Automatic Customization Framework for Efficient Vehicle Routing System Deployment

Jussi Rasku, Tuukka Puranen, Antoine Kalmbach, and Tommi Kärkkäinen

Abstract Vehicle routing systems provide several advantages over manual transportation planning and they are attracting growing attention. However, deployment of these systems can be prohibitively costly, especially for small and medium-sized enterprises: the customization, integration, and migration is laborious and requires operations research expetise. We propose an automated configuration workflow for vehicle routing system and data flow customization, which provides the necessary basis for more experimental work on the subject. Our preliminary results with learning and adaptive algorithms support the assumption of applicability of the proposed configuration framework. The strategies presented here equip implementers with the methods needed, and give an outline for automating the deployment of these systems. This also opens up new directions for research in vehicle routing systems, data exchange, model inference, automatic algorithm configuration, algorithm selection, software customization, and domain-specific languages.

1 Introduction

Globalization, increased goods consumption, and economic changes pose challenges on transportation logistics. With increasing scale, tightened competition, and environmental concerns, dispatchers stretch their planning capabilities to the limit. Handling all the factors may even be impossible for the human planner [21], which has spawned an interest in commercial automated route planning systems. Combined with the rapid development of IT, this has created a transportation planning tools industry serving operators working with the increasingly complex logistics systems [14].

Jussi Rasku, e-mail: jussi.rasku@jyu.fi · Tommi Kärkkäinen, e-mail: tommi.karkkainen@jyu.fi
Department of Mathematical Information Technology, University of Jyväskylä, FI-40014, Finland

Tuukka Puranen e-mail: tuukka.puranen@nfleet.fi · Antoine Kalmbach
NFleet Oy, Pajatie 8 F, Jyväskylä, FI-40630, Finland

The advantages of these systems are well known: savings up to 30% in operational costs, reduced planning time, and minimization of human error [42]. Drexl [9] also note the macroeconomic benefits such as improved traffic flow and lowered emissions. If applied at a large scale, deployment of Vehicle Routing Systems (VRSs) can lead to significant economic and environmental benefits.

A VRS is described in Drexl [9] as follows: it is an operational planning software that can read in data with vehicles, drivers, depots, customers, and their respective requests connected to geographical locations. The data defines the specific problem scenario. A map view is often used for visual data verification. A VRS then allows manual, interactive or fully automated (optimization-based) construction of routes. The algorithms, that can build a routing plan conforming to the operational rules such as work time regulations, are the key feature of the system. Finally, the system interacts with an existing resource management system, or allows saving and printing the plans for operational use.

The operation environment for a VRS is complex and dynamic [42, 5, 31] and poses hard to match requirements for commercial software. In a survey from an industry viewpoint, Hoff et al [14] raised a concern that while academic Vehicle Routing Problem (VRP) research has provided efficient algorithms for these problems, they typically use idealized models which omit important facets such as driver breaks, work time regulations, turning restrictions, variation of service times, and congestion. According to Partyka and Hall [32] the providers are having difficulties in providing holistic solutions due to this complexity.

In addition to shortcomings of the idealized models, different logistic operators have differing requirements [5]. As it is not commercially viable to build a unique VRS for each of them [42] the product is made customizable. Here, a VRS designed for easy deployment needs to capture the features of the common VRP variants. Additionally, solving the problems effectively calls for robustness, adaptivity, and reactivity [42].

According to Partyka and Hall [32] routing installations require heavy customization which is mainly done manually. A survey of the Dutch VRS market by Kleijn [21] agrees: most of the software was at least partially tailor-made. The issue has been identified also in academic research. Puranen [36] proposes the use of Software Product Lines (SPL) as a mass-customization strategy. It is a promising approach, as these techniques exploit commonalities in a system to effectively manage variation. Applying SPL in other application domains report order of magnitude reductions in time-to-market, engineering overhead, error rates, and cost [24]. Preliminary experiments in [36] suggest that these benefits are achievable also with VRSs. The extended rationale for the work, as presented in [31], is that the underutilization of route optimization is not due to the shortcomings of models and algorithms, but due to problems in deployment.

The challenges in implementation and deployment call for an approach that could forward the adoption of optimization. In this paper, we propose such an approach as a set of actions and techniques to automate the flow of data through a VRS. Acting upon presented ideas allows utilization of the recent advances in Software Engineering, Machine Learning, and Operations Research.

In related works Desrochers et al [8] describe a VRS that could be used by a consultant with a basic understanding of mathematical programming. Similarly, Maturana et al [30] describe a decision support system generator that substantially lowers the cost of developing such systems, although in their solution the model and data structures has to be still defined manually. Also, Hoff et al [14] envision a tool exhibiting some of the properties presented here. Despite these ideas, a planning and decision support system that allows as extensive automation as ours has not been previously described. No customization framework for the needed automation methods has existed, and their interaction within VRSs has not been previously explored. In this paper, we address this by providing an automated configuration workflow for VRSs and a review on the automation methods for different phases of the process. The customization framework should be of interest not only to operations researchers, but also to providers of VRSs.

