SavingsAnts for the Vehicle Routing Problem

Karl Doerner, Manfred Gronalt, Richard F. Hartl, Marc Reimann,
Christine Strauss, and Michael Stummer

Institute of Management Science, University of Vienna, Brünnerstrasse 72,
A-1210 Vienna, Austria
{karl.doerner, manfred.gronalt, richard.hartl, marc.reimann,
christine.strauss}@univie.ac.at, a9705488@unet.univie.ac.at
http://www.bwl.univie.ac.at/bwl/prod/index.html

Abstract. In this paper we propose a hybrid approach for solving vehicle routing problems. The main idea is to combine an Ant System (AS) with a problem specific constructive heuristic, namely the well known Savings algorithm. This differs from previous approaches, where the subordinate heuristic was the Nearest Neighbor algorithm initially proposed for the TSP. We compare our approach with some other classic, powerful meta-heuristics and show that our results are competitive.

1 Introduction

The European situation in freight transportation reflects the need for improved efficiency, as the traffic volume increases much faster than the street network grows. Thus, given the current efficiency, this will eventually lead to a breakdown of the system. However, with rapidly increasing computational power intelligent optimization methods can be developed and used to increase the efficiency in freight transportation and circumvent the above mentioned problem.

The VRP involves the design of a set of minimum cost delivery routes, originating and terminating at a depot, which services a set of customers. Each customer must be supplied exactly once by one vehicle route. The total demand of any route must not exceed the vehicle capacity. The total length of any route must not exceed a pre-specified bound. This problem is known to be NP-hard (cf. [1]), such that exact methods like Dynamic Programming or Branch & Bound work only for relatively small problems in reasonable time. Thus, a large number of approximation methods have been proposed. Most of the recent approaches are based on meta-heuristics like Tabu Search, Simulated Annealing and Ant Systems.

The Ant System approach, belonging to a class of methods called Ant Colony Optimization, is based on the behavior of real ants searching for food. Real ants communicate with each other using an aromatic essence called pheromone, which they leave on the paths they traverse. If ants sense pheromone in their vicinity, they are likely to follow that pheromone, thus reinforcing this path. The pheromone trails reflect the 'memory' of the ant population. The quantity of the pheromone deposited on paths depends on both, the length of the paths as well as the quality of the food source found.

S. Cagnoni et al. (Eds.): EvoWorkshops 2002, LNCS 2279, pp. 11–20, 2002.

The observation of this behavior has led to the development of Ant Systems by Colorni et al. (see e.g. [2]) in the early nineties. In artificial terms the optimization method uses the trail following behavior described above in the following way. Ants construct solutions by making a number of decisions probabilistically. In the beginning there is no collective memory, and the ants can only follow some local information. As some ants have constructed solutions, pheromone information is built. In particular, the quantity of pheromone deposited by the artificial ants depends on the solution quality found by the ants. This pheromone information guides other ants in their decision making, i.e. paths with high pheromone concentration will attract more ants than paths with low pheromone concentration. On the other hand, the pheromone deposited is not permanent, but rather evaporates over time. Thus, over time, paths that are not used will become less and less attractive, while those used frequently will attract ever more ants.

This approach has been applied to a number of combinatorial optimization problems, such as the Graph Coloring Problem [3], the Quadratic Assignment Problem (e.g. [4]), the Travelling Salesman Problem (e.g. [5], [6]), the Vehicle Routing Problem ([7], [8]) and the Vehicle Routing Problem with Time Windows ([9]). Recently, a covergence proof for a generalized Ant System has been developed by Gutjahr ([10]).

In the previous approaches for the VRP ([7], [8]) the construction of solutions was based on a sequential tour building approach, which utilized a parametrized savings criterion. The main idea of our new approach is to transfer the simultaneous tour construction mechanism proposed in [11] into a rank based Ant System. Our computational findings show that a considerable improvement is achieved through this new approach.

