The Art of Negotiation: Developing Efficient Agent-Based Algorithms for Solving Vehicle Routing Problem with Time Windows

Petr Kalina[1], Jiří Vokřínek[2], and Vladimír Mařík[3]

[1] Intelligent Systems Group, Czech Technical University in Prague, Czech Republic
petr.kalina@fel.cvut.cz
[2] Agent Technology Center, Czech Technical University in Prague, Czech Republic
jiri.vokrinek@fel.cvut.cz
[3] Department of Cybernetics, Czech Technical University in Prague, Czech Republic
vladimir.marik@fel.cvut.cz

Abstract. We present an ongoing effort in developing efficient agent-based algorithms for solving the vehicle routing problem with time windows. An abstract algorithm based on a generic agent decomposition of the problem is introduced featuring a clear separation between the local planning performed by the individual vehicles and the global coordination achieved by negotiation. The semantics of the underlying negotiation process is discussed as well as the alternative local planning strategies used by the individual vehicles. Finally a parallel version of the algorithm is presented based on efficient search diversification and intensification strategies. The presented effort is relevant namely for (i) yielding results significantly improving on all previous agent-based studies, (ii) the inclusion of relevant widely-used benchmarks missing from these studies and (iii) the breadth and depth of the provided evidence and analysis including relevant comparison to the state-of-the-art centralized solvers.

Keywords: multi-agent systems, transportation logistics, optimization.

1 Introduction

The vehicle routing problem with time windows (VRPTW) is one of the most widely studied problems in the the area of operation research (OR), featuring in many real-world logistics or supply chain management applications. It is a problem of finding a minimal set of routes starting and ending at a single depot serving a set of geographically scattered customers, each within a specific time-window and with a specific demand of goods. The VRPTW has been extensively studied for nearly thirty years. Due to the high complexity of the problem itself as well as it's numerous applications the emphasis has been put in developing efficient heuristic algorithms providing good quality solutions in reasonable time, with most successful algorithms being centralized and monolithic e.g. [10,12].

The multi-agent based solving approaches, on the other hand, have not received much attention by the OR community. However, the multi-agent systems

V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 187–198, 2013.

are an emerging prominent architecture with respect to modeling new-generation systems based on smart actors and their intelligent coordination, promoting the autonomy of the actors and the clear separation of local decision making and the global cooperation that is beneficial in many real world scenarios. With the shift seen all over the industry towards the internet of things, smart objects and decentralized control systems, this work represents an effort to provide a sound alternative to the *classical*, centralized algorithms. Central to the presented effort is the focus on performance, analysis and relevant comparison to the state-of-the-art classical algorithms missing from previous similar studies.

2 Problem Statement and Notations

Let $\{1..N\}$ represent the set of customers with the depot denoted as 0. For each customer c_i let (e_i, l_i, s_i, d_i) denote the earliest/latest service start times (*time window*), service time and the demand. Let a sequence of customers $\langle c_0, c_1, ..c_m, c_{m+1}\rangle$ denote a single route with c_0 and c_{m+1} denoting the depot. Let D denote the vehicle capacity.

The objective of the VRPTW is finding a minimal set of routes serving all customers. For each route $\langle c_0, c_1, ..c_m, c_{m+1}\rangle$ the sum of corresponding customers' demands must be lower than the capacity of the vehicle serving the route $\sum_1^m d_i \leq D$ (capacity constraint) while the service at each customer c_i must begin within the interval given by (e_i, l_i) (time-windows constraints).

3 Related Work

As mentioned, the VRPTW has been extensively studied for many years. We refer the reader to the excellent surveys of the classical methods [2,3,14] — both exact and approximate — and focus mainly on relevant agent-based studies. The performance of individual algorithms is evaluated using the well known *cumulative number of vehicles* (CVN) metric, corresponding to the total number of routes across all problem instances over the corresponding benchmark.

3.1 Classical Algorithms

For the smaller Solomon's benchmark (100 customers) the best-known overall result was presented by [10] with a CVN of 405. The algorithm is based on the *ejection pools* principle, performing very good potentially unfeasible insertions of customers to individual routes and subsequently recovering the feasibility by ejecting some other customers from the unfeasible routes. The insertion-ejection phase is interleaved with a local search procedure dynamically improving the solution throughout the solving process.