For an overview, Section 2 provides a review of the trends in vehicle routing research. In Section 3 we recognize the opportunities for automation in customizing from data flow perspective. Section 4 we present our proposition for solving some pressing problems in VRS deployment and Section 5 reviews our preliminary experimental results. In Section 6 we conclude our study.

2 Trends in Vehicle Routing Problem Research

VRP has been under intensive research ever since was first introduced by Dantzig and Ramser [7]. VRP concerns the task of finding optimal *routes* for a *fleet* of *vehicles* leaving typically from a *depot* to serve a specified number of *requests*. Objectives can be anything from minimizing the number of vehicles or total travel distance to complex multiobjective business goals. Over 50 years of academic interest has experienced many shifts of research focus. The trends in VRP research as recognized, e.g., by Puranen [36] are illustrated in Figure 1.

models:	idealized	→	rich	→	unified	→	inferred
methods:	simple	→	refined	→	adaptive	→	learning

Fig. 1: Trends in vehicle routing research. Adapted from [36]

The early models were **idealized**, partly due to the limitations of computational hardware and solution methods of the time. Since the early days, the trend has been towards more complex and more realistic **"rich"** problems [42, 14]. Rich models extend the classical formulation with complex decisions and objectives as well as can introduce many operational constraints. In addition, a number of new aspects have been proposed; for example Hoff et al [14] call for more explicit handling of stochasticity and risk in the models, and stress the need for research on real time and dynamic routing. For reviews on rich VRP research in the context of commercial so-

lutions, we refer to Hasle and Kloster [13], Drexl [9], and Bräysy and Hasle [5]. Recently, there has been efforts to develop **unified** modeling approaches with generic and flexible modeling structures that can capture the aspects of different VRP variants [40, 13, 18, 48, 37]. Vidal et al [45] make a synthesis on previous research and propose a naming scheme for these variants. Unified modeling frameworks often provide a Domain Specific Language (DSL) for describing the problems. One of the contributions of this paper concerns the rightmost transition in modeling: we argue that the advances in unified modeling enable model **inference** where *composite* optimization model can be automatically or semi-automatically formed by inferring the composition of the model from the problem instance data.

The solution methodologies have followed a similar trend. The first methods relied on **simple** heuristics or on mathematical programming as in the original paper by Dantzig and Ramser [7]. The growing problem size and model complexity led to interest in more **refined** and sophisticated methods. However, due to scale of the real-life problems, exact solution methods cannot always be used. Thus, a number of heuristics and metaheuristics have been proposed. For surveys in solving VRPs, see, e.g., Toth and Vigo [44], Cordeau et al [6], and Laporte [26]. Recently, there has been interest in **adaptive** and self-adjusting methods where algorithms observe the optimization progress and react accordingly. This trend was recognized, for example, in the survey by Vidal et al [46]. A newer trend is the application of **learning** hyperheuristic systems, which involve using data-driven techniques that enable and disable different algorithms depending on the observed search space. This involves identifying situations similar to those found in the history data or knowledge model. For a survey on using hyperheuristics in combinatorial optimization we refer to Kotthoff [23].

The disadvantage of unified and "rich" models and refined versatile solving methods is that they may make the deployment more complicated [31]. Also, note that in most of the case studies in the aforementioned surveys, the derivation of the model, selection of the algorithms, and fine tuning of the methods is done manually by the researchers based on their expertise. Unfortunately, this does not scale in a commercial setting and poses a barrier for the deployment of VRSs.

3 Data Flow in a Vehicle Routing System

In this section, we outline the *data flow* through a VRS, or more specifically, how the problem instance is passed from system module to another. The flow of information is one of the main aspects affecting the deployment, integration, and utilization of the system. Describing the modular structure of a typical VRS in detail is omitted, and the reader is referred to Drexl [9], Puranen [36], and Bräysy and Hasle [5].

The data flow can be divided into phases as illustrated in Figure 2. First, the data is read from a data storage, such as files or relational database, and then constructed into a domain model (1). Domain model offers primitives for concepts such as truck, driver, and request. The domain model is then translated into optimization

model (2). This involves describing the decision variables, the objective function, and the necessary constraints. Note that a DSL or similar technique has to be used to capture the aspects of the specific routing problem. Result of this transformation is a mathematical optimization model that can be then completed with the problem instance specific variable values. The modeled problem can now be fed to the routing algorithms residing in the solver module (3). Effectiveness of the algorithms depends heavily on the algorithm parameters [15]. Thus, when adapting a VRS care is needed to derive a set of suitable parameter values (4). After the optimization (5) the results can be transformed back to the primitives of the domain model (6) which in turn are translated into an actionable plan (7).

