Solving a Vehicle Routing Problem with Ant Colony Optimisation and Stochastic Ranking

Alexander Hämmerle and Martin Ankerl

Profactor GmbH
Im Stadtgut A2, 4407 Steyr-Gleink, Austria
{alexander.haemmerle,martin.ankerl}@profactor.at

Abstract. In our contribution we are concerned with a real-world vehicle routing problem (VRP), showing characteristics of VRP with time windows, multiple depots and site dependencies. An analysis of transport request data reveals that the problem is over-constrained with respect to time constraints, i.e. maximum route durations and time windows for delivery at customer sites. Our results show that ant colony optimisation combined with stochastic ranking provides appropriate means to deal with the over-constrained problem. An essential point in our investigations was the development of problem-specific heuristics, guiding ants in the construction of solutions. Computational results show that the combination of a refined distance heuristic, taking into account the distances between customer sites when performing pickup operations at depots, and a look-ahead heuristic, estimating the violation of maximum route durations and delivery time windows when performing pickup operations, provides the best results for the VRP under consideration.

Keywords: logistics, heuristics, routing, ant colony optimisation, constraint satisfaction.

1 Introduction

In their daily business, forwarding companies are confronted with the necessity to minimise transport cost. Capacity utilisation of vehicles has to be maximised, while total travel times have to be minimised. The real-world problems can be abstracted into vehicle routing problems (VRPs), providing the models to study the efficiency of optimisation algorithms. In its basic variant the VRP is concerned with assigning transport requests to a homogeneous fleet of vehicles starting from a single depot and to construct optimal routes for these vehicles. The introduction of constraints gives rise to a whole family of VRP variants, considering vehicle capacities, time windows for deliveries, multiple depots, site dependencies etc. In this paper we are concerned with a real-world VRP, showing characteristics of three general types of vehicle routing problems as classified by [8]: vehicle routing problem with time windows (VRPTW), multi-depot vehicle routing problem (MDVRP) and side dependent vehicle routing problem (SDVRP). In the VRPTW, time windows are assigned to customer sites, constraining the possible delivery dates. In the MDVRP, vehicles start their routes

R. Moreno-Díaz et al. (Eds.): EUROCAST 2013, Part I, LNCS 8111, pp. 259–266, 2013.

from multiple depots, and SDVRP imposes constraints on the accessibility of customer sites - a customer site requires specific vehicle types and excludes others. All of these VRP variants consider vehicle capacities. The vehicle routing problem with multiple depots and time windows MDVRPTW has been tackled by [2,9,4]. To the best of our knowledge there is no existing research paper on MDVRPTW with site dependencies.

In order to solve real-world VRPs, a variety of metaheuristic approaches have been proposed, amongst them ant colony optimisation (ACO). In a number of research publications ACO showed good performance when applied to large-scale and real-world instances. [3] used a multi ants colony system to solve time dependent VRP and tested their algorithm on a real-world case. [10] enhanced a savings based ant system by decomposing the VRP into smaller sub-problems. [11] investigated into the application of ACO to real-world VRP instances exhibiting characteristics of VRP with time windows, time dependent VRP, dynamic VRP and VRP with pickup and delivery. They conclude that "... ACO has been shown to be one of the most successful metaheuristics for the VRP and its application to real-world problems demonstrates that it has now become a fundamental tool in applied operations research."

Real-world VRP are characterised by a considerable amount of constraints that have to be observed during vehicle routing optimisation. Therefore, ACO has been integrated with several constraint handling mechanisms, amongst them constraint programming and penalty-based techniques. Of particular interest is the approach presented by [7] and [5] where ACO has been combined with stochastic ranking, a constraint handling mechanism originally proposed by [12]. In stochastic ranking a stochastic bubble sort algorithm ranks solutions according to their objective value or their amount of constraint violation, respectively. Top ranked solutions, including feasible and infeasible solutions, pass information to the next iteration (via pheromones in the case of ACO). [7] used ACO in combination with stochastic ranking to solve single machine job scheduling problems. Despite its low computational complexity stochastic ranking showed good performance when applied to weakly or moderately constrained problems.

