c The optimization criterion:
VRP: the default value is $c = minimize\ total\ travel\ time$.
JSP: the default value is $c = minimize\ makespan$.

Pure VRP and JSP problems are generated with the default parameters, specified above. Problems are tested for solubility, and are rejected if not found to be soluble within 5 minutes of CPU time.

5.2 Solution Technology

In our experiments we use the ILOG optimization suite. The scheduling library (ILOG Scheduler 5.2) has global constraint propagation [1, 16, 18] within constructive tree search as its core technology, while the vehicle routing library (ILOG Dispatcher 3.2) focuses on local search [10].

As part of the routing technique, we apply first accept neighbourhood search enhanced by the guided local search (GLS) metaheuristic [22] with a penalty factor of 0.4. To construct a neighbourhood 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 uses the first solution found by the scheduling technique. The scheduling technique is constructive depth first search with standard slack based heuristics, global constraint propagation, and edge finding [1].

5.3 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:

$$\lambda = \frac{C_{sched}}{C_{rout}}, \tag{4}$$

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 is outperforming scheduling technology when $\lambda > 1$.

When $\lambda < 1$, scheduling technology dominates. And, finally, when $\lambda = 1$ both technologies perform equally.

Apart from λ, we also check if technologies are capable of finding solutions within a specified time limit. The CPU time bound is chosen to be 10 minutes because our preliminary tests showed that there was no significant improvement in solutions for both solving technologies after 10 minutes.

6 The Empirical Results

6.1 Base cases: pure VRPs and JSPs

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

In Fig. 1 we display two scatters of λ, one for the base VRP instances and the other for the base JSP instances. Clearly, the routing technology dominates for the pure VRP instances ($\lambda > 1$) and the scheduling technology dominates for the pure JSPs ($\lambda < 1$). In the experiment with pure JSPs the routing technology wasn't able to come up with a first solution independently because the first solution construction heuristics are sensitive to impurities in the VRPs such as resource allocation constraints and temporal relationships between activities [6]. Therefore in this and all subsequent experiments with JSPs both technologies started search from common first scheduling solutions.

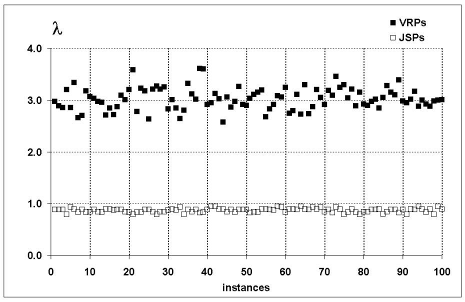

Fig. 1. Base case: λ's for 100 pure JSPs and 100 pure VRPs. The default parameters for the VRPs are $p = 1.0$, $\rho = 1.0$, $pc = 0$, $\sigma = 0.9$, both technologies start search independently. The default parameters for the JSPs are $p = 0.1$, $\rho = \infty$, $pc = 450$, $\sigma = 0.986$; search starts from common initial solutions found by the scheduling technique.

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

6.2 Alternative Resources (p)

VRP instances To investigate the influence of resource alternatives we create a series of problem sets by varying the vehicle specialization p, such that $p \in \{0.05, 0.1, 0.25, 0.5, 1.0\}$. 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 numbers of vehicle alternatives, we take a visit and randomly select without replacement so many vehicles of the fleet. All other parameters take their default values (i.e. $pc = 0$, $\rho = 1.0$, $\sigma = 0.9$, and c minimizing total travel time).

In Fig. 2 we present λ with respect to p, expressed as a percentage. Two contours are presented, one for λ for first solution found (λ_{first}) by both techniques, and λ for the best solution found (λ_{best}) by both techniques. We see that as specialization of the fleet increases (i.e. p decreasing) the performance of the routing technology degrades, and ultimately when we have one vehicle per visit the technologies perform approximately equally. Both contours exclude any instance that could not be solved by the routing technology in 10 minutes.

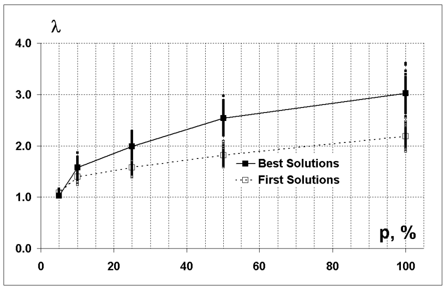

Fig. 2. VRPs: What happens when we specialize vehicles? p varies, $pc = 0$, $\rho = 1.0$, $\sigma = 0.9$, $c = min\ travel\ time$. Search starts from independently found first solutions.