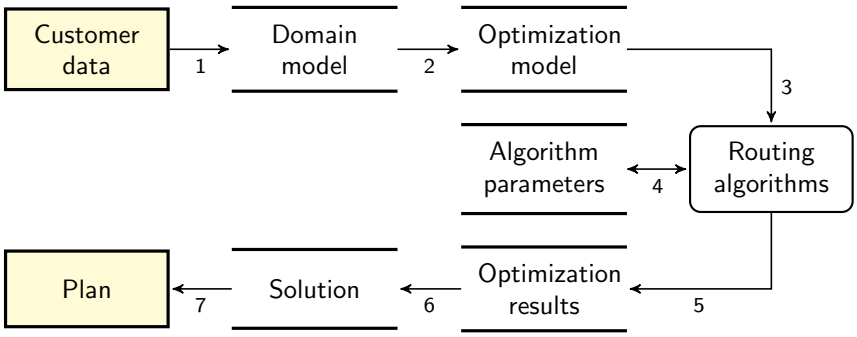


Fig. 2: Data flow of a problem instance trough a VRS.

Provided that the VRS is generic enough to model a wide set of different and "rich" VRP variants, and that it includes a set of modern metaheuristics and local search based routing algorithms, the biggest effort in adopting a VRS for a new customer is to make sure that the data is read and processed correctly [31]. VRS providers have several options to manage the data flow:

(a) Force an identical data flow for all customers. This will remove much of the flexibility and only a narrow set of problem types can be efficiently addressed by the VRS.
(b) Customize the data flow manually on a case-by-case basis. Here multiple model variants and use cases can be supported, but the customization heavily depends on manual work and expertise.
(c) Outsource the customization to a third party or to the customer by providing a way to externally configure the system via, e.g. a DSL. The challenge is to provide enough training and sufficient tools for the third party.
(d) Automate the customization so that fixing any given set of functionality inside the VRS is done automatically based on the customer input and data. In addition, if the provider has access to the history data during the customization, the automation might even be learning, that is, with every new modeled problem and deployment the software gains experience.

Designing the software in a way that the flexibility is maximized makes the system applicable in larger number of different contexts, thus making the approaches (c) and (d) feasible. The challenge for (c) is that many logistic operators are small, and lack the necessary expertise to understand the inherent complexity in selecting, configuring and deploying VRSs [31]. Therefore, out of these, the automation based approach (d) is the one that is more scalable and cost-effective. This validates the need for the proposed customization framework.

Each of the data conversion phases Figure 2 expose a potential point of customization. In practice, this *variability* is exposed by configurable behavior of the software system, and it needs to be managed somehow. From a theoretical perspective, this has been addressed by the techniques in the area of software product line engineering (SPLE) [35]. In SPLE, the developed system is divided into two layers: domain layer and application layer. The domain layer of the system captures the generic properties of the current domain, and the application layer is used to define customized application instances with *variability points*. It is a predefined point in the system, in which variation between the applications occur [19]. The specialized expertise required in the customization of VRSs prohibits manually managed mass customization. Instead, we suggest the use of machine-learning based adaptive mass customization techniques, and argue that these represent one of the key technologies in achieving cost-effective routing system deployment.

4 The Automation and Customization Framework

Our main contribution is an outline, or a vision, of how highly automated and easy-to-deploy VRS could be constructed. This customization framework could also enable experimentation with different automation approaches, but here we concentrate on the techniques we have either successfully applied ourselves, or see as pragmatic solutions to the presented opportunities for workflow automation. We limit ourselves to well-known methods used in related fields, and assume a generic solver module capable of expressing a wide set of "rich" VRP variants. The section follows the structure of Figure 2 with each phase having a corresponding subsection.

4.1 Interpreting Customer Data

Input data $\rightarrow$ ***Domain model***

Interpreting the customer data and transforming it into routing problem starts with the creation of a domain model, which represents the real world entities that form the routing problem. The transformation task consists of taking the problem data as an input and then extracting the data into the domain model. In the simplest case, one can specify a data format that is required and the VRS simply parses this data

into a model. It becomes problematic if the parser needs to support different formats. Maintaining numerous many-to-one mappings can quickly become an onerous task.

A likely scenario for data integration is a relational dataset, such as relational database, but in general, any kind of flat dataset with interconnected files can be used. To illustrate, one part of the dataset could consist of ordinary files that pertain to drivers and vehicles, and the other deliveries and locations. Finding semantic links between the relations in these datasets is what we call *join inference*, which in turn is based on foreign key discovery [1, 41]. We propose join inference as a model that can learn the semantic links between a set of relations. It is used to produce a cohesive union of data, the joined relation.

After join inference has been done, we propose the use of schema mapping [4] to extract the required information from the data. Schema mapping consists of finding pairings between two schemas. A *schema* is a formal description of the information contained in a relation; crudely, this would be a set of data attributes, or column headers. Having to find these attribute pairings makes the problem a kind of *data exchange problem* [22], where the goal is to take data from different sources and assimilate it, in this context, to the domain model of a VRS.

