A Savings based Ant System for the Vehicle Routing Problem

Marc Reimann
Center for Business Studies
University of Vienna
Vienna, Austria
Tel. ++43 1 4277 38096
Fax: ++43 1 4277 38094
e-mail: marc.reimann@univie.ac.at

Michael Stummer
Center for Business Studies
University of Vienna
Vienna, Austria

Karl Doerner
Center for Business Studies
University of Vienna
Vienna, Austria

Abstract

In this paper we study the merit of using the well known Savings algorithm within the framework of an Ant System to tackle the Vehicle Routing Problem (VRP). First, we show the influence of the pheromone information on the solution quality, by comparing the Ant System with a randomized implementation of the Savings algorithm. Second, we evaluate our approach on benchmark data sets. Finally, we provide numerical results about the average and worst case behavior of our algorithm.

1 INTRODUCTION

In this paper we show how a powerful problem specific algorithm can be incorporated into the framework of an Ant System. This is exemplified for a problem from distribution logistics, namely the Vehicle Routing problem (VRP).

The VRP is a well known combinatorial optimization problem, which has been extensively studied for the last 40 years. It involves the construction of a set of vehicle tours starting and ending at a single depot and satisfying the demands of a set of customers. Constraints ensure that each customer is served by exactly one vehicle, vehicle capacities are not exceeded and a prespecified upper bound on the maximum tour length is respected.

The VRP belongs to the class of NP-hard problems (cf. Garey and Johnson, 1979). Therefore no efficient exact solution methods are available, and the existing solution approaches are of heuristic nature. Recently the focus of research on this problem was on the use of meta-heuristics such as Tabu Search, Simulated Annealing and Ant Systems.

The Ant System is a new meta-heuristic developed in the nineties (cf. Colorni et al., 1991). It's inspiration comes from the observation of trail laying - trail following behavior of some ant species. As they move in search for food, the individual ants of these species deposit an aromatic essence called pheromone on the ground. The amount deposited generally depends on the quality of the food sources found. Other ants, observing the pheromone are likely to follow the pheromone trail, with a bias towards stronger trails. Thus, the pheromone trails reflect the 'memory' of the ant population, and over time trails leading to good food sources will be reinforced while paths leading to remote sources will be abandoned.

Within the framework of the Ant System the above mentioned details were implemented in the following way. The artificial ants construct solutions for a given combinatorial optimization problem by taking a number of decisions probabilistically. First these decisions are only based on some local information (e.g. a heuristic rule), as there is no artificial pheromone available. Gradually the collective memory is built up, as some ants, depending on the solution quality found, are allowed to lay artificial pheromone on the paths they used. The amount of pheromone laid also depends on the solution quality. Other ants are then guided in their decision making. Over time paths with high pheromone concentration will attract more ants than paths with low concentration. Thus, these paths will be reinforced and the artificial ants are (hopefully) guided to promising regions of the search space.

This approach has been applied to a number of combinatorial optimization problems, such as the Graph Coloring Problem (c.f. Costa and Hertz, 1997), the Quadratic Assignment Problem (e.g. Stuetzle and Dorigo, 1999), the Travelling Salesman Problem (e.g. (Dorigo and Gambardella, 1997), (Bullnheimer et al., 1999a)), the Vehicle Routing Problem ((Bullnheimer et al., 1999b), (Bullnheimer et al., 1999c)) and the

Vehicle Routing Problem with Time Windows (Gambardella et al., 1999). Recently, a convergence proof for a generalized Ant System has been developed by Gutjahr (Gutjahr, 2002).

In Doerner et al. (Doerner et al., 2002), we have proposed the incorporation of a problem specific heuristic algorithm, namely the well known Savings algorithm into an Ant System for the VRP. The results there have shown the potential of the method. In this paper we present a modified version of the algorithm, where the modifications stem from observations made on the behavior of our original algorithm. These modifications have led to a significant improvement of the performance. Furthermore, we thoroughly evaluate the algorithm. First, we study the learning behavior by comparing cases with and without learning, respectively. Second, we show that the best results found by our approach are competitive to state of the art results. Third, we examine the average and worst case behavior of our algorithm and the effects of problem characteristics on these measures.