In this paper we use an ACO algorithm integrated with stochastic ranking in order to solve a real-world VRP. To the best of our knowledge the application of ACO combined with stochastic ranking to VRP has not been reported before. The remainder of this paper is structured as follows. In section 2 we describe the real-world VRP under consideration. Our problem solving approach is outlined in section 3. In section 4 we discuss computational results, pointing out the importance of strong, problem-specific heuristics guiding the ACO algorithm in the construction of solutions. Our conclusions are presented in section 5.

2 Problem Description

Throughout the paper we discuss a particular real-world VRP instance, where 398 customers have to be supplied from two depots. The optimisation objective is the minimisation of travel times. We investigate a static planning problem

with single day planning horizons, i.e. at the time of optimisation all transport requests to be scheduled for the next day are known in advance. Each customer may request several shipments per day. In the considered problem instance we are dealing with 789 transport requests. A fleet of 265 vehicles is available to supply the customers. The fleet is inhomogeneous: four types of vehicle are available, with many different vehicle capacities within each type. Each vehicle is assigned to a depot, at the beginning and at the end of a route the vehicle has to be located at its depot.

2.1 Data Model

In the following we describe the core entities making up the data model for our VRP instance. A *vehicle* is characterised through its type, capacity (measured in pallet spaces), a list of up to 3 shifts and the distance matrix relevant for the vehicle. Examples for vehicle types are semitrailer or road train. A shift is specified through an origin (the site where the vehicle starts its shift), a destination (the site where the vehicle has to be at the end of the shift), a shift start time and a shift end time. In the considered problem instance the origin as well as the destination of a shift is the depot where the vehicle is assigned to. Each entry in a distance matrix specifies the travel time between a pair of sites. We are using two different distance matrices, reflecting vehicle type specific travel times.

A *customer site* or *depot* is characterised through its geographic coordinates, up to 2 time windows, a service time and a list of supported vehicle types. Time windows specify the opening hours of a site, whereas the service time defines the average time required for a loading/unloading operation. The list of supported vehicle types specifies which vehicle types are able to access the site for loading/unloading operations. Finally a *transport request* is characterised through an origin, a destination and the amount of pallet spaces the load requires.

2.2 Constraints

Four types of constraints narrow the space of feasible solutions: (1) Loading/ unloading operations should not be performed outside of specified time windows, (2) vehicle shifts limit the maximum duration of routes, (3) the supported vehicle types of a customer site / depot impose site dependency constraints on vehicles and (4) the total load of a vehicle must not exceed its capacity.

An analysis of transport request data reveals that the problem is over-constrained. We looked at the duration of transport for individual requests, i.e. the travel time from depot to customer site and back to the depot, plus service times for loading at depot and unloading at customer site. By comparing transport times with duration of vehicle shifts we noticed that in the worst case 146 transport requests cannot be fulfilled without violation of shift constraints. Worst case means that only vehicles using the distance matrix with longer travel times are used to fulfil requests with long transport times. In the best case 108 transport requests violate shift constraints, employing as much as possible vehicles using the distance matrix with shorter travel times.

3 Solving the Vehicle Routing Problem

3.1 Solution Construction

ACO algorithms are constructive solution methods, where a number of artificial ants move through a construction graph, that represents a particular problem. In our application a vertex in the construction graph corresponds to a logical customer site or to a logical depot. A logical depot represents the pickup location of a specific transport request, the origin of a vehicle's shift or the destination of a vehicle's shift. Examples: two requests with pickup at the same physical depot result in two different logical depots. Two vehicles with a single shift and shift origins / destinations at the same physical depot give rise to four different logical depots, two for shift origins and two for shift destinations. Due to the fact that customers may request multiple transports we had to introduce logical customer sites, one for each request. Example: a physical customer site requests three transports, resulting in three logical customer sites with the same geographic coordinates. With the above definition of a vertex in the construction graph at hand, a solution generated by an ant corresponds to a path through the construction graph connecting all vertices.