4.2 Inferring the Optimization Model

Domain model → Optimization model

After mapping the input data to the domain model, it must be translated into a format understood by the VRP solver. This includes choosing an optimization model. We were unable to find related work on automating this step. Therefore, we proceed to propose four approaches for implementing such an automated transformation:

1. **Separate models:** methods from Section 4.1 can be used on domain model to map it against a selection of optimization models. Out of these, the one with the best fit is selected and completed with instance data. This is suitable approach only if a VRS has support for a limited number of VRP variants.
2. **Coupled models:** a number of domain and optimization models are coupled together with predefined pairwise transformations. Data interpretation from Section 4.1 is done with all domain models in the coupling set and then the one with the best schema mapping (along some criteria) is selected. This has the same constraints as the previous approach.
3. **Model composition:** the optimization model is composed of different objects that may correspond to partial objectives, decisions, or constraints. Filled domain model is matched against each optimization model component and if a threshold is crossed, the component is included to the composite model.
4. **Model reduction:** alternatively, the initial optimization model may be "complete" or unified in a sense that is capable of expressing all the supported VRP features. After doing schema mapping between the domain and optimization models, the unused elements, for which the variable values were not set, are removed from the optimization model.

Besides domain model, other sources for deducing the optimization model include e.g. the vocabulary used in the data. To illustrate, the field revealing that the transportation involves people, refers to use of a dial-a-ride optimization model. The unified naming convention for VRP variants in Vidal et al [46] might prove to be useful in recognizing the different modeling constructs for the model inference. We note that the feasibility of applying automation in this phase is uncertain, mostly because of the lack of prior published research on the topic.

4.3 Selecting the Suitable Optimization Algorithms

Optimization model → Algorithm performance predictions

As mentioned earlier, industrial solutions tend to favor algorithms based on heuristics [42, 5], and many implement a collection of different algorithms to gain extra flexibility. It is also known, that the performance of an algorithm varies greatly between different routing variants and even problem instances [15]. Therefore, it is important to use an algorithm that is efficient in solving the given problem. Portfolio-based algorithm selection techniques such as SATzilla [49] use statistical models to select the algorithm for solving a given problem instance. In VRS this approach could be applied to select the higher level algorithmic components: a metaheuristic could be selected based on the instance characteristics and predicted performance.

Another way to improve solver performance is the utilization of so called *hyperheuristics*. Instead of using a single algorithm or a manually constructed combination, a hyperheuristic acts as a high level learning "supervisor" algorithm that *selects and combines* lower level algorithms from a portfolio on the fly.

Similar ideas have been tried in VRP, for example by using several simple heuristics in varied order to escape local optima. Pisinger and Ropke [34] proposed a mehtod, where adaptive heuristic selection is done among intensification and diversification heuristic operators. Garrido has proposed the use of hyperheuristics to select local search operators in solving different VRP variants [11]. VRP was also one of the problem domains in Cross-domain Heuristic Search Challenge (CHeSC2011) where a number of domain independent hyperheuristics were evaluated [47].

We note that these schemes should be useful when adapting an industrial VRS to a new set of end user provided sample problem instances. Our experimental work to explore these possibilities is in preparation.

4.4 Configuring the Optimization Algorithm

Optimization model & Observed performance → Algorithm parameter values

The algorithms used to solve hard computational problems often reveal parameters that can be used to change the behaviour of the algorithm and adapt it into solving a specific problem instance [15]. The settings of the algorithm parameters have a

substantial effect on the performance of the algorithm. However, setting them manually is a non-trivial task requiring expertise and effort trough experimentation [15]. Therefore, automatic search approaches have been proposed to what is in literature known as the problem of *automated algorithm configuration (AAC)*.

In practice, AAC can be used to automatically adapt the a routing solver for each VRS deployment. This allows the VRS provider to get the best performance out of the implemented algorithms. Also, after enough experimentation, archetypes of routing problems might emerge. With this history data the previous configuration effort could be reused to provide more varied algorithm defaults for the solver. In fact, several AAC methods have proven successful also with VRP [33]. Of particular interest in this context is the work in Becker et al [3], where they tuned the parameters of a commercial VRP solver with real-world routing problem instances. Our recent experimental work [39] verifies this and gives suggestions on which AAC methods to use to configure VRP metaheuristics.

Current state-of-the-art methods like SMAC from Hutter et al [17] or I/F-Race from Balaprakash et al [2] support all parameter types, are able to use extra information like the parameter structure, interactions or hierarchies, and use several intensification techniques that aim to save computationally expensive parameter configuration evaluations. The benefits of can be striking: Hutter et al [16] were able to achieve up to 50-fold speedup over the default parameters of the CPLEX solver.

The main challenge of applying AAC in routing, however, is that the runtime on real world routing cases may be hours, especially in presence of complex constraints [3]. Fortunately, the focus of the AAC research has been recently shifting onto overcoming these challenges, see e.g. Mascia et al [29].

4.5 Solving the VRP Problems

Optimization model & Algorithms and their parameter values → Optimization results

The solver module is responsible of performing the optimization, which contains the tasks of mapping of tasks to vehicles as well as routing the vehicles as efficiently as possible according to the objective function. The search is performed until a predefined stopping criterion has been met, or the user ends the process.