The remainder of this paper is organized as follows. In the next section we provide a problem formulation and an overview of related works on the VRP. After that we give a detailed description of our new approach. Section 4 contains the results of the computational study we performed. We conclude with a discussion of our findings.

2 PROBLEM FORMULATION AND RELATED WORKS

The VRP can be formulated in the following way[1]. Let $G = (V, E, c)$ be a complete graph, with $n + 1$ nodes $(v_0, ..., v_N)$ corresponding to the customers $i = 1, ..., N$ and the depot $i = 0$, and the edge set $((v_i, v_j) \in E \; \forall \; v_i, v_j \in V)$. With each edge $(v_i, v_j) \in E$ is associated a non-negative weight c_{ij}, which refers to the travel costs between nodes v_i and v_j and a non-negative weight t_{ij}, which refers to the travel time between the nodes. Furthermore, with each node $v_i, i = 1, ..., N$ is associated a non-negative demand d_i, which has to be satisfied, as well as a service time δ_i. The service time at the depot is set to $\delta_0 = 0$. At the depot a fleet of size K is available, where each vehicle has a capacity of Q^k and the maximum driving time for each vehicle is T^k.

[1]This formulation is closely related to the formulation presented in (Christofides, 1985).

Let x_{ij}^k denote the binary decision variables with the following interpretation:

$$x_{ij}^k = \begin{cases} 1 & \text{if vehicle } k \text{ visits node } v_j \\ & \text{immediately after node } v_i \\ 0 & \text{otherwise.} \end{cases}$$

Then the objective can be written as

$$minimize \sum_{i=0}^{N} \sum_{j=0}^{N} \sum_{k=1}^{K} c_{ij} x_{ij}^k \tag{1}$$

under the following restrictions

$$\sum_{i=1}^{N} \sum_{j=1}^{N} x_{ij}^k d_i \leq Q^k \quad 1 \leq k \leq K \tag{2}$$

$$\sum_{i=0}^{N} \sum_{j=0}^{N} x_{ij}^k (t_{ij} + \delta_i) \leq T^k \quad 1 \leq k \leq K \tag{3}$$

$$\sum_{i=0}^{N} x_{ij}^k - \sum_{l=0}^{N} x_{jl}^k = 0 \quad 1 \leq k \leq K, 0 \leq j \leq N \tag{4}$$

$$\sum_{i=0}^{N} \sum_{k=1}^{K} x_{ij}^k = \begin{cases} 1 & 1 \leq j \leq N \\ K & j = 0 \end{cases} \tag{5}$$

$$\sum_{i \in S} \sum_{j \in S} x_{ij}^k \leq |S| - 1 \quad \forall S \subseteq \{1, ..., N\}, 1 \leq k \leq K \tag{6}$$

$$x_{ij}^k \in \{0, 1\} \quad 1 \leq k \leq K, 0 \leq i, j \leq N \tag{7}$$

The objective (1) is to minimize the total travel costs. Constraints (2) ensure that no vehicle is overloaded. Constraints (3) require that the maximum driving time for each vehicle is respected. Constraints (4) ensure that if a vehicle visits a customer it also leaves the customer. Constraints (5) require that all customers are visited once, and that the depot is left K times. Subtour elimination is ensured through constraints (6). Finally, constraints (7) are the usual binary constraints.

A large number of researchers have addressed this problem with different approaches. Early approaches dealt with simple heuristics for constructing solutions like the Savings algorithm (Clarke and Wright, 1964) or the Sweep algorithm (Gillett and Miller, 1974), as

well as with the application of more or less sophisticated improvement mechanisms like the 2-opt algorithm (Croes, 1958). We use the idea of the Savings algorithm in our Ant System approach, so we will discuss this algorithm in more detail in the next section.