An improved algorithm presented in [12] is based on similar concepts. An ejection-pools mechanism is used accompanied by a powerful insertion method denoted as *squeeze* as well as a search diversification *perturb* procedure. The *squeeze*

method also employs a specific adaptive local search procedure used to repair potentially unfeasible intermediate solution using heuristic carried over from the previously mentioned work. The algorithm achieves a CVN of 10290 over the extended Homberger's benchmark (200–1000 customers).

3.2 Agent-Based Algorithms

As mentioned above, none of the previous agent-based studies were particularly successful. The common pitfalls were the lack of relevant comparison to the state-of-the-art algorithms and a generally weak performance.

The algorithm presented in [5] builds on the concepts of a Shipping Company and a Shipping Company Truck. The planning is based on the well known contract net protocol (CNP) accompanied by a *simulated trading* improvement strategy based on [1]. No relevant performance assessment is provided.

The algorithm presented by [9] is based on agents representing the customers, individual routes and a central planner agent. A sequential insertion procedure based on Solomon's I1 heuristic is followed by an improvement phase in which the agents propose moves gathered in a "move pool" with the most advantageous move being performed. Also a route elimination routine is periodically invoked which is not well described in the text. The algorithm achieves a CVN of 436 over the Solomon's benchmark. No relevant runtime information is provided.

In [4] an algorithm is introduced based on Order Agent — Scheduling Agent — Vehicle Agent hierarchy. The algorithm is based on a modified CNP insertion procedure limiting the negotiation to agents whose routes are in proximity of the task being allocated, focusing on the efficiency of the negotiation process rather than solution quality. No relevant performance information is provided.

4 Algorithm for VRPTW Based on Agent Negotiation

This work thus represents a rigorous effort aimed at developing competitive agent-based VRPTW algorithms. An abstract negotiation framework is presented featuring the clear separation between the local planning of individual vehicles and the global coordination achieved by negotiation. The alternative negotiation semantics are discussed as well as the alternative local planning strategies. Finally a parallel algorithm is presented. Only brief overview is provided, referring the reader to our previous works for further details [7,8].

4.1 Abstract Algorithm

Underlying the negotiation based solving process is a fitting agent-based decomposition of the solved problem, featuring a top layer represented by a Task Agent, middle layer represented by an Allocation Agent and a fleet of Vehicle Agents present at the bottom level of the architecture.

Task Agent acts as an interface between the algorithm's computational core and the surrounding infrastructure. It is responsible for registering the tasks and submitting them to the underlying Allocation Agent.

`Input:` Ordered set of customers C, Fleet of empty vehicles — initial solution σ
`Output:` Solution σ — complete or partial based on success of the process

Procedure *negotiate*(C, σ)
1: Init reallocate counters $r[c] := 0$ for all $c \in C$;
2: **while** (exists$(c \in C)$, $r[c] \leq reallocationLimit$)
3: $dynamicImprove(\sigma)$;
4: Select first $t \in \{c \in C, r[c]$ minimal$\}$;
5: $I := \{c \in \sigma$, $costCommit(c, v)$ is minimal$\}$;
6: **if** $(I \neq \emptyset)$ **then**
7: Randomly select $v \in I$;
8: $commit(c, v)$;
9: remove c from C;
10: **else**
11: $r[c] := r[c] + 1$;
12: **endif**
13: **endwhile**
14: $finalImprove(\sigma)$;
15: **return** σ;

Fig. 1. The Abstract Global Coordination Process

Allocation Agent instruments the actual solving process by negotiating with the Vehicle Agents. The negotiation is conducted based upon task commitment and decommitment cost estimates provided by the Vehicle Agents.

Vehicle Agent represents an individual vehicle serving a route. It provides the Allocation Agent with the above mentioned inputs. These are computed based on local (private) Vehicle Agent's plan processing.

Figure 1 illustrates the abstract global coordination negotiation process instrumented by the Allocation Agent. In essence it corresponds to a series of *negotiation* interactions between the Allocation Agent and the vehicles represented by the Vehicle Agents. The customers are allocated to individual vehicles based on the commitment cost estimates provided by the vehicles (lines 5–10) computed based upon the particular local planning strategy being used.