From the process perspective, the ability to predict and adjust the runtime is a major concern. The same system may be operated under tight time-constraints for planning, whereas some users prefer the added robustness of a more thorough search. It is probable that this variability is exposed e.g. as stopping criteria.

Another viewpoint to solver module customization is the availability of computational resources. In many cases, the routing system is still run in a desktop environment, but increasingly, optimization services are available through cloud services [5]. This opens a new dimension in the customization, namely the allocated computing time, resources and priority based on the customer requirements, service level agreements, and instance characteristics, which all adds in to the complexity of deploying the system.

4.6 Interpreting the Optimization Results
Optimization model & Optimization results → Domain model (solution)

The optimization solver module usually returns the resulting plan in the mathematical format it uses internally. The interpretation of the optimization results has a direct connection to the construction of the optimization model. Whereas in model construction the decision variables are selected based the data in the domain model, in this phase the values of the decision variables need to be interpreted back to the relations and values of the entities in the domain model. We can use an inverse transformation of the one in Subsection 4.2 to decode the solution.

One issue in the interpretation of the results is the type of the decoding. It may be that the decoding is not one-to-one. That is, there may be multiple possible plans the optimized solution can be decoded to. For example, in a classical VRP all the trucks are identical and it does not matter how the vehicles and routes are mapped Puranen [36]. This potential unambiguity may have undesired consequences if it is not taken into account.

4.7 Producing a Formatted Plan
Domain model (solution) → Output plan

Ultimately the user of a VRS needs to apply the plan into practice. Different users have different formats, output data requirements, and reporting needs, so in the final data transformation step an automated VRS could adapt its output to the format most convenient to the end user.

If the interfaced system includes plan generation, it could be enough to do the schema mapping procedure from Section 4.1 in reverse. The existing system would then compose the output document to that is to be handed to the drivers. Another option is to infer the structure of an example document using methods such as table extraction, visual object and information extraction, and entity identification [27, 25]. This produces a template which then can be filled with the relevant data from the solver. Similar technique has been used, for example, in web page content and structure extraction to reformat the web page content for mobile clients [25].

5 Preliminary Experimental Results

To demonstrate the potential of automatic configuration of route optimization algorithms, we configured the three metaheuristics (Record-to-Record travel *RTR*, simulated annealing *SA*, and ejection *EJ*) provided by the VRPH library [12] on four real world based benchmark instances from the literature. For details of the experimental setup see Rasku et al [39].

The target problem instances were `F-n45-k4`, `F-n72-k4`, and `F-n135-k7` from [10] with 45, 72 and 135 requests and the 385 request instance `tai385` from Taillard [43]. The `tai385` instance is generated based on the locations and census of population data from canton of Vaud in Switzerland, whereas the instances `F-n45-k4` and `F-n135-k7` are from a day of grocery deliveries from the Ontario terminal of National Groceries Limited. The `F-n72-k4` instance data is obtained from Exxon associated case involving the delivery of tires, batteries and other accessories to gas stations. We used SMAC [17] (version 2.3.5) and Iterated F-Race [2] implementation described in [28] (version 1.0.7) and restricted to evaluation budget of 500 invocations. Each metaheuristics was configured separately for each of the target problem instance. A 30 second cutoff was used for the solvers.

Table 1: Average AAC results for all solver-instance pairs.

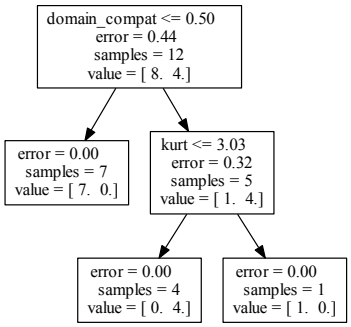
Results are given as percentage from the best known solution (relative optimality gap). Statistically better ($p < 0.05$) results are bolded, evaluation budget violations of more than 5% are in italics, and the standard deviation over 100 VRP solutions is given in parentheses.