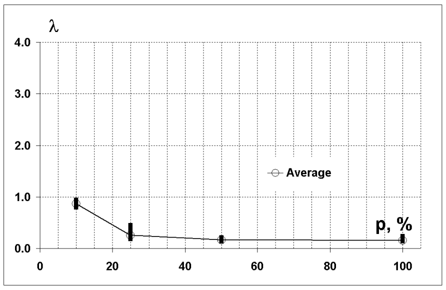

Fig. 3. JSPs: What happens when we vary machine specialization? p varies, $pc = 450$, $\rho = \infty$, $\sigma = 0.986$, $c = min\ makespan$. Search starts from common first scheduling solutions.

JSP instances We generate sets of JSPs with a varying percentage of resources allowed for each operation such that $p \in \{0.1, 0.25, 0.5, 1.0\}$. This corresponds to each operation having 1, 3, 5 and 10 alternative resources. All other parameters take their default values ($pc = 450$, $\rho = \infty$, $\sigma = 0.986$, $c =$ minimizing the makespan).

In Fig. 3 we show how λ depends on the number of alternative resources allowed for an operation. We could expect that as p grows and, consequently, problems become more VRP-like, the routing technology should be favoured. However, we see the opposite effect. The JSP turns out to be not so amenable to the routing technology because, on the one hand, there are many temporal constraints leading to the rejection of many moves and, on the other, a solution neighbourhood for the problem instances of this size is very large.

6.3 Precedence Constraints (pc)

VRP instances To study the influence of precedence constraints we vary pc, such that $pc \in \{0, 50, 450, 1200\}$. 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 such that 5 chains each involving 5 visits are equivalent to a pc of 50, 10 chains with 10 visits each to a pc of 450, 4 chains with 25 visits each to a pc of 1200.

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

JSP instances To experiment with varying numbers of precedence constraints, apart from the pure 10×10 JSPs with a pc of 450, we generate additional problem sets such that $pc \in \{0, 40, 126, 450\}$. With respect to the number pc of precedences between operations in isolation, removing precedence constraints correspond to making the instances more like the VRP.

Experiments show 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. Consequently, when $pc = 0$, the first solution found by the scheduling technique is an optimal solution for any instance.

6.4 Operation Duration vs Transition Time (ρ)

VRP instances VRP instances were generated with values of ρ (visit duration to transition times) 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 12km/h to 72km/h.

In [5] it was argued that higher ρ's will tend to favour scheduling techniques because large values of ρ correspond to small transition times compared to operation durations, typical of scheduling problems. Our results demonstrate, however, that in isolation this parameter does not bring about such a performance difference and routing technology continues to outperform the scheduling technique (when $\rho = 0.25$, $\lambda \approx 2.9$ and, when $\rho = 3.0$, $\lambda \approx 3.2$). This is not what we expected.

JSP instances We now generate JSP instances with 10 machines, 10 jobs and with varying values of ρ (activity duration to transition times) as follows: $\rho \in \{0.5, 1.0, 3.0, 10.0, \infty\}$. All operation durations were set at 20 minutes. A ρ of ∞ corresponds to pure JSPs with infinitely fast transitions between operations on a machine.

We observe that decreasing ρ for the JSPs marginally improves the relative performance of the routing technique (when $\rho = \infty$, $\lambda \approx 0.87$ and, when $\rho = 0.5$, $\lambda \approx 1.04$).

6.5 Optimization Criterion (c)

VRP instances We now solve pure VRPs using two different optimization criteria: minimize total travel (i.e. from a scheduling perspective, minimize the sum of setups), and minimize makespan (complete all the visits as early as possible). In Fig. 4 we have 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 behaviour 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 standard VRP: the only difference is the optimization criterion.

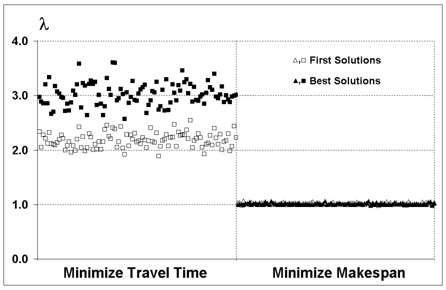

Fig. 4. VRPs: What happens when we swap optimization for pure VRPs? On the left, *minimize travel time*, and on the right *minimize makespan*. Search starts from independently found first solutions.

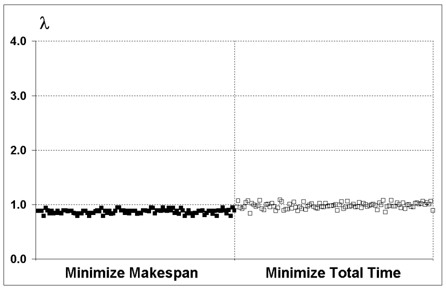

Fig. 5. JSPs: What happens when we swap optimization for pure JSPs? On the left, *minimize makespan*, and on the right *minimize total time*. Search starts from common first scheduling solutions.