In the last decade the use of meta-heuristics was investigated. First approaches were based on Tabu Search (c.f. Osman, 1993, Gendreau et al., 1994 and Rego and Roucairol, 1996) and Simulated Annealing (Osman, 1993). Recently an Ant System approach for the VRP was proposed by Bullnheimer et al. ((Bullnheimer et al., 1999b),(Bullnheimer et al., 1999c)).

Overviews on exact and approximate methods can be found in Laporte (Laporte, 1999) and Laporte and Semet (Laporte and Semet, 1999), respectively. Meta-heuristics have been reviewed by Gendreau et al. (Gendreau et al., 1999).

As our approach is based on an Ant System, we will now briefly describe the algorithm proposed for the VRP by Bullnheimer et al. (Bullnheimer et al., 1999c). In their paper, the construction of solutions is done using the Nearest Neighbor algorithm. This algorithm starts with the assignment of an arbitrary customer to the first vehicle and in each decision step chooses to visit the customer closest to the current location until the capacity or time constraints of the vehicle are violated. At this point the vehicle returns to the depot, another vehicle is initialized and the procedure is repeated until all customers are assigned. In the case of the Ant System, this Nearest Neighbor algorithm is made stochastic, such that the closest location is not always chosen, but rather all unvisited locations have a positive probability to be chosen. In Bullnheimer et al. (Bullnheimer et al., 1999c) this probability was calculated via a parametrized Savings function. After an ant has constructed a solution the tours of this solution are improved using the 2-opt algorithm (Croes, 1958). At the end of an iteration, i.e. when all ants have generated their solutions, pheromone is updated according to a rank-based scheme with elitists. This means that not all ants are allowed to update the pheromone information, but only the m best ants. The amount of pheromone written depends on the solution quality found as well as on the rank of the ant. In addition to the best ants of the iteration, the global best solution found during the process is updated as if a number of elitist ants had used it in the current iteration.

3 THE SAVINGS BASED ANT SYSTEM ALGORITHM

In this section we propose our implementation of the Savings based Ant System. The Ant System framework of our algorithm is identical to the one proposed in Bullnheimer et al. (Bullnheimer et al., 1999c) and mainly consists of the iteration of three steps:

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

Our approach differs in the actual implementation of the three steps as described below.

3.1 SOLUTION GENERATION

The solution generation technique we implemented is the main contribution of our work. As discussed above, solution construction in an Ant System for the VRP has so far been based on the Nearest Neighbor construction mechanism. Note, that in this constructive mechanism vehicles are filled one at a time.

As opposed to that, each of our ants constructs a solution based on the well known Savings algorithm (Clarke and Wright, 1964). We will now describe the main structure of this algorithm and propose the modifications we applied in order to be able to use it in the Ant System context.

The Savings algorithm starts from a solution where all customers are served on separate tours. After that for each pair of customers i and j the following savings measure is calculated:

$$s_{ij} = d_{i0} + d_{0j} - d_{ij}, \tag{8}$$

where d_{ij} denotes the distance between locations i and j and the index 0 denotes the depot. Thus, the values s_{ij} 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.

Our modifications are related to the use of pheromone information in the decision making. 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

$$\xi_{ij} = [s_{ij}]^{\beta}[\tau_{ij}]^{\alpha} \tag{9}$$

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 (10), where $\mathcal{P}_{ij}$ is the probability of choosing to combine customers i and j on one tour.

$$\mathcal{P}_{ij} = \begin{cases} \frac{\xi_{ij}}{\sum_{(h,l)\in\Omega_k} \xi_{hl}} & \text{if } \xi_{ij} \in \Omega_k \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

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

3.2 LOCAL SEARCH

After the ants have constructed their solutions but before the pheromone is updated each ants' solution is improved by applying a local search. In the paper by Bullnheimer et al. (Bullnheimer et al., 1999c) as well as in our original algorithm (Doerner et al., 2002) the local search algorithm used was the 2-opt algorithm (c.f. Croes, 1958). 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. In the context of the VRP it was applied separately to all vehicle routes built by the ants. The main idea of this algorithm is to improve the routing of each tour.