The remainder of this paper is organized as follows. In the next section we briefly describe the Savings algorithm before we propose our new approach, which we will refer to as SavingsAnts. After that we will report on computational results with our SavingsAnts. In Section 4 we conclude with a discussion of our findings.

2 Savings and Ant System Algorithms for VRPs

In this section we describe the Savings algorithm and the Ant Systems algorithm. The basic structure of our Ant System algorithm is identical to the one proposed in [8]. Thus, we will focus on our improvements to the original algorithm.

2.1 The Savings Algorithm

The Savings algorithm, proposed in [11], is the basis of most commercial software tools for solving VRPs in industrial applications. It is initalized with the assignment of each customer to a separate tour.
After that for each pair of customers i and j the following savings measure is calculated:

$$s(i,j) = d(i,0) + d(0,j) - d(i,j), \tag{1}$$

where $d(i, j)$ denotes the distance between locations i and j and the index 0 denotes the depot. Thus, the values $s(i, j)$ contain the savings of combining two customers i and j on one tour as opposed to serving them on two different tours.

In the iterative phase, customers or partial tours are combined according to these savings, starting with the largest savings, until no more combinations are feasible. A combination is infeasible if it violates either the capacity or the tourlength constraints.

The result of this algorithm is a (sub-)optimal set of tours through all customers.

2.2 The Ant System Algorithm for the VRP

Bullnheimer et al. ([7], [8]) have first applied the Ant System to the VRP. Their approach centers on the similarity of VRPs with TSPs, namely the fact that for a given clustering of customers the problem reduces to several TSPs. Thus, their Ant System is strongly influenced by the Ant System algorithms applied to the TSP. On the contrary, our approach, to our best knowledge, is the first combination of a heuristic algorithm for the VRP with an Ant System.

The Ant System algorithm mainly consists of the iteration of three steps:

- Generation of solutions by ants according to private and pheromone information
- Application of a local search to the ants' solutions
- Update of the pheromone information

In addition to that our approach features a fourth step, namely:

- Augmentation of the Attractiveness list, which stores the desirability of all feasible combinations.

The implementation of these four steps is described below.

Generation of Solutions. As stated above, the solution generation technique we implemented is the main contribution of this paper. So far, in Ant Systems solutions for the VRP have been built using a Nearest Neighbor heuristic (see e.g. [7], [8]). As opposed to that we use the Savings algorithm described above to generate solutions. To that end we need to modify the deterministic version of the algorithm. This modification is done in the following way.

Initially, we generate a sorted list of attractiveness values ξ_{ij} in decreasing order. These attractiveness values feature both the savings values as well as the pheromone information.
Thus the list consists of the following values

$$\xi_{ij} = [s(i, j)]^{\beta}[\tau_{ij}]^{\alpha} \tag{2}$$

where τ_{ij} denotes the pheromone concentration on the arc connecting customers i and j, and α and β bias the relative influence of the pheromone trails and the

savings values, respectively. The pheromone concentration τ_{ij} contains information about how good the combination of two customers i and j was in previous iterations.

In each decision step of an ant, we consider the k best combinations still available, where k is a parameter of the algorithm which we will refer to as 'neighborhood' below.

Let Ω_k denote the set of k neighbors, i.e. the k feasible combinations (i, j) yielding the largest savings, considered in a given decision step, then the decision rule is given by equation 3, where $\mathcal{P}_{ij}$ is the probability of choosing to combine customers i and j on one tour.

$$\mathcal{P}_{ij} = \begin{cases} \frac{\xi_{ij}}{\sum_{(h,l)\in\Omega_k} \xi_{hl}} & \text{if } \xi_{ij} \in \Omega_k \\ \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

The construction process is stopped when no more feasible combinations are possible.

Local Search. Following [8] we apply the 2-opt algorithm (c.f. [12]) to all vehicle routes built by the ants, before we update the pheromone information. The 2-opt algorithm was developed for the traveling salesman problem and iteratively exchanges two edges with 2 new edges until no further improvements are possible.