	Quality on the target instance			Quality on the other instances		
Target	Defaults	I/F-Race	SMAC	Defaults	I/F-Race	SMAC
F-n45 EJ	0.49 (0.35)	**0.12 (0.23)**	**0.15 (0.25)**	2.57 (2.19)	**2.21 (1.41)**	2.70 (2.07)
F-n45 RTR	0.48 (0.40)	0.01 (0.04)	**0.00 (0.00)**	11.25 (0.40)	**5.32 (3.01)**	**6.02 (3.56)**
F-n45 SA	0.30 (0.34)	**0.03 (0.14)**	**0.01 (0.09)**	8.91 (1.54)	**6.55 (4.97)**	**7.68 (6.70)**
F-n72 EJ	0.99 (2.15)	**0.19 (1.03)**	**0.16 (1.11)**	**1.98 (0.54)**	2.15 (0.88)	2.11 (0.82)
F-n72 RTR	6.63 (0.28)	**0.00 (0.00)**	**0.00 (0.00)**	4.94 (0.51)	**3.86 (1.01)**	**3.66 (1.02)**
F-n72 SA	3.80 (1.75)	**0.05 (0.15)**	**0.02 (0.09)**	5.06 (0.52)	**2.66 (1.11)**	**3.06 (1.41)**
F-n135 EJ	0.24 (0.29)	**0.19 (0.28)**	**0.17 (0.15)**	2.96 (2.58)	2.01 (1.61)	**1.88 (0.92)**
F-n135 RTR	1.62 (0.07)	0.06 (0.08)	**0.02 (0.03)**	9.94 (0.57)	**4.71 (3.00)**	5.65 (2.17)
F-n135 SA	0.11 (0.07)	*0.14 (0.14)*	**0.08 (0.06)**	8.99 (1.57)	***6.42 (3.65)***	**6.25 (2.89)**
tai385 EJ	1.23 (0.28)	1.10 (0.23)	**1.02 (0.18)**	1.92 (2.46)	**0.72 (0.53)**	**0.76 (0.43)**
tai385 RTR	2.91 (0.27)	1.00 (0.22)	**0.88 (0.18)**	8.61 (0.48)	3.99 (2.09)	**3.47 (1.85)**
tai385 SA	4.67 (0.40)	***1.04 (0.24)***	1.18 (0.27)	3.74 (2.06)	***2.15 (2.51)***	5.79 (4.63)

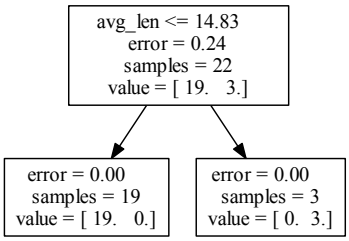
Results of the configuration runs are presented in Table 1. On average, the quality of the results was improved by 1.65 percentage points with the use of AAC, which means that the relative optimality gap was closed by 73%. Furthermore, the performance of the metaheuristics was more consistent when configured, as can be observed from the standard deviations.

Because the metaheuristics were configured for each instance separately, we acknowledge the danger of overtuning [16]. To observe the effect, the rightmost three columns of Table 1 present the performance of the resulting parameter configurations on the other three remaining instances. These columns can be interpreted as the result of a 4-fold cross-validation. The configurators overfit only for the `F-n72-k4`, and for all other targets the solution quality is statistically significantly improved on average by 2.4 percentage points (a 37% improvement). The solver behavior becomes slightly more erratic as can be perceived from the standard deviations. How-

ever, this is likely to be a byproduct of the improved solution quality and the more rugged fitness landscape of a multi-instance problem set. As suggested by the results in [39], if tuned on the entire instance set, the robustness of the solvers on similar instances is expected to improve.



(a) Capacity



(b) Location

Fig. 3: Decision trees for the schema mapping of two domain model attributes

Our proposed solution to increase the level of automation in the data import phase is presented in [20]. To summarize, Kalmbach [20] provided a formulation for the data import and model inference problem, presented a decision trees [38] based approach for join inference and schema mapping, and explored its applicability in importing of schemaless routing instance data. Two decision trees for *capacity* and request *location* mapping are provided in Figure 4.4 as an illustration of the generated inference rules. The proof-of-concept tool is able to recognize the nature of each column in a column-oriented input for the generated test data, and is thus capable of generating simple mapping rules between input and the domain model.

6 Conclusions

Vehicle routing systems provide several advantages over manual transportation planning, but the deployment of these systems is in many cases laborious and costly. In addition, migration from the current system with associated customization and integration challenges create practical obstacles that prevent the latest advances in operations research from being disseminated to wide use. The focus in academic research is in modeling and solving efficiency whereas in commercial routing systems usability, flexibility, and scalability are more important. Tighter interaction between the two is needed in order to effectively solve real-world routing problems [5].

The advances in technologies such as GPS and RFID, and drop in data warehousing prices, have made transportation big data collection possible and econom-

ical. Concurrently, logistics operators have begun to see the information as a vital asset that can be used in decision making. This opens new possibilities for machine learning, for which the accuracy is dependent of the amount and availability of data that can be used to train the models. Therefore, these trends have paved the road for a new generation of vehicle routing systems that can utilize machine learning to automate the customization and deployment. This in turn has the potential to increase the effectiveness and robustness as the system can be adapted automatically to the particularities of a problem instance. The goal is to diminish the importance of an operations researcher in the deployment process and consequently to permit higher scalability and more widespread deployment of route optimization.

In this paper, we have outlined a customization framework for the automation of data transformation operations inside a routing system. Our framework recognizes seven transformation steps, each open for system customization. We also provide suggestions on automating these steps. Our preliminary empirical results are promising, but further experimental work is required to establish whether all the proposed techniques are fully applicable in practice.