In preliminary tests we found that the solutions obtained using only the 2-opt algorithm are acceptable as reported in Doerner et al. (Doerner et al., 2002). While the individual routes were of high quality, the 'sub-optimal' clustering of customers led to the main deviation of these results from the best known solutions.

Thus, we modified the algorithm in the following way. In addition to the 2-opt algorithm, we first apply a local search based on *swap* moves to an ants solution. A *swap* move aims at improving the solution by exchanging two customers from different tours, i.e. a customer i from tour k is exchanged with a customer j from tour l. The idea of this local search is to improve the clustering of the solution. The use of *swap* moves was proposed by Osman (Osman, 1993) for the VRP.

So in our new approach, we first aim to improve the clustering and if no more improvements are possible, we subject each resulting cluster to a 2-opt algorithm in order to improve the routing.

3.3 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 Bullnheimer et al. (Bullnheimer et al., 1999a) the pheromone update is done as follows

$$\tau_{ij} := \rho\tau_{ij} + \sum_{\mu=1}^{m} \Delta\tau_{ij}^{\mu} + \sigma\Delta\tau_{ij}^{*} \tag{11}$$

where $0 \leq \rho \leq 1$ is the trail persistance and $\sigma = m + 1$ 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 m 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 $\Delta\tau_{ij}^{\mu} = (m - \mu + 1)/L^{\mu}$. Arcs belonging to neither of those solutions just lose pheromone at the rate $(1-\rho)$, which constitutes the trail evaporation.

After the pheromone information has been updated the attractiveness values ξ_{ij} are augmented with the new pheromone information as in equation (9).

Note, that in our original approach as proposed in (Doerner et al., 2002), this attractiveness list was re-sorted after each iteration. The intuition for this re-sorting was the following. In the beginning the attractive-

ness 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.

However, this re-sorting lead to fast, and premature convergence and thus was left out of the modified algorithm presented here. This, as a positive side effect, also lead to a minor decrease in computation time which to some degree offset the additional effort needed for the new local search.

4 COMPUTATIONAL STUDY

In this section we will evaluate our proposed approach. First we will describe the standard benchmark problem instances for the VRP. Afterwards we will provide a comparison between a stochastic savings algorithm, where no learning occurs and our new approach, as well as with the approach described in Bullnheimer et al. (Bullnheimer et al., 1999c). Next we will compare our results with state of the art results of other meta-heuristic approaches. Finally, we present information on the average and worst case behavior of our algorithm.

4.1 THE BENCHMARK PROBLEM INSTANCES

All our computations were performed on a set of benchmark problems described in (Christofides et al., 1979). Information on these instances is collected in Table 1.

The instances $C1 - C10$ are random problems, i.e. the customers are located randomly in the plane, while instances $C11 - C14$ are clustered problems, i.e. the customer locations are clustered. All instances are capacity constrained. In addition to that, the instances $C6 - C10$ and $C13 - C14$ are restricted with respect to tourlength. 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: Characteristics Of The Benchmark Problem Instances

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

n ... number of customers
Q ... vehicle capacity
L ... maximum tour length
δ ... service time
best publ. ... best published solution
(a) Taillard, 1993
(b) Rochat and Taillard, 1995

4.2 EVALUATION OF THE LEARNING BEHAVIOR

Let us first analyse the learning behavior of our approach. This will give us a first insight into the performance of the Ant System. As stated above our Ant System is very similar to the Ant System described in (Bullnheimer et al., 1999c). Thus, we chose basically the same parameter settings, namely n artificial ants, $\alpha = \beta = 5$ and $\sigma = 6$ elitist ants.