Pheromone Update. After all ants have constructed their solutions, the pheromone trails are updated on the basis of the solutions found by the ants. According to the rank based scheme proposed in [6] and [8], the pheromone update is as follows

$$\tau_{ij}^{new} = \rho\tau_{ij}^{old} + \sum_{\mu=1}^{\sigma-1} \Delta\tau_{ij}^{\mu} + \sigma\Delta\tau_{ij}^{*} \tag{4}$$

where $0 \leq \rho \leq 1$ is the trail persistance and σ is the number of elitists. Using this scheme two kinds of trails are laid. First, the best solution found during the process is updated as if σ ants had traversed it. The amount of pheromone laid by the elitists is $\Delta\tau_{ij}^{*} = 1/L^{*}$, where L^{*} is the objective value of the best solution found so far. Second, the $\sigma - 1$ best ants of the iteration are allowed to lay pheromone on the arcs they traversed. The quantity laid by these ants depends on their rank μ as well as their solution quality L^{μ}, such that the μ-th best ant lays $\Delta\tau_{ij}^{\mu} = (\sigma - \mu)/L^{\mu}$. Arcs belonging to neither of those solutions just lose pheromone at the rate $(1 - \rho)$, which constitutes the trail evaporation.

Augmentation of the Savings List. After the pheromone information has been updated the attractiveness values ξ_{ij} are augmented with the new pheromone information as in equation 2.

After the augmentation the attractiveness values are again sorted in decreasing order. This mechanism is the second important contribution of our new

approach. In the beginning the attractiveness values are sorted according to the savings values, as the pheromone is equal on all arcs. As learning occurs, and some arcs are reinforced through the update of the pheromone information, the attractiveness values ξ_{ij} change, as they become more and more biased by the pheromone information. Thus, values that were initially high but turned out not to be in good solutions will decrease, while combinations with initially low values that appeared in good solutions will become more attractive. As the attractiveness values are re-sorted after each iteration, this leads to dynamic effects. In particular, 'good' arcs are reinforced twice. First, they receive more pheromone than others, and second as their attractiveness increases they are considered earlier in the constructive process.

3 Numerical Analysis

In this section we will present numerical results for our new approach and compare them with results from previous Ant System approaches as well as different meta-heuristics.

3.1 The Benchmark Problem Instances

The numerical analysis was performed on a set of benchmark problems described in [13]. The set of benchmark problems consists of 14 instances containing between 50 and 199 customers and a depot. The first ten instances were generated with the customers being randomly distributed in the plane, while instances 11-14 feature clusters of customer locations. All instances are capacity constrained. In addition to that, the instances 6-10 and 13-14 are also restricted with respect to tour length. In these instances, all customers have identical service times δ. Apart from the additional time constraints, instances 1-5 and 6-10 are identical. The same is true for instances 11-12 and 13-14. Table 1 contains the data for the 14 problem instances [1].

3.2 Parameter Settings

In order to keep the results comparable and to isolate the effects of our new approach, we chose to basically use the same parameter values as proposed in [8]. Thus, we used n artificial ants, $\alpha = \beta = 5$ and $\sigma = 6$ elitist ants. We only found that for our approach an evaporation rate $\rho = 0.95$ is preferable to $\rho = 0.75$ as proposed in earlier works.

Apart from that we varied the number of iterations in order to be able to estimate the performance of our algorithm for different run times. More specifically we will provide results for $\lfloor n/2 \rfloor$, n and $2 \cdot n$ iterations together with the corresponding computation times.[2]

[1] The instances can be found at http://www.ms.ic.ac.uk/jeb/orlib/vrpinfo.html

[2] The algorithms were implemented in Borland C and run on a Pentium 3 with 900MHz.