To evaluate the proposed customization framework, we reflect it against the framework for analyzing VRS deployment published in [31]. Neittaanmäki and Puranen [31] recognize several practical adoption and deployment barriers for the VRSs. They see the involvement of an optimization expert as a prohibiting investment and call for an increased automation of the deployment process. Their deployment process is split into three phases: data, process, and system integration. To see in which extent our proposed customization framework can resolve the 18 barriers they recognized, we proceed to give some possible solutions to the recognized issues: In data integration step, the missing, low quality and incomplete data could be automatically imputed, or at least recognized with machine learning. The data structure inference from Subsection 4.1 can help when acquiring and combining the data from existing systems. In addition, because of the techniques proposed in Subsection 4.2, it takes less expertise to generate the optimization model. As demonstrated by our experiments, the plan quality can be improved, sometimes significantly, using automatic algorithm configuration (Subsection 4.4). Use of automation results into lower perceived complexity and improved usability that can instill trust in the users to the system and to the plans it generates. On the system integration level, the automation makes integration easier and faster, which in turn can make the system deployment cheaper, less dependent on expertise and other resources, and flexible to the current and future changes in operations.

Taken together, we argue that in order to bring the latest academic routing knowledge to the hands of logistics operators in a massive scale, the automatic configuration approach, as presented in this paper, is needed. The recent trends in VRP research seem to converge towards generic reusable modeling and highly adaptive and configurable modeling frameworks, but we have shown that several other areas in practical system integration and deployment need to be considered in order to effectively apply these into practice. This requires extensive further studies in several disciplines, but should provide a promising area of research with a potential for a wide array of practical benefits.

References

[1] Acar AC, Motro A (2009) Efficient discovery of join plans in schemaless data. In: Proceedings of the 2009 International Database Engineering & Applications Symposium, ACM, New York, NY, USA, IDEAS '09, p 111

[2] Balaprakash P, Birattari M, Stützle T (2007) Improvement strategies for the F-Race algorithm: Sampling design and iterative refinement. Tech. rep., IRIDIA, Université Libre de Bruxelles

[3] Becker S, Gottlieb J, Stützle T (2006) Applications of racing algorithms: an industrial perspective. In: Proceedings of the 7th international conference on Artificial Evolution, Springer-Verlag, Berlin, Heidelberg, EA'05, pp 271–283

[4] Bellahsene Z (2011) Schema Matching and Mapping. Springer

[5] Bräysy O, Hasle G (2014) Software Tools and Emerging Technologies for Vehicle Routing and Intermodal Transportation, SIAM, chap 12, pp 351–380. MOS-SIAM Series on Optimization

[6] Cordeau JF, Gendreau M, Hertz A, Laporte G, Sormany JS (2005) New heuristics for the vehicle routing problem. In: Logistics Systems: Design and Optimization, Springer-Verlag, New York, chap 9, pp 279–297

[7] Dantzig GB, Ramser JH (1959) The truck dispatching problem. Management Science 6(1):80–91

[8] Desrochers M, Jones CV, Lenstra JK, Savelsbergh MWP, Stougie L (1999) Towards a model and algorithm management system for vehicle routing and scheduling problems. Decision Support Systems 25(2):109–133

[9] Drexl M (2011) Rich vehicle routing in theory and practice. Tech. Rep. 1104, Gutenberg School of Management and Economics, Johannes Gutenberg University Mainz

[10] Fisher ML (1994) Optimal solution of vehicle routing problems using minimum k-trees. Operations Research 42(4):626–642

[11] Garrido P, Riff MC (2010) DVRP: a hard dynamic combinatorial optimisation problem tackled by an evolutionary hyper-heuristic. Journal of Heuristics 16(6):795–834

[12] Groër C, Golden B, Wasil E (2010) A library of local search heuristics for the vehicle routing problem. Mathematical Programming Computation 2(2):79–101

[13] Hasle G, Kloster O (2007) Industrial vehicle routing. In: Geometric modelling, numerical simulation, and optimization, Springer, pp 397–435

[14] Hoff A, Andersson H, Christiansen M, Hasle G, Løkketangen A (2010) Industrial aspects and literature survey: Fleet composition and routing. Computers & Operations Research 37(12):2041–2061

[15] Hoos HH (2012) Automated algorithm configuration and parameter tuning. In: Autonomous Search, Springer, pp 37–71

[16] Hutter F, Hoos HH, Leyton-Brown K (2010) Automated configuration of mixed integer programming solvers. In: CPAIOR, Springer, Lecture Notes in Computer Science, vol 6140, pp 186–202