In preliminary studies we found that for our approach an evaporation rate $\rho = 0.95$ is preferable to $\rho = 0.75$ (as proposed in (Bullnheimer et al., 1999c)). We also varied the population sizes and the number of iterations, but found that n ants and $2 \cdot n$ iterations provide a good compromise between computation time and solution quality. Finally, we tested different sizes of the neighborhood, i.e. different numbers of alternatives in each decision step of an ant and found that $k = \lfloor n/4 \rfloor$ again yields the best results with respect to both computation times and solution quality. More details about these results can be found in (Doerner et al., 2002).

To analyse the learning behavior of our approach and the performance of our new Ant System approach more generally, we compare three cases.

- Savings based Ants: our new approach
- Stochastic Savings algorithm: this refers to our new approach with $\alpha = 0$, i.e. the influence of the pheromone information is deliberately set to zero and no learning occurs
- Standard Ant System: this refers to the algorithm by (Bullnheimer et al., 1999c)

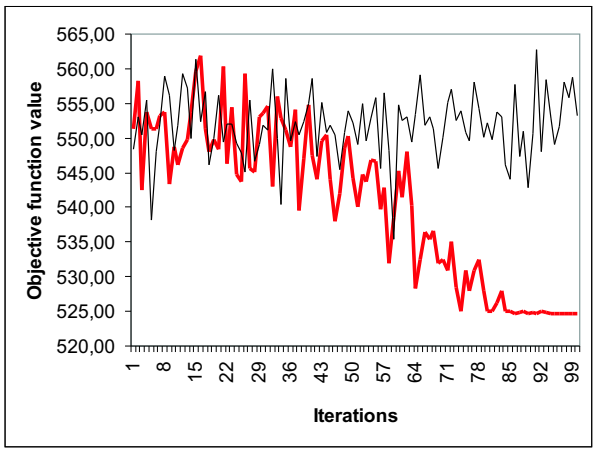

Figure 1: Comparing The Learning Behavior Of Our Savings Based Ants (bold line) With The Stochastic Savings Algorithm

Figure 1 shows the behavior of our new approach as compared to the Stochastic Savings algorithm for a typical run of problem $C1$. We can observe two important results. First, in the beginning the two algorithms are almost identical. The ants have not yet gathered enough pheromone, so the influence of the trails is small. Indeed, it seems that the Stochastic Algorithm finds better solutions more quickly as it is not disturbed by the emerging structure in the pheromone information. Second, the Savings based Ants start to gain from the pheromone memory and continually improve their solutions until they converge to an optimum (which in this case is slightly worse than the global optimum), while the Stochastic Savings algorithm does of course neither learn nor converge to a certain level, but rather oscillates around an average solution that is much worse than the solutions found by our new Savings based Ant System.

Let us now compare the three algorithms with respect to the best solution found over 10 runs for each of the 14 instances. Note, that for all three approaches we generated an identical number of solutions. The results for the three approaches are summarized in Table 2. For each instance we indicate the best solution found with any of the three algorithms in bold.

Table 2: Comparison Of Savings Based Ants With Stochastic Savings And A Standard Ant System

Instance	Savings based AS	SSA	SAS
C1	524.63	530.26	**524.61**
C2	**838.60**	851.63	844.31
C3	**828.67**	847.70	832.32
C4	**1040.09**	1077.57	1061.55
C5	**1303.53**	1356.71	1343.46
C6	**555.43**	560.35	560.24
C7	**909.68**	932.46	916.21
C8	866.87	901.52	**866.74**
C9	**1171.34**	1247.32	1195.99
C10	**1416.05**	1520.17	1451.64
C11	**1042.11**	1047.09	1065.21
C12	**819.56**	**819.56**	**819.56**
C13	**1545.12**	1566.96	1559.92
C14	**866.37**	866.79	**866.37**

SSA...Stochastic Savings algorithm
SAS...Standard Ant System

From Table 2 we can observe two interesting results. First, clearly our new Savings based Ant System outperforms the other two algorithms. In 10 out of the 14 instances it finds strictly better solutions than the other two approaches. For 2 more instances our new approach finds the same solution as one or both of the other algorithms. Only two instances can be solved more effectively with the standard Ant System. However, if we look at the results more closely, we see that our algorithm generally finds significantly better solutions than the other two approaches, whereas when it does not find the best solution, it is very close to the result obtained by the standard Ant System.

