Dynamic Ant Colony Optimisation

Daniel Angus and Tim Hendtlass

Centre for Intelligent Systems and Complex Processes
Swinburne University of Technology
VIC 3122
Australia
{182921,thendtlass }@swin.edu.au

Abstract. Ant Colony optimisation has proved suitable to solve static optimisation problems, that is problems that do not change with time. However in the real world changing circumstances may mean that a previously optimum solution becomes suboptimal. This paper explores the ability of the ant colony optimisation algorithm to adapt from the optimum solution to one set of circumstances to the optimal solution to another set of circumstances. Results are given for a preliminary investigation based on the classical travelling salesperson problem. It is concluded that, for this problem at least, the time taken for the solution adaption process is far shorter than the time taken to find the second optimum solution if the whole process is started over from scratch.

Keywords: Meta-heuristics, Optimisation, Ant Colony Optimisation.

1 Introduction

Solving optimisation problems presents considerable challenges to researchers and practitioners alike. These often intractable problems usually arise in large industries such as telecommunications, transportation and electronics where even slight increases in solution quality can translate to increased company profit, lower consumer prices and improved services. As a result of this, numerous optimisation techniques have been studied, developed and refined.

The traditional operations research techniques of branch and bound, cutting planes and dynamic programming have been widely used, but can be computationally expensive for large and complex problems. As a result, newer meta-heuristic search algorithms such as simulated annealing [11], tabu search [7] and genetic algorithms [8] have been applied as they generally find good solutions in a moderate amount of computational time.

To further complicate the problem, external factors can, and often will, alter the resources available to perform the task, with the result that a previously optimal, or at least efficient, solution can suddenly become sub-optimal or even impossible. Most methods require that the whole optimisation process be redone with the new resource constraints, which may be a very time consuming task.

The most recent trend is to use algorithms inspired by evolutionary concepts. This paper uses an ant colony optimisation algorithm (ACO) [5] to solve a dynamic optimisation problem. ACO is a collection of meta-heuristic techniques. They are multi agent systems in which agents (ants) have very little individual intelligence but evolve a collective intelligence about a problem over time using simulated chemical markers. Unlike evolutionary algorithms, ACO does not use generational replacement, but intra-generational interaction.

Real ants do not retreat to their nest and start over if something (e.g. a foot) blocks their current efficient path, rather they adapt the path to suit the new constraint. This paper shows how an ACO algorithm can be structured so that it can adapt to a change in constraints

2 The ACO Algorithm

ACO is modeled on the foraging behaviour of Argentine ants. The seminal work by Dorigo [2] showed that this behaviour could be used to solve discrete optimisation problems. This section gives a brief overview of the ant colony mechanics using the Ant Colony System (ACS) meta-heuristic and the travelling salesperson problem (TSP) together with other applications. Those readers familiar with the concepts of ACO and TSP may skip Section 2.1.

2.1 Algorithm description

It is convenient to describe ACS with the TSP metaphor. Consider a set of cities, with known distances between each pair of cities. The aim of the TSP is to find the shortest path that traverses all cities exactly once and returns to the starting city. The ACS paradigm is applied to this problem in the following way. Consider a TSP with n cities. Cities i and j are separated by distance $d(i,j)$. Scatter m ants randomly on these cities ($m \leq n$). In discrete time steps, all ants select their next city then simultaneously move to their next city. Ants deposit a substance known as 'pheromone' to communicate with the colony about the utility (goodness) of the edges. Denote the accumulated strength of pheromone on edge (i,j) by $\tau(i,j)$.

At the commencement of each time step, Equations 1 and 2 are used to select the next city j for ant k currently at city i. Equation 1 is a greedy selection technique favouring cities which possess the best combination of short distance and large pheromone levels. Equation 2 partially balances this by allowing a probabilistic selection of the next city.

$$j = \left\{ \begin{array}{c} \arg\max_{u \in J_i^k} \left\{ \left[\tau_{iu}\left(t\right)\right] \cdot \left[\eta_{iu}\right]^{\beta} \right\} \\ J \end{array} \right\} \begin{array}{l} \text{if q} \leq \text{q}_0 \\ \text{if q} > \text{q}_0 \end{array} \tag{1}$$

Where $\eta_{ij} = 1/d_{ij}$ and $J \in J_i^k$ is chosen according to the probability:

$$p_{ij}^k\left(t\right) = \frac{\left[\tau_{ij}\left(t\right)\right] \cdot \left[\eta_{ij}\right]^{\beta}}{\sum_{l \in J_i^k} \left[\tau_{il}\left(t\right)\right] \cdot \left[\eta_{il}\right]^{\beta}} \tag{2}$$