Ants generate solutions iteratively by following artificial pheromone trails through the construction graph. At each iteration, ants re-enforce "good solutions" (according to an objective function) with additional pheromone deposits, thus increasing the probability for segments of good solutions to be re-used in subsequent iterations. However, it is not just pheromone trails that guide ants during the construction of solutions. A problem-specific heuristic function also influences the choices made by ants. Such a heuristic is especially important in early iterations, when sufficient pheromone information is not yet available. The importance of pheromones and heuristics is expressed in equation 1, describing an ant's probability to move from vertex i to vertex j in the construction graph.

$$p_{i,j} = \frac{\tau_{i,j}^{\alpha} \cdot \eta_{i,j}^{\beta}}{\sum_{k \in N_i} \tau_{i,k}^{\alpha} \cdot \eta_{i,k}^{\beta}} \tag{1}$$

In equation 1, $\tau_{i,j}$ denotes the pheromone value on the edge from vertex i to vertex j, and $\eta_{i,j}$ is the heuristic value associated with the move from vertex i to vertex j. The parameters α and β are weights for the influence of pheromones and heuristic values. The neighbourhood of vertex i, i.e. all vertices reachable with a single move, is denoted by N_i.

Our ACO implementation is based on the MAX-MIN ant system originally proposed by [13]. To avoid pre-mature convergence of search the MAX-MIN algorithm introduces upper and lower boundaries on pheromone values, thus limiting differences in pheromone trail intensities. We modified the MAX-MIN algorithm with adaptive values for α and β, resulting in faster convergence and better solution qualities [1]. Solution construction in ACO algorithms conceptually supports concurrent computation, with ants determining solutions in parallel. We have exploited this feature with the implementation of a concurrency mechanism, based on the Reduction design pattern described in [6].

3.2 Heuristics

To study the influence of different heuristics on solution quality and convergence we have implemented the following standard heuristics: tightest time window, earliest due date, earliest start time and shortest distance (travel time). Applying these heuristics or combinations thereof to the problem specified in section 2 we noticed that the results are not very satisfactory, as the standard heuristics are not able to ensure that transport requests with deliveries to customer sites in close proximity to each other are grouped together in as few routes as possible. This is achieved with a *refined distance heuristic*, considering the travel time between customer sites when evaluating pickups at depots.

Time windows at customer sites and vehicle shifts constrain the space of feasible solutions. An efficient heuristic should consider these time constraints when guiding ants in the construction of solutions. Pickup operations are of particular interest, as they increase the vehicle's workload. We have implemented a *look-ahead heuristic* for the evaluation of pickup operations, estimating the time of arrival at the shift destination and calculating the violation of time constraints. Due to the over-constrained nature of the problem we had to introduce a tolerance with respect to delays. A move whose look-ahead return value exceeds the tolerance is discarded as infeasible.

3.3 Stochastic Ranking of Solutions

In stochastic ranking, the evaluation of a solution s generated by an ant takes into account the objective function $O(s)$ and the amount of constraint violation of s. After each iteration, the solutions constructed by the ants in the iteration are added to a fixed size ranking list. A stochastic bubble sort algorithm sorts the ranking list, where a probability parameter P_f denotes the probability for a pair of solutions to be compared according to their objective values, cf. [7,12]. Hence with probability $1-P_f$ the pair of solutions is compared according to their respective amount of constraint violation. After the ranking procedure the list is pruned, i.e. all solutions with a rank higher than the size of the list are removed. The remaining solutions in the list will deposit pheromones, influencing ants' decisions in future iterations. The amount of a pheromone deposit is given by equation 2, with L denoting the size of the ranking list.

$$\Delta\tau_{i,j} = 1/\left(L \cdot O(s)\right)\ if(i,j) \in s;\ 0\, otherwise \tag{2}$$

4 Discussion of Computational Results

Two questions were guiding our computational experiments. (1) What is the influence of different heuristics on algorithm performance, and (2) what is the relationship between solution quality and computation time. All experiments used the problem instance described in section 2 and they were performed on an Intel Core Duo CPU with 1.99 GHz running under Windows XP. The parameter P_f was set to 0.4, as this value yielded the best results.

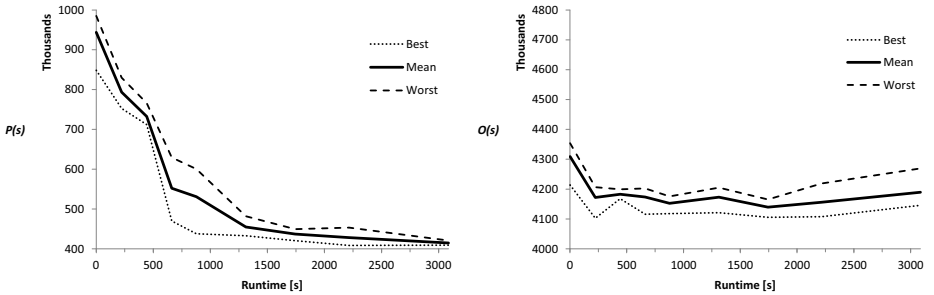

Fig. 1. Runtime performance for solutions with globally best $P(s)$ and $O(s)$ with look-ahead and refined distance heuristic. Look-ahead tolerance set to 5400 s.