Second, for the clustered problems $C11 - C14$ a striking observation can be made. For these instances, the Stochastic Savings algorithm, which performs worst for the random problems, finds very good solutions. More specifically, it is competitive with the Standard Ant System. The structure of these problems seems to be strongly exploited by the Savings algorithm. In fact, as the Savings algorithm tends to combine customers that are close to each other and far from the depot and is able to build tours simultaneously, it is very likely to start building partial tours for each cluster. Thus, the assignment of a customer belonging to a certain cluster, to a tour for another cluster, leading to long unnecessary movements between clusters is unlikely.

4.3 EVALUATION OF OUR SAVINGS BASED ANTS AGAINST STATE OF THE ART RESULTS

Let us now turn the analysis of our algorithm with respect to absolute effectiveness. To that end we compare the results obtained with our Savings based Ant System with two Tabu Search approaches, which are currently the best meta-heuristic approaches for the problem. We will measure the performance in terms of deviation of the best solution obtained with each method from the best known solution available for each instance as presented in Table 1.

The algorithms we compare our Savings based Ant System with are the parallel tabu search algorithm (PTS) from (Rego and Roucairol, 1996) and the TABUROUTE algorithm (TS) from (Gendreau et al., 1994).

The last two rows of Table 3 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 Tabu Search And The Savings Based Ant System

Instance	PTS		TS		Savings based AS	
	RPD	min[1]	RPD	min[2]	RPD	min[3]
C1	0.00	1.05	0.00	6.0	0.00	0.06
C2	0.01	43.4	0.06	53.8	0.40	0.34
C3	0.17	26.3	0.40	18.4	0.31	1.37
C4	1.55	48.5	0.75	58.8	1.14	8.41
C5	3.34	77.1	2.42	90.9	0.94	33.2
C6	0.00	2.38	0.00	13.5	0.00	0.06
C7	0.00	20.6	0.39	54.6	0.00	0.40
C8	0.09	18.9	0.00	25.6	0.11	1.48
C9	0.14	29.9	1.31	71.0	0.76	9.91
C10	1.79	42.7	1.62	99.8	1.45	39.3
C11	0.00	11.2	3.01	22.2	0.00	3.44
C12	0.00	1.57	0.00	16.0	0.00	1.33
C13	0.59	1.95	2.12	59.2	0.26	7.22
C14	0.00	24.7	0.00	65.7	0.00	1.44
	RPD (avg.)					
C1-C10	0.71		0.70		0.51	
C11-C14	0.15		1.28		0.06	

[1]Minutes on 4 parallel Sun Sparc 4 machines.
[2]Minutes on a Silicon Graphics Workstation (36MHz).
[3]Minutes on a Pentium III (900 MHz).

From Table 3 it can be clearly seen, that our algorithm outperforms the two Tabu Search approaches with respect to solution quality. The average deviation of our Savings based Ant System over all instances is only 0.38%, while the parallel tabu search is on average 0.55% away from the best known solution, and the TABUROUTE algorithm has an average deviation of 0.81%. Particularly, on the clustered problems the superior performance of our algorithm can be observed. This is due to the fact, that the Savings algorithm itself is known to perform very well on the clustered problems. This was also confirmed by our results in the last section.

Looking at computation times, we have to consider the following issues: first, the machines used differ greatly. Moreover, the PTS algorithm, was performed on parallel machines. Second, the times given for the Tabu Search approaches denote the time to find the best solution, whereas our computation times denote the time to perform $2 \cdot n$ solutions. For most problems the best solution was found earlier in the search process. Keeping these points in mind, it seems that our approach finds competitive solutions very fast as compared to the other methods.