Why is there such a difference when we change optimization criterion? One answer might be the underlying techniques used in the scheduling technology. 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 (e.g., sum of transition times) is being optimized, the respective constraint has to be considered at each stage of search simultaneously with resource constraints [1] thus making it harder to achieve large domain reductions. The scheduling technology we used is built to exploit the powerful constraint propagation that is possible using such criterion as makespan, whereas the routing technology is not. We might therefore expect when we have a VRP, and the goal is to complete all visits as soon as possible, that we might be better solving it as a scheduling problem.

JSP instances We now experiment with two different optimization criteria for pure JSPs, makespan and total processing time. Total processing time minimization can be thought of as minimizing the sum of makespans over all machines.

In Fig. 5 we have a scatter plot with the minimization of the makespan on the left and the minimization of total processing time on the right. When minimizing makespan (scatter on the left) the routing technology is consistently dominated by the scheduling technology (the mean λ is 0.868). When we switch to minimizing total time (scatter on the right) the routing technology starts to compete with the scheduling technology (the mean λ is 0.977).

These results again demonstrate the importance of the optimization criterion for the performance of the routing and scheduling techniques. Note that even though the values of λ have not changed significantly for the JSPs, the change in favour of the routing technique does occur in the area of zero transition cost (i.e., where arcs in the routing graph have a zero cost)! Because the cost function calculation in the routing technique is entirely arc-based, we expect this change to be even greater for mixed JSPs with non-zero transition costs.

6.6 Temporal slack (σ)

VRP instances To experiment with slack, we generate VRPs with normalized slack $\sigma \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$. All other parameters take default values. As σ increases, time windows get larger with respect to visit durations.

Experiments demonstrate that, as slack increases, routing technology improves relative to scheduling technology (from a λ of approximately 2.7 when $\sigma = 0.5$ to a λ of 3.0 when $\sigma = 0.9$). Conversely, when slack decreases it appears that scheduling technology is improving, yet continues to be dominated by the routing technology. We might conclude from this that as slack decreases, and problems become more temporally constrained, we might incorporate scheduling technology into the routing technology.

JSP instances We generate additional problem sets such that $\sigma \in \{0.65, 0.75, 0.85, 0.95, 0.986\}$. All other parameters take default values.

It turns out that tightening slack for JSPs leads to a λ of 1.0 (when $\sigma = 0.986$, $\lambda \approx 0.87$ and, when $\sigma = 0.65$, $\lambda \approx 1.0$). This is because the first solution found by the scheduling technique tends to be optimal as we are making the

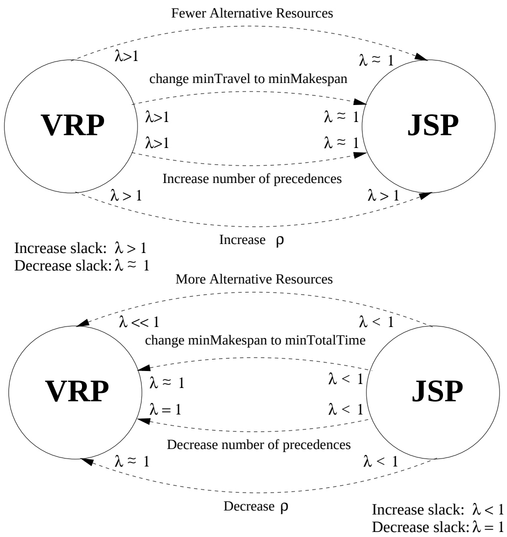

Fig. 6. Influence of various problem characteristics on the performance of VRP and JSP technologies. Arrows represent reformulation direction, arrow labels show the varied parameter. $\lambda > 1$ means routing technology outperforms scheduling technology, when $\lambda < 1$ scheduling technology is a winner, $\lambda \approx 1$ shows that both technologies perform equally.

problems more temporally constrained. Consequently, tightening slack in JSPs makes both techniques perform equally when search starts from common initial scheduling solutions. Conversely, increasing σ leads to the relative improvement of scheduling technology against routing technology.

6.7 Summary of Results

We have varied five different parameters, each in isolation, in two symmetric sets of experiments. In the first series of experiments, we started from pure VRPs and transformed them systematically so that they became more like JSPs. In the second series, we started from pure JSPs and moved towards VRPs.

We can transform a JSP into a VRP or vise versa by simultaneously varying the number of alternative resources, the number of temporal constraints between operations, the optimization criterion and operation duration vs. transition time at the same time. Varying slack does not contribute to this problem transformation.