Note that $q \in [0,1]$ is a uniform random number and q_0 is a parameter. To maintain the restriction of unique visitation, ant k is prohibited from selecting a city that it has already visited. The cities that have not yet been visited by ant k are indexed by $J_k(t)$. The parameter β controls how much importance is placed on the distance to the next city in choosing the next path. Linear dependence on $\tau(ij)$ ensures preference is given to links that are well traversed (i.e. have a high pheromone level). The pheromone level on the selected edge is updated according to the local updating rule in Equation 3.

$$\tau_{ij}(t) \leftarrow (1-\rho) \cdot \tau_{ij}(t) + \rho \cdot \tau_0 \tag{3}$$

Where:

ρ is the local pheromone decay parameter, $0<\rho<1$.

τ_0 is the initial amount of pheromone deposited on each of the edges.

Upon conclusion of an iteration (i.e. once all ants have constructed a tour), global updating of the pheromone takes place. Edges that compose the best solution are rewarded with an increase in their pheromone level, which is expressed in Equation 4.

$$\tau_{ij}(t) \leftarrow (1-\rho) \cdot \tau_{ij}(t) + \rho \cdot \Delta\tau_{ij}(t) \tag{4}$$

Where:

$$\Delta\tau_{ij}(t) = 1/L^{+} \tag{5}$$

L^{+} is the length of the best (shortest) tour found so far.

Unless otherwise stated, values used in experiments were:

$\beta = 2$, $\rho = 0.1$, $q_0 = 0.9$, $\tau_0 = 0.00002149$, $m = 14$

2.2 Algorithm modification for dynamic problems

It is assumed that the algorithm knows when the constraints have been changed. If the algorithm is running continuously and it has been some time since the current solution being used was found the pheromone levels on the edges associated with that solution may be high. Even so some ants will still be exploring alternate paths owing to the stochastic component of the ant decision making process. The speed of adaption may be compromised if very high levels of pheromone are allowed to remain. If all edges on the best path have their pheromone set to a small number while the pheromone level at all other edges is set to zero, all information regarding the relative merits of alternate paths would have been removed. A suitable compromise between these two extreme actions is to normalise pheromone levels on a city's edges. That is for city i the j pheromone levels τ_{ij} are replaced by τ_{ij}/τ_{imax} where τ_{imax} is the maximum pheromone level on any of city i's edges.

3 The test problem used

According to Dorigo and Di Caro [5], the TSP is a popular test problem for ACO methods because a) it is relatively easy to adapt the ACO meta-heuristics to this problem and b) it is a problem that is intuitively understood and as such it is one of the most studied problems in the combinatorial optimisation literature. In addition, this class of meta-heuristics is relatively new and research has therefore been directed at defining and exploring its features using well understood problems (such as the TSP). Many papers have solved the TSP using ACO methods including the seminal Dorigo and Gambardella [3, 4], Dorigo, Maniezzo and Colorni [6] and Stützle and Dorigo [10] (this paper contains a summary of all ACO applications to the TSP to date).

The Burma 14 data set is a useful small test data set for TSP algorithms. It is a symmetric TSP consisting of 14 pairs of data points, as shown in table 1.The node coordinates correspond to a geographical latitude and longitude of the corresponding point on the earth. TSPLIB95 [9] contains a detailed explanation of the calculation of distances using a geographical data set.

City	Latitude	Longitude	City	Latitude	Longitude
0	16.47	96.10	**7**	17.20	96.29
1	16.47	94.44	**8**	16.30	97.38
2	20.09	92.54	**9**	14.05	98.12
3	22.39	93.37	**10**	16.53	97.38
4	25.23	97.24	**11**	21.52	95.59
5	22.00	96.05	**12**	19.41	97.13
6	20.47	97.02	**13**	20.09	94.55

Table 1. The coordinates of the Burma 14 data set.

Burma 14 has a large but manageable number of solutions, many of which have lengths that are very close to the absolute minimum path length (see figure 1). With this data set, non-exhaustive search methods have a high probability of finding a good but not optimal answer. However, if they can do this in a reasonably short time, they may be suitable methods for real life applications that require as good an answer as can be obtained in the available time.

The best path solution is shown in figure 2.