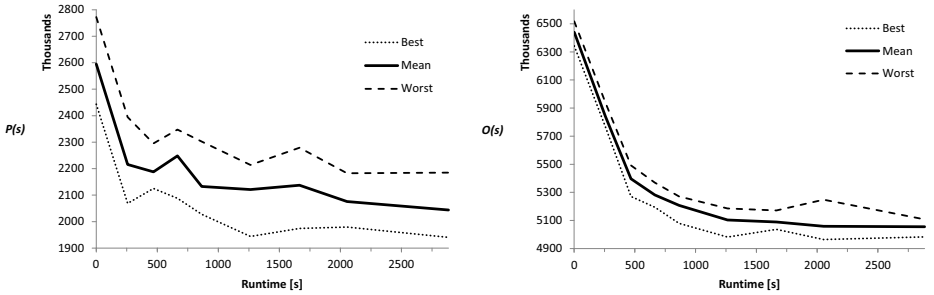

Fig. 2. Runtime performance for solutions with globally best $P(s)$ and $O(s)$ with look-ahead and simple distance heuristic. Look-ahead tolerance set to 5400 s.

To examine the runtime performance all experiments were carried out with the following numbers of iterations: 1, 50, 100, 150, 200, 300, 400, 500 and 700. Solutions found in the first iteration are especially interesting, as the quality of these solutions solely depends on the chosen heuristics. For each number of iterations we performed 5 independent runs, calculating mean values as well as worst/best values for globally best $O(s)$ and $P(s)$, where $P(s)$ denotes the amount of constraint violation for solution s. Figure 1 reports the results for the best heuristics we have found, namely a combination of look-ahead and refined distance heuristic. The look-ahead tolerance was set to 5400 s, yielding the best results for our problem instance. We notice a strong reduction in constraint violation with increasing runtime, whereas the objective function does not show such a reduction.

Figure 2 shows a different picture. It depicts results obtained with a combination of look-ahead (tolerance set to 5400 s) and a simple distance heuristic with $\eta_{i,j} = 1/t_{i,j}$, where $t_{i,j}$ denotes the time required for travelling from vertex i to vertex j. The reduction of $O(s)$ in figure 2 is significant, and we note a strong reduction in $P(s)$. However, looking at the initial values for $O(s)$ and $P(s)$, found in the first iteration, we notice that the simple distance heuristic

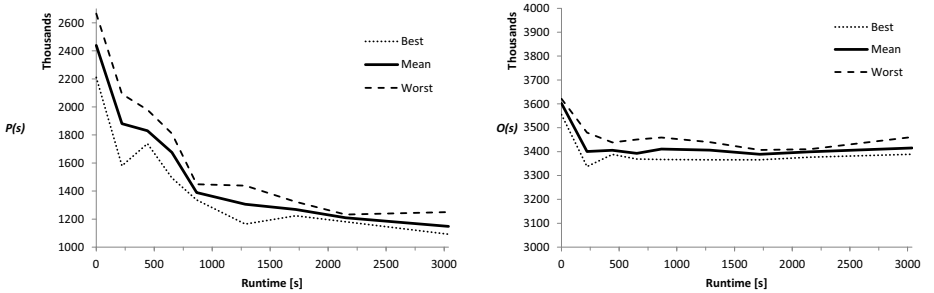

Fig. 3. Runtime performance for solutions with globally best $P(s)$ and $O(s)$ with look-ahead and refined distance heuristic. Look-ahead tolerance set to 14400 s.

produces very bad initial solutions compared to the refined distance heuristic. Starting from these low quality solutions the optimisation algorithm is able to reduce $O(s)$ and $P(s)$ simultaneously. With good initial cost values found by the refined distance heuristic, the algorithm, with $P_f = 0.4$ biased towards reduction of constraint violation, significantly reduces the amount of constraint violation while maintaining the quality level of objective values from early iterations.