4.4 AVERAGE AND WORST CASE BEHAVIOR OF THE PROPOSED METHOD

So far we have evaluated our approach according to the best solutions found for each instance in 10 trials. In this section we will take a look at the average and worst case behavior of our algorithm in these 10 trials for each of the 14 instances.

What is mainly of interest, is the question whether the average or worst deviation depends on the problem size and characteristics. Therefore we will now provide results for the different problem sizes and for the different problem characteristics, namely random and clustered.

Let us first look at problem characteristics. In Table 4 we can see the average and worst case behavior of our Savings based Ant System, averaged over the random and clustered problems, respectively.

From these results it becomes once again obvious that our algorithm performs particularly well on the clustered problems. However, the worst case behavior of 1.92% for the random problem seem still to be more than reasonable.

While we have now confirmed the strong performance of our Savings based Ant System for the clustered problems let us finally turn to the question how av-

Table 4: Influence Of Problem Characteristics On The Average And Worst Case Behavior Of Our Savings Based Ant System

Deviation in %	**Random Problems**	**Clustered Problems**
Average	1.10	0.14
Worst Case	1.92	0.18

erage and worst case behavior relate to the problem size. In order to answer this question, we have only looked at problems $C1 - C10$. The intuition for this is the following. The clustered problems $C11 - C14$ have sizes of 100 and 120 customers. Thus, they are medium sized. However, due to their clustered structure they can be better solved than problems with random distribution of customers of equal and even smaller size. In order not to bias our results on the dependence of average and worst case behavior on problem size we have thus decided to ignore problems $C11 - C14$.

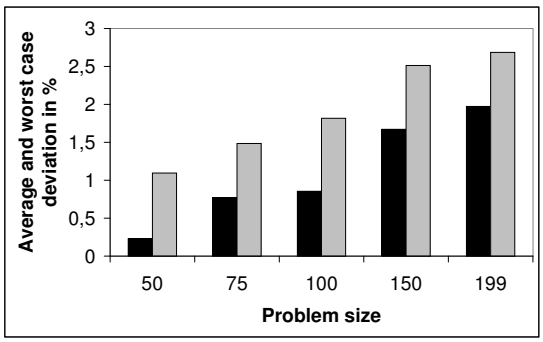

Figure 2: Average(Dark Columns) And Worst Case Behavior Of Our Savings Based Ants For Different Problem Sizes

Figure 2 shows that both average and worst case results increase with problem size however at a decreasing rate. This result is particularly encouraging as it suggests that even for larger problems our Savings based approach should be able to find robust results. However, the validity of this statement needs to be tested in future work.

5 CONCLUSIONS

In this paper we have investigated the merit of the incorporation of a powerful problem specific algorithm for the VRP, namely the Savings algorithm, into an Ant System framework.

We have first presented a mathematical formulation, followed by an overview of existing research in the area. Afterwards we have described our approach in detail.

Through tests on the standard benchmark problem instances we were able to show, that our algorithm exhibits adaptive learning, outperforms existing Ant System as well as state of the art Tabu Search approaches and shows satisfying average and worst case behavior.

Finally, we should add, that our approach features two important issues concerning real world problems. First, these problems are generally clustered, as in cities the density of customer locations is higher than in rural areas. We showed, that for clustered problems our algorithm works particularly well. Second, computation time is often crucial in real world applications, and our results suggest, that our Savings based Ant System compares favorably to other techniques with respect to this objective.

Acknowledgments

This work was supported by the Austrian Science Foundation under grant SFB #010 'Adaptive Information Systems and Modelling in Economics and Management Science' and by the Oesterreichische Nationalbank (OeNB) under grant #8630. We are grateful to Herbert Dawid and four anonymous referees for their valuable comments on the paper.

References

B. Bullnheimer, R. F. Hartl and Ch. Strauss (1999a): A new rank based version of the ant system: a computational study. *Central European Journal of Operations Research* **7**(1):25–38.