4 Results and Discussion

The results of 10000 trials of an ACS solving this data set, each from a different random starting position, are shown in figure 3. Only 18 times in 10000 trials was the minimum path found, with the second best path being found 2974 times. However the speed with which the solutions were found is remarkable, being an

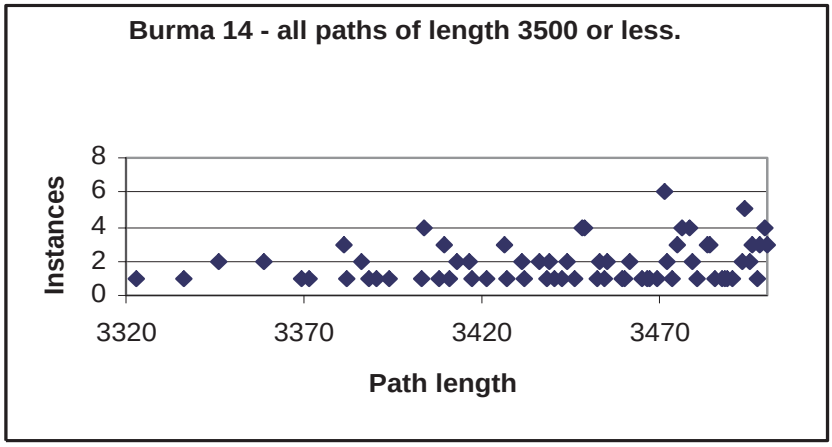

Fig. 1. The frequency of all possible paths with a length less than 3500.

average of 90 milliseconds (433 iterations on average). For comparison, using the same machine, an efficient exhaustive search took two and a half hours.

To simulate a changing environment the ants were allowed to first find a good (but not necessarily ideal) path for the full data set. When stability was achieved (arbitrarily chosen to be when 500 iterations had passed without any change in best path found) one city was removed from the path and the ants were required to adapt the now nonviable path to a new path that met the altered constraints.

The best path lengths for the altered data set are shown in table 2.

4.1 Adapting a very good 14 city solution to a reduced set of cities.

After finding the second best solution to the 14-city problem (path length equal to 3336) one city was removed from the list of cities to be visited by the ants. Before the ants were allowed to continue, the pheromone levels at each city were normalised relative to the path segment involving that city with the highest pheromone concentration. Without normalisation the very high pheromone levels on what was the best path would inhibit the ants from exploring other options, leading to path adaptation times that approached the time required to solve the reduced problem from scratch. Reducing the pheromone levels increased the relative importance of visibility while still retaining access to the collective wisdom that had been previously developed concerning the value of path segments. Path segments that used to lead to a now removed city could no longer be followed by an ant and so decisions that would have lead to travel to the removed city would now be based on visibility and the remaining (probably) low pheromone levels. The average results obtained from 10000 runs (with random initialisation) adapting to a smaller city set are shown in table 3. These results

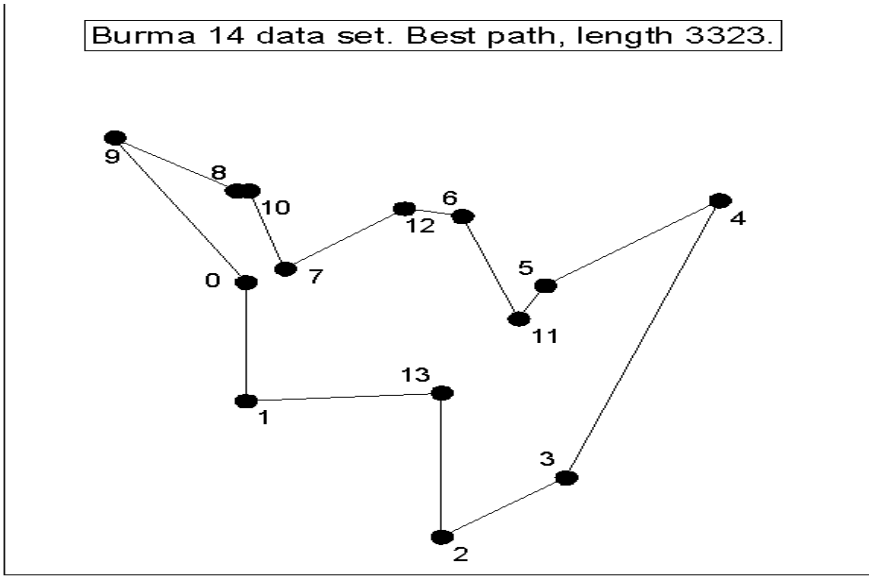

Fig. 2. The shortest path for the Burma 14 data set.