Within the *dynamic* or the *final* improvement phases (lines 3, 14) the partial solution being constructed can also be modified in a series of further interactions between the agents in order to e.g. escape local minima, address secondary optimization criteria etc. Thus a particular *negotiation semantics* is given by specifying the exact semantics for the dynamic and final improvement phases.

Both the underlying architecture and the negotiation based coordination process are abstract. On the other hand the particular local planning strategies used by the individual vehicles may reflect the potentially rich semantics of the solved real-world problem e.g. heterogeneity of the fleet, loading constraints, complex optimization objectives etc. We argue that due to the clear separation between the abstract global coordination achieved by negotiation and the local planning strategies used by the individual vehicles the agent-based problem decomposition presents some key advantages for modeling specific real-world environments.

Upon termination, in case there are still some unserved customers (solution σ is not complete), the process can be restarted with a different fleet size or an additional vehicle can be added. Determining a fitting fleet size for the initial solution σ is thus a significant factor affecting the algorithm's efficiency, as is the ordering of the customers within the set C. These two attributes are subject to meta-optimization within the parallel algorithm discussed later in the text.

4.2 Negotiation Semantics

One of the core research objectives behind the presented effort is to explore the possible semantics of the negotiation process and the alternative local planning strategies in an effort to provide an efficient algorithm for the VRPTW.

We considered three general algorithm settings with respect to the used negotiation semantics: (i) *Algorithm-B* baseline setting with neither dynamic nor final improvement phases being employed, (ii) *Algorithm-FI* with only the final improvement being employed and finally (iii) the full fledged *Algorithm-DI* featuring both improvement phases.

Three alternative improvement methods were considered with respect to the *dynamic* and *final* improvement phases, described in detail in [7]. The basic semantics in all three cases is that each vehicle identifies some subset of its customers and each of these customers is then allocated in an auction process to the vehicle with the cheapest commitment cost estimate. We refer to this action as a *reallocation* of a single customer. Thus the three methods differ in the choice of the particular subset of customers to be reallocated within each route.

4.3 Local Planning Strategies

Two particular local planning strategies were considered described in [7]. Both are based on the well known *cheapest insertion* heuristic principle. The *travel time savings* (TTS) heuristic is notoriously known to the OR community. The commitment cost estimates correspond the the increase in travel time caused by the insertion of the corresponding customer to the route. The TTS heuristic thus leverages the spatial aspects of the problem, preferring customers in proximity of the corresponding agent's route irrespective of their time-windows properties. It has been shown [15,11] however, that a strategy exploiting also the temporal relations of the individual deliveries can yield significantly better results.

Thus the *slackness savings heuristic* (SLS) introduces elements to the cost structure based on the constraining effects the insertion of a customer has on the corresponding route schedule. Consider a route with customers having wide time-windows spaced within the route in such a way that the service at each customer starts at the beginning of the corresponding time-window. Then possibly a detour can be made prior to serving any of the customers with the resulting shift in the route schedule not resulting in the unfeasibility of the route. Now imagine adding a customer to the end of the route with a very short time-window. The considered potential detours are no longer feasible. This corresponds to a reduction of the route *slackness* — a situations the SLS heuristics effectively helps to avoid.

4.4 Algorithm Complexity

Given a problem with N customers, the abstract algorithm complexity is

$$O^{vn} + O^{ord} + N \times \left(O^{dyn} + O^{alloc}\right) + O^{fin} \tag{1}$$

where O^{vn} is the complexity of estimating the initial fleet size, O^{ord} being the complexity of reordering the set of customers C prior to the solving process. The O^{dyn} and O^{fin} correspond to the complexities of the *dynamic* and *final* improvement phases while O^{alloc} represents the complexity of the auction process of finding the agent with the best commitment cost estimate.

The complexity of the algorithm thus corresponds to the particular negotiation semantic and local planning strategy being used. In the full *Algorithm-DI* setting the resulting algorithm worst case complexity is $O(N^3)$ in case of the TTS heuristic and $O(N^4)$ for the SLS heuristic [7].