Fig. 6 pictorially summarizes our study. The pure VRP and pure JSP domains 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 solving techniques were brought about by varying:

the number of alternative resources p The removal of alternative vehicles in the VRPs favours the scheduling technology but the addition of alternative machines to the JSPs proves even less amenable to the routing technique than pure JSPs;

the number of temporal constraints between activities pc An increase in pc in VRPs favours the scheduling technique, and, symmetrically, a decrease in pc in the JSP improves the performance of the routing technique;

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

7 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 solving technologies. We have identified five such parameters: vehicle specialization, complex temporal relationships between operations, the optimization criterion, operation duration to travel time ratio, and temporal slack.

In the experiments, we started from pure VRPs and, separately, from pure JSPs and varied those parameters, each in isolation, to assess their influence on the performance of the solving techniques. In our study we used existing VRP and scheduling technology found in commercially available tools. This may appear naive. However, we believe that this is a critical feature of our study. The current research literature boasts a number of powerful approaches for solving combinatorial optimization problems (e.g., constraint programming, mixed integer programming, SAT solvers, etc.). These approaches, however, require significant skill and experience to apply to their full potential. In this study, therefore, we were not interested in the algorithmic customizations we could make to scheduling or VRP technology to allow it to solve "impure" problems better. Rather we were interested in the problem characteristics and their impact on existing solution techniques. One of our long term goals is to automate the process 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.

References

1. Ph. Baptiste, C. Le Pape, and W. Nuijten. *Constraint-Based Scheduling: Applying Constraint Programming to Scheduling Problems.* Kluwer Academic Publishers, 2001.
2. J. C. Beck and M. S. Fox. Dynamic problem structure analysis as a basis for constraint-directed scheduling heuristics. *Artificial Intelligence*, 117(1):31–81, 2000.

3. J. C. Beck and P. Refalo. A hybrid approach to scheduling with earliness and tardiness costs. *Annals of Operations Research*, 2002. in press.
4. J.C. Beck, 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.
5. J.C. Beck, P. Prosser, and E. Selensky. On the reformulation of vehicle routing problems and scheduling problems. In *LNAI 2371, Proceedings of the Symposium on Abstraction, Reformulation and Approximation (SARA 2002)*, pages 282–289, 2002.
6. J.C. Beck, P. Prosser, and E. Selensky. Vehicle routing and job shop scheduling: What's the difference? In *Proceedings of the 13th International Conference on Artificial Intelligence Planning and Scheduling*, 2003.
7. A. Cesta, A. Oddi, and S.F. Smith. A constraint-based method for project scheduling with time windows. *Journal of Heuristics*, 8(1):109–136, 2000.
8. G. Clarke and J.W. Wright. Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. *Operations Research*, 46:93–100, 1964.
9. A.J. Davenport and J.C. Beck. An investigation into two approaches for constraint directed resource allocation and scheduling. In *INFORMS*, 1999.
10. B. DeBacker, V. Furnon, P. Shaw, P. Kilby, and P. Prosser. Solving vehicle routing problems using constraint programming and metaheuritics. *Journal of Heuristics*, 6:5001–523, 2000.
11. E. Fink. Systematic approach to the design of representation-changing algorithms. In *Symposium on Abstraction, Reformulation and Approximation (SARA)*, 1995.
12. E. Fink. How to solve it automatically: Selection among problem-solving methods. In *4th Int. Conference on Artificial Intelligence Planning Systems*, 1998.
13. F. Focacci, 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.
14. M.S. Fox. *Constraint-Directed Search: A Case Study of Job Shop Scheduling*. PhD thesis, Carnegy Mellon University, Intelligent Systems Laboratory, The Robotics Institute, Pittsburgh, PA, 1983.
15. M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W.H. Freeman and Company, New York, 1979.
16. P. Laborie. 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.
17. C. Le Pape. 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.
18. W. P. M. Nuijten. *Time and resource constrained scheduling: a constraint satisfaction approach.* PhD thesis, Department of Mathematics and Computing Science, Eindhoven University of Technology, 1994.
19. E. Selensky. On mutual reformulation of shop scheduling and vehicle routing. In *Proceedings of the 20th UK PLANSIG*, 2001.
20. S. Smith and C. Cheng. Slack based heuristics for constraint satisfaction scheduling. In *Proceedings of the Eleventh National Conference on Artificial Intelligence (AAAI-93)*, pages 139–144, 1993.
21. M. Solomon. Algorithms for the Vehicle Routing and Scheduling Problem with Time Window Constraints. *Operations Research*, 35:254–365, 1987.
22. C. Voudouris and E.P.K. Tsang. Guided Local Search. *European Journal of Operational Research*, 113(2):80–110, 1998.