This finding is also reflected in figure 3, where we report results obtained with a relaxed look-ahead (tolerance set to 14400 s) and a refined distance heuristic. Again the refined distance heuristic is able to produce good initial values for $O(s)$, being further reduced in the first iterations, but steadying throughout the remainder of the optimisation. Due to the relaxed look-ahead the values for $O(s)$ are lower than in figure 1: the algorithm uses the larger space of feasible solutions, i.e. solutions not violating the look-ahead tolerance, to produce solutions with lower values for $O(s)$. However, this comes at the expense of increased constraint violation.

5 Conclusions

When choosing ant colony optimisation to solve real-world VRP, tailored heuristics are an important aspect in order to achieve satisfactory results. Our investigations showed that the application of problem-specific heuristics significantly improved the quality of solutions, compared to the usage of simple heuristics like shortest distance. In our experiments with a real-world, over-constrained VRP we focussed on the evaluation of globally best solutions with respect to objective value or constraint violation, respectively. However, the algorithmic approach presented in this paper provides more solutions than just globally best ones. The ranking list allows a user to choose from a multitude of solutions, balancing the trade-off between low objective value and low constraint violation in a different way. The probability parameter P_f provides the simple means to adjust the bias of the optimisation algorithm towards reduction of objective value or constraint violation. Based on our results we consider the combination

of ant colony optimisation with stochastic ranking to be the basis of a useful tool, flexibly supporting logistics practitioners in the solving of vehicle routing problems, even if they are over-constrained.

Acknowledgement. This work was co-funded by the Austrian Research Promotion Agency (FFG) under the project number 200760.

References

1. Ankerl, M., Hämmerle, A.: Applying Ant Colony Optimisation to Dynamic Pickup and Delivery. In: Moreno-Díaz, R., Pichler, F., Quesada-Arencibia, A. (eds.) EUROCAST 2009. LNCS, vol. 5717, pp. 721–728. Springer, Heidelberg (2009)
2. Cordeau, J.F., Laporte, G., Mercier, A.: A Unified Tabu Search Heuristic for Vehicle Routing Problems with Time Windows. Journal of the Operational Research Society 52, 928–936 (2001)
3. Donati, A.V., Montemanni, R., Casagrande, N., Rizzoli, A.E., Gambardella, L.M.: Time Dependent Vehicle Routing Problem with a Multi Ant Colony System. European Journal of Operational Research 185(3), 1174–1191 (2008)
4. Dondo, R., Cerdá, J.: A cluster-based optimization approach for the multi-depot heterogeneous fleet vehicle routing problem with time windows. European Journal of Operational Research 176(3), 1478–1507 (2007)
5. Held, M.: Analysis and Improvement of Constraint Handling in Ant Colony Algorithms. Thesis, Clayton School of Information Technology, Monash University (2005)
6. Intel^{TM}: Threading Building Blocks Design Patterns V1.0, Text file (2010), `http://threadingbuildingblocks.org/` (last accessed April 27, 2011)
7. Meyer, B.: Constraint Handling and Stochastic Ranking in ACO. Evolutionary Computation 3, 2683–2690 (2005)
8. Pisinger, D., Ropke, S.: A general heuristic for vehicle routing problems. Computers & Operations Research 34(8), 2403–2435 (2007)
9. Polacek, M., Hartl, R.F., Doerner, K.: A Variable Neighborhood Search for the Multi Depot Vehicle Routing Problem with Time Windows. Journal of Heuristics 10, 613–627 (2004)
10. Reimann, M., Doerner, K., Hartl, R.F.: D-Ants: Savings Based Ants divide and conquer the vehicle routing problem. Computers & Operations Research 31, 563–591 (2004)
11. Rizzoli, A.E., Montemanni, R., Lucibello, E., Gambardella, L.M.: Ant colony optimization for real-world vehicle routing problems. Swarm Intelligence 1, 135–151 (2007)
12. Runarsson, T.P., Yao, X.: Stochastic Ranking for Constrained Evolutionary Optimization. IEEE Transactions on Evolutionary Computation 4(3), 284–294 (2000)
13. Stützle, T., Hoos, H.H.: MAX-MIN Ant System. Future Generation Computer Systems 16(8), 889–914 (2000)