5 Parallel Algorithm with Diversification and Intensification Strategies

As already mentioned, the important factors contributing to the efficiency of the algorithm are (i) the improvement methods to be used within the abstract negotiation process, (ii) the used local planning strategy and (iii) the used initial ordering of the set of customers C.

By analyzing the three discussed general algorithm settings, we discovered that the algorithm is sensitive to the choice of the initial customer ordering in all three cases. The sensitivity was most pronounced with the simpler *Algorithm-B* and *Algorithm-FI* settings and less so with the full *Algorithm-DI* setting. Interestingly, none of the orderings was dominant across all problem instances. To the contrary, each ordering performed well on a different subset of problems instances, suggesting that the instances differ in their nature favoring some particular orderings. Given a specific problem instance we thus found that (i) some orderings perform well for both the simple and the complex algorithm settings, (ii) the best results are most consistently found by the more complex settings over these orderings and (iii) that using these orderings even the simpler settings often return very good results.

Let the term *particular algorithm* denote an algorithm with a specific negotiation semantic and local planning strategy. Given a set of customer orderings Ω and a set of particular algorithms Δ the tuple $[O \in \Omega, A \in \Delta]$ corresponds to a single executable *algorithm instance* within the *algorithm configuration space* $\Omega \times \Delta$. The introduced parallel algorithm is thus based on traversing the *diversified algorithm configuration space* $\Omega' \times \Delta'$. In order to exploit the characteristics of the agent-based algorithm discussed above a specific search diversification and search intensification strategies were introduced. A *diversified set of particular algorithms* Δ' was tailored based on extensive experimentation and tuning of parameters using all three discussed algorithm settings. Likewise a *diversified*

Table 1. Solution quality comparison to the state-of-the-art algorithms

Size	*Classical* [10,12]	*Agents* [9]	*Algorithm-DI*	*Parallel*
100	405	+31 (7.7%)	+24 (5.9%)	+16 (4.0%)
200	694	–	+21 (3.0%)	+12 (1.7%)
400	1380	–	+38 (2.8%)	+29 (2.1%)
600	2065	–	+56 (2.7%)	+43 (2.1%)
800	2734	–	+89 (3.3%)	+71 (2.6%)
1000	3417	–	+115 (3.4%)	+83 (2.4%)
All	10695	–	+343 (3.2%)	+254 (2.4%)

set of orderings Ω' was generated from a set of canonical analytically sound orderings using two specific *ordering diversification operators* introduced in [8].

The parallel algorithm traverses the set Δ' starting with the simplest settings and moves towards the most complex ones. For each particular algorithm $A_i \in \Delta'$ the whole set Ω' is traversed with all the corresponding algorithm instances being executed in parallel. Then, prior to processing the next particular algorithm $A_{i+1} \in \Delta'$, the set Ω' is pruned and some orderings are discarded based on the specific *ordering pruning strategy* being used. Three alternative pruning strategies were introduced — the *BP* (Basic Pruning) strategy, the *CSP* (Minimal Covering Set) strategy and the hybrid *CSP+BP* strategy discussed in [8].

Also importantly, the parallel algorithm also efficiently mitigates the previously identified performance penalty resulting from restarting the negotiation process with an increased fleet size when the resulting solution σ is not complete, having a multiplicative effect on the resulting complexity [7]. This is achieved by (i) executing individual algorithm instances with the initial fleet size always targeting a new best found solution and (ii) terminating them prematurely in case they are not likely to yield such an improvement.

Thus the parallel algorithm uses an efficient *search diversification strategy* based on traversing the diversified configuration space $\Omega' \times \Delta'$ generated using two specific ordering diversification operators and by using alternative negotiation semantics within the abstract algorithm negotiation process. The inherent complexity increase is partially offset by an efficient *search intensification strategy* consisting of ordering pruning, improved restarts strategy and terminating the not promising algorithm instances. For further details refer to [8].

6 Experimental Validation

The experimental evaluation is based on the two widely used benchmarks of Homberger and Solomon [6,15]. Together these benchmarks provide a total of 356 problem instances of 6 different sizes of 100 – 1000 customers featuring 6 instance types differentiated by topology and time windows properties.