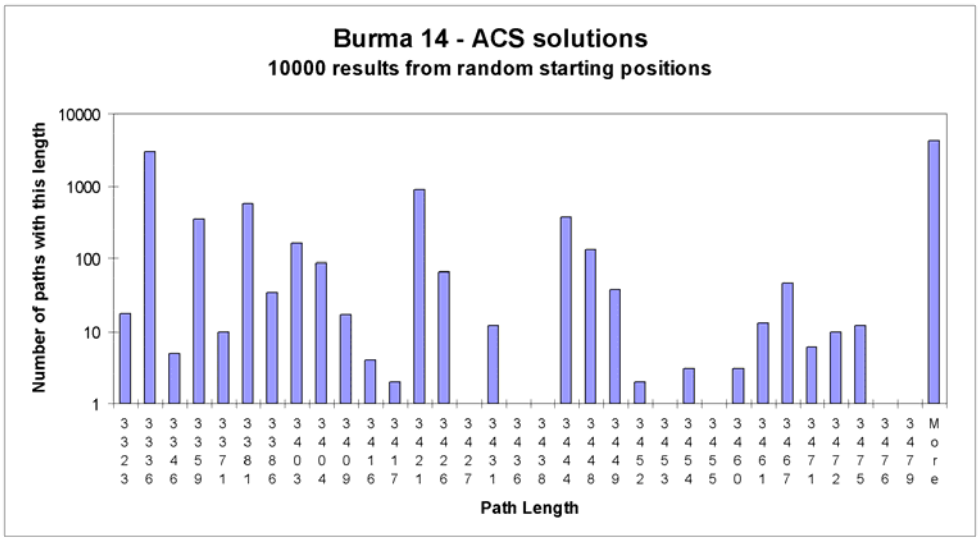

Fig. 3. The path lengths found using ant colony optimisation.

Data set	Number of paths	Shortest length
Burma14	3,113,510,400	3323
Burma14 minus city 0	239,500,800	3277
Burma14 minus city 1	239,500,800	3218
Burma14 minus city 2	239,500,800	3133
Burma14 minus city 3	239,500,800	3161
Burma14 minus city 4	239,500,800	2696
Burma14 minus city 5	239,500,800	3322
Burma14 minus city 6	239,500,800	3311
Burma14 minus city 7	239,500,800	3232
Burma14 minus city 8	239,500,800	3322
Burma14 minus city 9	239,500,800	2808
Burma14 minus city 10	239,500,800	3301
Burma14 minus city 11	239,500,800	3309
Burma14 minus city 12	239,500,800	3315
Burma14 minus city 13	239,500,800	3158

Table 2. The data sets used for these experiments.

indicate a decrease in the amount of time required to find a new solution utilizing adaptation without a substantial loss in solution quality, and in some cases increased solution quality using adaptation. Table 4 outlines the most commonly found solutions which all lay within the top 5 best solutions for the respective data sets. The table highlights the most common path length found (after the indicated city has been removed), the percentage of times (out of 10,000 runs) this solution was found, and the average iterations taken to reach a solution for that specific path. Table 5 shows three detailed examples of good, average and poor quality solutions using adaptation versus starting from scratch. Good solutions being when adaptation outperformed starting from scratch, average being when the two methods were comparable and poor being when starting from scratch outperformed adaptation.

4.2 Adapting a very good less than 14 city solution to a 14 city solution.

Tests were also conducted in which, after first establishing a good path for a smaller city set, one or more cities were added to the list. Again stability on the smaller path was declared after 300 iterations without improvement in the best path length and the pheromone levels were normalised before the ants were started on the revised problem. The average results obtained from 100 runs (with random initialisation) are presented in a previous paper [1]. The previous work outlined the speed with which a new city is adapted into the existing solution and a negligible path increase when compared to restarting the problem and neglecting any previous pheromone information. Based on the previous findings

City removed	Adapting		Starting from scratch	
	Average iterations	Average length	Average iterations	Average length
0	408	3506	490	3392
1	399	3365	459	3292
2	316	3313	403	3338
3	327	3408	422	3445
4	318	2809	408	2826
5	392	3538	466	3498
6	380	3433	428	3385
7	404	3510	388	3435
8	314	3500	376	3494
9	339	2971	383	2909
10	376	3566	353	3500
11	324	3366	436	3411
12	417	3470	467	3407
13	446	3278	418	3199

Table 3. Average results from 10000 runs.