Table 1. Characteristics of the benchmark problem instances

Instance	n	Q	L	δ	best publ.
Random Problems					
C1	50	160	∞	0	524.61 [14]
C2	75	140	∞	0	835.26 [14]
C3	100	200	∞	0	826.14 [14]
C4	150	200	∞	0	1028.42 [14]
C5	199	200	∞	0	1291.45 [15]
C6	50	160	200	10	555.43 [14]
C7	75	140	160	10	909.68 [14]
C8	100	200	230	10	865.94 [14]
C9	150	200	200	10	1162.55 [14]
C10	199	200	200	10	1395.85 [15]
Clustered Problems					
C11	120	200	∞	0	1042.11 [14]
C12	100	200	∞	0	819.56 [14]
C13	120	200	720	50	1541.14 [14]
C14	100	200	1040	90	866.37 [14]

n ... number of customers
Q ... vehicle capacity
L ... maximum tour length
δ ... service time
best publ. ... best published solution

Apart from that, given our dynamic savings list, we performed runs with different sizes of the neighborhood, i.e. with different numbers of alternatives in each decision step of an ant. We provide results for neighborhood sizes of $k = \lfloor n/5 \rfloor$, $k = \lfloor n/4 \rfloor$, $k = \lfloor n/2 \rfloor$ and $k = n$.

3.3 Experiments with the Size of the Neighborhood

In this section let us first provide some results we obtained using the different neighborhood sizes. In Table 2 we show for all neighborhood sizes the deviations (RPD) of both our best ($best$) and average ($avg.$) solutions (over 10 runs of all 14 problem instances) from the best known solutions after $\lfloor n/2 \rfloor$, n and $2 \cdot n$ iterations. In addition to that we report the corresponding computation times in seconds.

Table 2 mainly shows two different effects. First, we see that increasing the size of the neighborhood generally increases the computation times. This is clear

Table 2. Effects of neighborhood sizes and numbers of iterations on solution quality and computation times.

neighborhood size k	number of iterations: $\lfloor n/2 \rfloor$ RPD (in %) best	avg.	CPU sec.	n RPD (in %) best	avg.	CPU sec.	$2 \cdot n$ RPD (in %) best	avg.	CPU sec.
$\lfloor n/5 \rfloor$	0.93	1.63	134.37	0.8	1.44	268.73	0.8	1.42	537.46
$\lfloor n/4 \rfloor$	1.12	1.77	134.31	0.7	1.41	268.62	0.7	1.41	537.24
$\lfloor n/2 \rfloor$	2.09	2.76	183.26	0.83	1.42	366.51	0.83	1.41	733.03
n	3.21	4.31	260.38	1.04	1.63	520.75	0.96	1.54	1041.50

as an increased size of the neighborhood increases the computational effort for decision making.

Second, the solution quality generally deteriorates with increased neighborhood size. This effect is partially reversed between $\lfloor n/5 \rfloor$ and $\lfloor n/4 \rfloor$. There we see that the best solutions we found were obtained using $k = \lfloor n/4 \rfloor$. However, we also see that for small computation times $k = \lfloor n/5 \rfloor$ is preferable. This becomes clear from the following observation. In the first iterations pheromone has very little influence on decision making, thus the ants explore the search space. As pheromone is built, the attractiveness list changes adaptively, favoring combinations that were successful in previous iterations, i.e. led to good solutions. In this phase exploration is slowly replaced with exploitation. After a number of iterations the list will be sorted in such a way, that the combinations associated with the $n - m$ largest attractiveness values can all be chosen, i.e. these combinations reflect the best found solution. At that point the algorithm reaches some kind of 'natural' convergence and the ants (almost) only exploit the pheromone information. Thus, the algorithm gradually turns from exploration of the search space in early iterations, to exploitation of the neighborhood of good solutions in later iterations.

The exploration phase in the first iterations of the algorithm depends crucially on the neighborhood size. A small neighborhood, while leading to good solutions quickly, will allow only insufficient exploration of the search space and the algorithm converges too early to a sub-optimal level.

This effect is reduced with increased size k of the neighborhood. However, the other extreme, a neighborhood size of n leads to better exploration but inferior solutions in the first iterations. While it may be able to find very good solutions in the long run its computation times are prohibitive to make it an interesting alternative.

Thus, the conclusion that can be drawn from this analysis is clear cut. The best size of the neighborhood is $k = \lfloor n/4 \rfloor$.