Table 2. Runtime comparison to the state-of-the-art algorithms

	Nagata[12]	*Lim*[10]	*Parallel*	
Size	Avg. RT	Avg. RT	Avg. RT	Anytime RT
200	1 min	10 min	10 s	57 ms
400	1 min	20 min	2 min	300 ms
600	1 min	30 min	8 min	2 s
800	1 min	40 min	24 min	7 s
1000	1 min	50 min	54 min	14 s

6.1 Overall Solution Quality Analysis

The comparison to the state-of-the-art algorithms in terms of the primary optimization criteria is presented by Table 1. The results are listed for individual problem instance sizes. The *Classical* [10,12] and the *Agents* [9] columns correspond to the state-of-the-art classical and agent-based algorithms. The *Algorithm-DI* column corresponds to the full *Algorithm-DI* setting combined with the SSL local planning strategy as presented in [7]. The last column corresponds to the parallel algorithm with the *CSP+BP* pruning strategy from [8]. The results correspond to the *cumulative number of vehicles* (CVN) for the second column and the respective absolute and relative error for the remaining columns.

In overall the parallel algorithm achieved a CVN of 10949 over all the benchmark instances, corresponding to a 2.4% relative error with respect to the state-of-the-art centralized algorithms, equalling the best known results for 64% of the problem instances. This represents a significant improvement over all previously presented agent based algorithms.

6.2 Runtime and Convergence Analysis

The comparison in terms of runtime with the state-of-the-art algorithms is presented by Table 2. The listed values correspond to the average runtime for individual instance sizes. The last two columns correspond to the *average composite runtime* — the sum of runtimes of all algorithm instances — and the *average anytime runtime* — the time when the best solution was first found considering a parallel execution of the competing instances within the parallel algorithm.

The results illustrate exceptional anytime convergence of the parallel algorithm, with the time before the best solution is found outperforming even the state-of-the-art solvers. The composite runtime is also competitive. We must note, however, that: (i) compared algorithms outperform presented algorithm in terms of CVN and (ii) are not computationally bound.

6.3 Negotiation Semantics Analysis

The abstract algorithm enables for a number of alternative particular algorithms to be tailored based on the particular negotiation semantics and the local

Table 3. Alternative particular algorithms comparison

Algorithm Setting	Slackness Savings	Travel Time Savings
Algorithm-B	7.6%	23.1%
Algorithm-FI	5.5%	11.1%
Algorithm-DI	3.0%	5.2%

planning strategy being used. To provide an insight into the influence of the used configuration to the resulting solution quality we assessed six relevant particular algorithms based on the three general algorithm settings and the two introduced local planning strategies. The results — corresponding to the relative error in terms of CVN over the 200 customer instances — are listed in Table 3.

With the *Algorithm-B* setting there is no possibility to recover from a potentially bad customer allocations taking place in the early stages of the solving process. For example, an early allocation may render some of the subsequent allocations unfeasible due to the time window or capacity constraints, effectively preventing some parts of the search space from being traversed. The *Algorithm-FI* setting extends the *Algorithm-B* setting by allowing some customer reallocations during the final stage of the allocation process. At this stage, however, the partial solution σ is already tightly constrained and the chance of reallocating a customer within σ is correspondingly small. The full *Algorithm-DI* setting significantly outperforms the *Algorithm-FI* setting. Arguably this is due to the fact that the improvements are performed dynamically throughout the allocation process on smaller and therefore less constrained partial solutions.

In overall, our experiments proved that the number of successful reallocations within the dynamic and the final improvement phases is actually very limited, with the success ratio dropping significantly towards the end of the solving process. This further highlights the fact that it is very difficult to escape local minima once the solution σ gets denser. We argue that this could be effectively addressed by introducing extensions to the negotiation semantics allowing for more complex *trading moves* than simple task reallocations. Our preliminary experiments — not part of this study — suggest that introducing a semantics enabling the vehicles to bid even for such customers that don't fit into the corresponding routes (due to capacity or time-window constraints) but at prices reflecting also the necessary ejection of some other customers from the route might yield interesting opportunities.