[17] Hutter F, Hoos H, Leyton-Brown K (2011) Sequential model-based optimization for general algorithm configuration. In: Learning and Intelligent Optimization, Springer, pp 507–523
[18] Irnich S (2008) A unified modeling and solution framework for vehicle routing and local search-based metaheuristics. INFORMS Journal on Computing 20(2):270–287
[19] Jacobson I, Griss M, Jonsson P (1997) Software reuse: architecture, process and organization for business success. ACM Press/Addison-Wesley Publishing Co. New York, USA
[20] Kalmbach A (2014) Fleet inference : importing vehicle routing problems using machine learning. Master's thesis, University of Jyväskylä, Department of mathematical information technology
[21] Kleijn MJ (2000) Tourenplanungssoftware: ein vergleich für den niederländischen markt. Internationales Verkehrswesen 52(10):454–455
[22] Kolaitis PG (2005) Schema mappings, data exchange, and metadata management. In: Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, ACM, New York, NY, USA, PODS '05, p 6175
[23] Kotthoff L (2014) Algorithm selection for combinatorial search problems: A survey. AI Magazine 35(3):48–60
[24] Krueger CW (2002) Easing the transition to software mass customization. In: Linden F (ed) Software Product-Family Engineering, Lecture Notes in Computer Science, vol 2290, Springer Berlin Heidelberg, pp 282–293
[25] Krüpl-Sypien B, Fayzrakhmanov RR, Holzinger W, Panzenböck M, Baumgartner R (2011) A versatile model for web page representation, information extraction and content re-packaging. In: Proceedings of the 11th ACM symposium on Document engineering, ACM, New York, USA, pp 129–138
[26] Laporte G (2007) What you should know about the vehicle routing problem. Naval Research Logistics 54(8):811–819
[27] Lin X, Hui C, Nelson G, Durante E (2006) Active document versioning: from layout understanding to adjustment. In: Taghva K, Lin X (eds) Document Recognition and Retrieval XIII, SPIE, SPIE Proceedings, vol 6067
[28] López-Ibánez M, Dubois-Lacoste J, Stützle T, Birattari M (2011) The irace package, iterated race for automatic algorithm configuration. Tech. rep., IRIDIA, Université Libre de Bruxelles
[29] Mascia F, Birattari M, Stützle T (2013) An experimental protocol for tuning algorithms on large instances. In: Learning and Intelligent Optimization, Springer
[30] Maturana S, Ferrer JC, Barañao F (2004) Design and implementation of an optimization-based decision support system generator. European Journal of Operational Research 154(1):170–183
[31] Neittaanmäki P, Puranen T (2015) Scalable deployment of efficient transportation optimization for smes and public sector. In: Advances in Evolutionary and Deterministic Methods for Design, Optimization and Control in Engineering and Sciences, Springer, pp 473–484

[32] Partyka J, Hall R (2012) Software survey: Vehicle routing. OR/MS Today 39(1)
[33] Pellegrini P, Birattari M (2006) The relevance of tuning the parameters of metaheuristics. Tech. rep., IRIDIA, Université Libre de Bruxelles
[34] Pisinger D, Ropke S (2007) A general heuristic for vehicle routing problems. Computers & Operations Research 34(8):2403–2435
[35] Pohl K, Böckle G, van der Linden FJ (2005) Software Product Line Engineering: Foundations, Principles and Techniques. Springer
[36] Puranen T (2011) Metaheuristics meet metamodels – a modeling language and a product line architecture for route optimization systems. PhD thesis, University of Jyväskylä, Jyväskylä studies in computing;1456-5390;134
[37] Puranen T (2012) Producing routing systems flexibly using a VRP metamodel and a software product line. In: Operations Research Proceedings 2011, Springer, pp 407–412
[38] Quinlan JR (1986) Induction of decision trees. Machine learning 1(1):81–106
[39] Rasku J, Musliu N, Kärkkäinen T (2014) Automating the parameter selection in VRP: An off-line parameter tuning tool comparison. In: Modeling, Simulation and Optimization for Science and Technology, Computational Methods in Applied Sciences, vol 34, Springer, pp 191–209
[40] Ropke S, Pisinger D (2006) A unified heuristic for a large class of Vehicle Routing Problems with Backhauls. European Journal of Operational Research 171(3):750–775
[41] Rostin A, Albrecht O, Bauckmann J, Naumann F, Leser U (2009) A machine learning approach to foreign key discovery. In: 12th International Workshop on the Web and Databases (WebDB)
[42] Sörensen K, Sevaux M, Schittekat P (2008) Multiple neighbourhood search in commercial VRP packages: Evolving towards self-adaptive methods. In: Adaptive and multilevel metaheuristics, Springer, pp 239–253
[43] Taillard E (1993) Parallel iterative search methods for vehicle routing problems. Networks 23(8):661–673
[44] Toth P, Vigo D (eds) (2002) The vehicle routing problem. SIAM
[45] Vidal T, Crainic TG, Gendreau M, Prins C (2012) A unified solution framework for multi-attribute vehicle routing problems. Tech. rep., CIRRELT
[46] Vidal T, Crainic TG, Gendreau M, Prins C (2013) Heuristics for multi-attribute vehicle routing problems: A survey and synthesis. European Journal of Operational Research 231(1):1 – 21
[47] Walker JD, Ochoa G, Gendreau M, Burke EK (2012) Vehicle routing and adaptive iterated local search within the hyflex hyper-heuristic framework. In: Learning and Intelligent Optimization - 6th International Conference, Springer, Lecture Notes in Computer Science, vol 7219, pp 265–276
[48] Welch PG, Ekárt A, Buckingham C (2011) A proposed meta-model for combinatorial optimisation problems within transport logistics. In: MIC 2011: The IX Metaheuristics International Conference, vol IX
[49] Xu L, Leyton-brown K (2008) SATzilla : Portfolio-based Algorithm Selection for SAT. Artificial Intelligence 32:565–606