3.4 Comparison between Our New Approach and Existing Meta-heuristics

Now that we know the 'appropriate' size of the neighborhood $k = \lfloor n/4 \rfloor$, we will compare the best results obtained with our new approach (denoted by SavingsAnts) after $2 \cdot n$ iterations using this neighborhood size with the results of the previous Ant System algorithm for the VRP as well as with Tabu Search and Simulated Annealing algorithms. These algorithms are: the ant system algorithm (AS) from [8], the parallel tabu search algorithm (PTS) from [16], the TABUROUTE algorithm (TS) from [17] and the simulated annealing algorithm (SA) from [18]. The results of this comparison are presented in Table 3.[3] First, we present the objective values obtained by the different algorithms for the 14 problems. The last two rows show for all algorithms the relative percentage deviation (RPD) over the best known solution. More specifically, the second to last row gives the average RPD for the random problems (C1-C10), while the last row shows the average RPD for the clustered problems (C11-C14).

Table 3. Comparison of five meta-heuristics

Instance	PTS	TS	SA	AS	SavingsAnts
C1	524.61	524.61	528	524.61	524.61
C2	835.32	835.77	838.62	844.31	838.6
C3	827.53	829.45	829.18	832.32	838.38
C4	1044.35	1036.16	1058	1061.55	1040.86
C5	1334.55	1322.65	1376	1343.46	1307.78
C6	555.43	555.43	555.43	560.24	555.43
C7	909.68	913.23	909.68	916.21	913.01
C8	866.75	865.94	866.75	866.74	870.1
C9	1164.12	1177.76	1164.12	1195.99	1173.42
C10	1420.84	1418.51	1417.85	1451.64	1438.72
C11	1042.11	1073.47	1176	1065.21	1043.89
C12	819.56	819.56	826	819.56	819.56
C13	1550.17	1573.81	1545.98	1559.92	1548.14
C14	866.37	866.37	890	867.07	866.37
RPD (avg.)					
C1-C10	0.71	0.70	1.26	1.76	0.92
C11-C14	0.15	1.28	4.17	0.88	0.16

[3] In this table we do not show computation times. A comparison of computation times is not reasonable as all the approaches were tested on different machines. So, while our algorithm consumes by far the smallest computation times, it was also run on the most powerful machine. However, from the execution times reported in Table 2 we are confident, that our algorithm is more than competitive with respect to computational effort.

From Table 3 can be seen that our algorithm shows competitive behavior when compared with the other meta-heuristics. The simulated annealing and previous ant system algorithms are clearly outperformed by our new ant system algorithm. On the other hand, only the parallel tabu search seems to be superior, while the TABUROUTE algorithm shows comparable results. Our SavingsAnts are particularly well suited for the clustered problems as the results for the test instances C11-C14 show. The average percentage deviation of our SavingsAnts' solutions from the best known solutions is only 0.16%.

The reason for the strong performance of the SavingsAnts for clustered problems can be stated as follows. The decision criterion favors the connection of customers that are close to each other and far from the depot. In connection with the simultaneous tour construction this is a good mechanism to identify clusters (c.f. [13]) and to avoid unnecessary connections of customers from two different clusters.

4 Conclusions and Future Research

In this paper we have shown the possible improvements to standard Ant System approaches for VRPs through the use of a problem specific heuristic, namely the Savings algorithm. The computational study performed shows the superior performance of our new approach over the existing Ant System algorithm. In particular, the average results of our approach improves the solution quality of the previous Ant System significantly.

Furthermore, we show that our approach is competitive when compared with other meta-heuristics such as Tabu Search and Simulated Annealing, in particular when applied to clustered problems.

Finally, the average deviation of less than 1.5% over the best known solutions indicates the strength of the proposed algorithm. Moreover, given the performance on the clustered problems this is particularly true for real world problems, which generally exhibit a clustered structure.