6.4 Local Planning Strategies Analysis

Table 3 also illustrates the relative success of the two proposed local planning strategies. The results show that the SLS strategy outperforms the TTS strategy in all examined algorithm settings. The SLS strategy is based on estimating the negative effects of the insertions in terms of reduction to the slackness of the corresponding routes possibly preventing future advantageous detours. Especially when applied iteratively throughout the solving process — within the dynamic

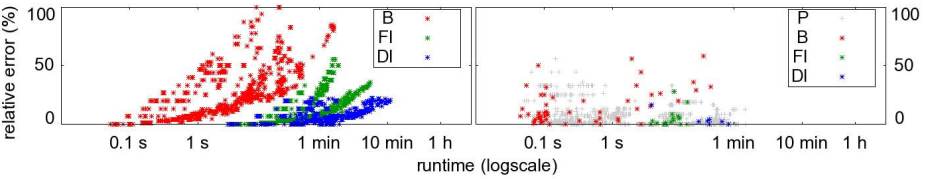

Fig. 2. Illustration of the impact of the introduced intensification strategies

improvement phase in the *Algorithm-DI* setting — the SLS heuristics efficiently prevents the algorithm from being trapped in local minima by modifying the emerging partial solution σ in a way as to increase the overall slackness of the routes and thus increase the chance of further successful allocations.

6.5 Search Diversification and Intensification Strategies Analysis

The highlighted results achieved by the parallel algorithm are based on employing the *search diversification strategy* based on traversing the diversified configuration space $\Omega' \times \Delta'$ generated using two specific ordering diversification operators and by using alternative negotiation semantics. The inherent complexity increase is partially offset by introducing a *search intensification strategy* consisting of ordering pruning, improved restarts strategy and terminating algorithm instances that are not likely to yield good solutions.

Table 1 shows the improvements in solution quality achieved by the introduced search diversification techniques. Compared to the *Algorithm-DI* setting the solutions were improved in 81 cases (23% of problem instances). Both the introduced ordering diversification operators contributed to the improvements in similar way. Our preliminary experiments proved, that the best orderings are found within close neighborhoods of the successful analytically sound orderings, while the orderings featuring greater level of randomization were much less successful. The used operators were thus specifically designed to enable traversing these neighborhoods. An interesting future research opportunity was identified in comparing these operators with well known *ordering crossover* and *mutation* operators used by the genetic ordering based algorithms [13].

The effect of the introduced intensification strategies is illustrated by Figure 2. The two charts capture the runtimes and relative errors of the individual algorithm instances processed by the parallel algorithm over a subset of 16 instances with 1000 customers each. The left chart corresponds to traversing the full configuration space $\Omega' \times \Delta'$ while the right chart corresponds to the intensification strategies being fully employed with the *CSP+BP* ordering pruning strategy being used. The results are grouped based on the used algorithm setting with the terminated algorithm instances being denoted as *P*. Note that the runtime is displayed in logarithmical scale. The results thus outline the dramatic effect the intensification strategies have on the overall runtime of the improved algorithm. The most significant improvements are achieved by (i) terminating

instances not potentially yielding good solutions (the P group) and (ii) limiting the number of executed complex algorithm instances by using an ordering pruning strategy (the results missing on the right chart, especially in the most complex *Algorithm-DI* setting group). In overall this attributes to over 6 times reduction in the overall composite runtime without a significant drop in either anytime convergence or overall quality.

7 Conclusion

This paper provides an insight into our ongoing effort aimed at developing efficient algorithms for the VRPTW based on agent negotiation providing a sound alternative to the classical centralized algorithms. Central to this effort is the exploration of alternative negotiations semantics and local planning strategies used within the agent-based solving process.

An abstract algorithm was introduced, based on a fitting agent decomposition of the solved problem. The abstract negotiation based solving process was briefly outlined. The alternative negotiation semantics were discussed based on three general algorithm settings and using three alternative improvement methods based on customer reallocations. Two particular local planning strategies were introduced as well based on the state-of-the-art insertion heuristics. Finally the full parallel algorithm was presented using a search diversification strategy based on traversing the diversified configuration space $\Omega' \times \Delta'$ and a search intensification strategy based on ordering pruning, improved restarts strategy and terminating algorithm instances that are not likely to yield good solutions.