B. Bullnheimer, R. F. Hartl and Ch. Strauss (1999b): Applying the ant system to the vehicle routing problem. In: S. Voss et al. (eds.), *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, Kluwer, Boston, 285–296.

B. Bullnheimer, R. F. Hartl and Ch. Strauss (1999c): An improved ant system algorithm for the vehicle routing problem. *Annals of Operations Research* **89**:319–328.

N. Christofides, A. Mingozzi and P. Toth (1979): The vehicle routing problem. In: N. Christofides et al. (eds.), *Combinatorial Optimization*, Wiley, Chicester, 315–338.

N. Christofides (1985): Vehicle Routing. In. E. L. Lawler et al. (eds.), The Traveling Salesman Problem, Wiley, Chicester, 431–448.

G. Clarke and , J. W. Wright (1964): Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research* **12**:568–581.

A. Colorni, M. Dorigo and V. Maniezzo (1991): Distributed Optimization by Ant Colonies. In: F. Varela and P. Bourgine (eds.), *Proceedings of the First European Conference on Artificial Life*, Elsevier, Amsterdam, 134–142.

D. Costa, A. Hertz (1997). Ants can colour graphs. *Journal of the Operational Research Society* **48**(3):295–305.

G. A. Croes (1958). A method for solving Traveling Salesman Problems. *Operations Research* **6**:791–801.

K. Doerner, M. Gronalt, R. F. Hartl, M. Reimann, Ch. Strauss and M. Stummer (2002): SavingsAnts for the Vehicle Routing Problem. In S. Cagnoni et al.(eds.), *Applications of Evolutionary Computing*, Springer LNCS 2279, Berlin/Heidelberg, 11–20.

M. Dorigo and L. M. Gambardella (1997): Ant Colony System: A cooperative learning approach to the Travelling Salesman Problem. *IEEE Transactions on Evolutionary Computation* **1**(1):53–66.

L. M. Gambardella, E. Taillard and G. Agazzi (1999): MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing Problems with Time Windows. In D. Corne et al.(eds.), *New Ideas in Optimization*, Mc Graw-Hill, London, 63–73.

M. R. Garey, D. S. Johnson (1979). *Computers and Intractability: A Guide to the Theory of NP Completeness.* W. H. Freeman & Co., New York.

M. Gendreau, A. Hertz and G. Laporte (1994): A tabu search heuristic for the vehicle routing problem. *Management Science* **40**:1276–1290.

M. Gendreau, G. Laporte and Y. Potvin (1999): Metaheuristics for the vehicle routing problem. GERAD Technical report G-98-52.

B. E. Gillet and L. R. Miller (1974): A heuristic algorithm for the vehicle-dispatch problem. *Operations Research* **22**:340–349.

W. J. Gutjahr (2002): ACO algorithms with guaranteed convergence to the optimal solution. *Information Processing Letters* **82**:145–153.

G. Laporte (1999): Exact algorithms for the travelling salesman problem and the vehicle routing problem. GERAD Technical report G-98-37.

G. Laporte and F. Semet (1999): Classical heuristics for the vehicle routing problem. GERAD Technical report G-98-54.

I. H. Osman (1993): Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. *Annals of Operations Research* **41**:421–451.

C. Rego and C. Roucairol (1996): A parallel tabu search algorithm using ejection chains for the vehicle routing problem. In: I. H. Osman and J. Kelly (eds.), *Meta-Heuristics:Theory and Applications*, Kluwer, Boston, 661–675.

Y. Rochat and E. D. Taillard (1995): Probabilistic Diversification and Intensification in Local Search for Vehicle Routing. *Journal of Heuristics* **1**:147–167.

T. Stützle, M. Dorigo (1999): ACO Algorithms for the Quadratic Assignment Problem. In D. Corne et al. (eds.), *New Ideas in Optimization*, Mc Graw-Hill, London, 33–50.

E. D. Taillard (1993): Parallel iterative search methods for vehicle routing problems. *Networks* **23**:661–673.