Future work will focus on further improvements on the approach. In particular, we will apply our multi-colony approach, as proposed in [19], to the problem. Apart from that we will focus on VRPs with additional features like multiple depots, backhauls and time windows.

Acknowledgments

This work was supported by the Austrian Science Foundation under grant SFB #010 and by the Oesterreichische Nationalbank (OeNB) under grant #8630.

We would also like to thank B.Bullnheimer for drawing our attention to Ant Systems.

Thanks are also due to three anonymous referees for valuable comments on the original version of the papers.

References

1. Garey, M. R. and Johnson, D. S.: Computers and Intractability: A Guide to the Theory of NP Completeness. W. H. Freeman & Co., New York (1979)
2. Colorni, A., Dorigo, M. and Maniezzo, V.: Distributed Optimization by Ant Colonies. In: Varela, F. and Bourgine, P. (Eds.): Proc. Europ. Conf. Artificial Life. Elsevier, Amsterdam (1991)
3. Costa, D. and Hertz, A.: Ants can colour graphs. Journal of the Operational Research Society **48**(3) (1997) 295–305
4. Stützle, T. and Dorigo, M.: ACO Algorithms for the Quadratic Assignment Problem. In: Corne, D., Dorigo, M. and Glover, F. (Eds.): New Ideas in Optimization. Mc Graw-Hill, London (1999)
5. Dorigo, M. and Gambardella, L. M.: Ant Colony System: A cooperative learning approach to the Travelling Salesman Problem. IEEE Transactions on Evolutionary Computation **1**(1) (1997) 53–66
6. Bullnheimer, B., Hartl, R. F. and Strauss, Ch.: A new rank based version of the ant system: a computational study. Central European Journal of Operations Research **7**(1) (1999) 25–38
7. Bullnheimer, B., Hartl, R. F. and Strauss, Ch.: Applying the ant system to the vehicle routing problem. In: Voss, S., Martello, S., Osman, I. H. and Roucairol, C. (Eds.): Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization. Kluwer, Boston (1999)
8. Bullnheimer, B., Hartl, R. F. and Strauss, Ch.: An improved ant system algorithm for the vehicle routing problem. Annals of Operations Research **89** (1999) 319–328
9. Gambardella, L. M., Taillard, E. and Agazzi, G.: MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing Problems with Time Windows. In: Corne, D., Dorigo, M. and Glover, F. (Eds.): New Ideas in Optimization. McGraw-Hill, London (1999)
10. Gutjahr, W. J.: A graph-based Ant System and its convergence. Future Generation Computing Systems. **16** (2000) 873–888
11. Clarke, G. and Wright, J. W.: Scheduling of vehicles from a central depot to a number of delivery points. Operations Research **12** (1964) 568–581
12. Croes, G. A.: A method for solving Traveling Salesman Problems. Operations Research **6** (1958) 791–801
13. Christofides, N., Mingozzi, A. and Toth, P.: The vehicle routing problem. In: Christofides, N., Mingozzi, A., Toth, P. and Sandi, C. (Eds.): Combinatorial Optimization. Wiley, Chicester (1979)
14. Taillard, E. D.: Parallel iterative search methods for vehicle routing problems. Networks **23** (1993) 661–673
15. Rochat, Y. and Taillard, E. D.: Probabilistic Diversification and Intensification in Local Search for Vehicle Routing. Journal of Heuristics **1** (1995) 147–167
16. Rego, C. and Roucairol, C.: A parallel tabu search algorithm using ejection chains for the vehicle routing problem. In: Osman, I. H. and Kelly, J. (Eds.): Meta-Heuristics:Theory and Applications. Kluwer, Boston (1996)
17. Gendreau, M., Hertz, A. and Laporte, G.:A tabu search heuristic for the vehicle routing problem. Management Science **40** (1994) 1276–1290
18. Osman, I. H.: Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. Annals of Operations Research **41** (1993) 421–451
19. Doerner, K. F., Hartl, R.F., and Reimann, M.: Are CompetANTS competent for problem solving - the case of a transportation problem. POM Working Paper 01/2001