The performance of the algorithm was evaluated using relevant widely used benchmarks providing relevant comparison to the state-of-the-art algorithms missing from previous studies, making this also the first agent-based algorithm to be assessed using the extended Homberger's benchmark. The parallel algorithm was able to equal the contemporary best known solutions achieved by the classical algorithms in 64% of the cases across both benchmarks with an average relative error of 2.4% in terms of the primary optimization criteria, while boasting an excellent parallel anytime characteristics, outperforming even the centralized algorithms in this respect. These results represent a significant improvement over all previously presented agent-based algorithms and suggest that agent-based solving techniques are relevant with respect to efficiently solving the VRPTW, supporting also the relevance of future research in this area.

In that respect, we argue that of particular relevance is the research of the possible semantics of the underlying negotiation process. For example the already promising results could be further improved by introducing a more complex negotiation semantics complementing the simple task reallocations, providing means to modify even the tightly constrained partial solutions and thus effectively escape local minima. Another opportunity was identified in assessing the suitability of known ordering crossover and mutation operators.

Acknowledgements. This effort was supported by the Grant Agency of the CTU in Prague, grant No. SGS12/188/OHK3/3T/13, the Ministry of Education, Youth and Sports of the Czech Republic, grant No. LD12044 and TUD COST Action TU1102 — Autonomic Road Transport Systems.

References

1. Bachem, A., Hochstattler, W., Malich, M.: The simulated trading heuristic for solving vehicle routing problems. Discrete Applied Mathematics 65(1-3), 47–72 (1996); First International Colloquium on Graphs and Optimization
2. Bräysy, O., Gendreau, M.: Vehicle routing problem with time windows, part I route construction and local search algorithms. Transportation Science 39(1), 104–118 (2005)
3. Bräysy, O., Gendreau, M.: Vehicle routing problem with time windows, part II metaheuristics. Transportation Science 39(1), 119–139 (2005)
4. Dan, Z., Cai, L., Zheng, L.: Improved multi-agent system for the vehicle routing problem with time windows. Tsinghua Science Technology 14(3), 407–412 (2009)
5. Fischer, K., Müller, J.P., Pischel, M.: Cooperative transportation scheduling: an application domain for dai. Journal of Applied Artificial Intelligence 10, 1–33 (1995)
6. Homberger, J., Gehring, H.: A two-phase hybrid metaheuristic for the vehicle routing problem with time windows. European Journal of Operational Research 162(1), 220–238 (2005)
7. Kalina, P., Vokřínek, J.: Algorithm for vehicle routing problem with time windows based on agent negotiation. In: Proceedings of the 7th Workshop on Agents in Traffic and Transportation, AAMAS (2012)
8. Kalina, P., Vokřínek, J.: Improved agent based algorithm for vehicle routing problem with time windows using efficient search diversification and pruning strategy. In: Proceedings of the 3rd Workshop on Artificial Intelligence and Logistice (AILog) of the 2012 European Conference on Artificial Intelligence (ECAI), pp. 13–18 (2012)
9. Leong, H.W., Liu, M.: A multi-agent algorithm for vehicle routing problem with time window, pp. 106–111. ACM (2006)
10. Lim, A., Zhang, X.: A two-stage heuristic with ejection pools and generalized ejection chains for the vehicle routing problem with time windows. INFORMS Journal on Computing 19(3), 443–457 (2007)
11. Lu, Q., Dessouky, M.M.: A new insertion-based construction heuristic for solving the pickup and delivery problem with hard time windows. European Journal of Operational Research 175, 672–687 (2005)
12. Nagata, Y., Bräysy, O.: A powerful route minimization heuristic for the vehicle routing problem with time windows. Operations Research Letters 37(5), 333–338 (2009)
13. Poon, P., Carter, J.: Genetic algorithm crossover operators for ordering applications. Computers and Operations Research 22(1), 135–147 (1995)
14. Ropke, S.: Heuristic and exact algorithms for vehicle routing problems, PHD Thesis, Technical University of Denmark (2005)
15. Solomon, M.M.: Algorithms for the vehicle routing and scheduling problems with time window constraints. Operations Research 35, 254–265 (1987)