City removed	Adapting	From Scratch
	Path Length / % found path / Average iterations	Path Length / % found path / Average iterations
0	3310 / 13.45% / 191	3310 / 25.30% / 620
1	3195 / 24.02% / 408	3195 / 35.50% / 672
2	3146 / 39.83% / 194	3146 / 15.10% / 531
3	3174 / 2.04% / 282	3219 / 0.50% / 45
4	2709 / 22.13% / 257	2709 / 10.20% / 542
4	2710 / 21.74% / 224	2710 / 11.70% / 420
5	3335 / 17.06% / 301	3335 / 19.70% / 534
6	3324 / 43.36% / 243	3324 / 23.00% / 584
7	3232 / 6.05% / 246	3232 / 9.90% / 533
8	3335 / 21.40% / 189	3335 / 11.40% / 481
9	2808 / 12.27% / 210	2808 / 23.30% / 402
10	3336 / 1.13% / 564	3336 / 3.40% / 492
11	3322 / 79.39% / 223	3322 / 33.50% / 563
12	3336 / 20.04% / 454	3336 / 21.80% / 674
13	3171 / 34.08% / 536	3171 / 39.60% / 597

Table 4. Commonly found paths. Except for city 4 which had two close paths (both commonly found) removing other cities produced one unique best path.

City Removed	Path place	Path length	Adapting			From Scratch	
			Segment changes	Occurrences	Average iterations	Occurrences	Average iterations
11 (good)	1	3309	4	0.04%	**274**	**0.10%**	625
	2	3322	1	**79.39%**	**223**	33.50%	563
	3	3332	3	0.09%	**14**	**1.30%**	23
	6	3367	3	1.58%	**304**	**8.00%**	320
0 (average)	1	3277	3	0.57%	**117**	**12.5%**	719
	2	3300	5	0.00%	-	**0.40%**	378
	3	3310	1	13.45%	**191**	**25.3%**	620
	4	3333	5	2.84%	**458**	**4.50%**	663
	6	3340	3	0.76%	**88**	**1.90%**	486
13 (poor)	1	3158	4	9.93%	785	**12.60%**	**486**
	2	3171	1	34.08%	**536**	**39.60%**	597
	3	3181	6	9.55%	562	**8.70%**	**376**
	5	3194	3	2.62%	208	**0.60%**	**203**

Table 5. Good, average and poor solutions using adaptation versus starting from scratch.

it is evident that adding cities would yield similar results to removing cities, with advantages in speed far outweighing disadvantages in solution quality.

4.3 Solution quality.

Time is not the only measure of success, the quality of the solution is also important. It should be noted that the results found, while good, were not in general optimal. Varying the relative importance of visibility versus pheromone shows an inverse relationship between the time taken and the quality of the solution found. This relationship appears to hold when finding both the original path and adapting to a new path using ACS. A detailed analysis of this relationship is outlined in a previous paper [1].

5 Conclusion.

Although the results above are obtained using a small, although non-trivial, TSP data set, they give a clear indication that ants can adapt a path in a shorter time than would be required to find that new path starting over. As the size of the TSP city set is increased, ACO shows an improved performance compared to alternate techniques. It might be anticipated that the adaptation advantage shown by ants on the B14 city set will be at least maintained on larger data sets.

Varying the importance of the pheromone compared to the city visibility allows a trade off to be made between the time taken to reach a solution and the quality of that solution. This may prove a useful control parameter in the context of needing to find as good a solution as possible to a problem within a definite and restricted time.

No matter if starting from scratch, or if adapting a previously found solution to suit a modified set of constraints, the quality of all the solutions found for this data set are within the top 0.01% of all possible solutions. This augurs well for the use of such an approach to real life problems with dynamic constraints, for example in scheduling applications.

References

1. Angus D and Hendtlass T. (2002) "Ant Colony Optimization Applied to Dynamically Changing Problem" Developments in Applied Artificial Intelligence. LNAI2358, Springer. pp 618-627.
2. Dorigo, M. (1992) Optimization, Learning and Natural Algorithms, PhD Thesis, Dipartimento di Elettronica, Politechico di Milano, Italy.
3. Dorigo, M. and Gambardella, L. (1997) "Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem", IEEE Transactions on Evolutionary Computing, 1, pp. 53-66.
4. Dorigo, M and Gambardella, L. (1997) "Ant Colonies for the Traveling Salesman Problem", Biosystems, 43, pp. 73-81.
5. Dorigo, M. and Di Caro, G. (1999) "The Ant Colony Optimization Meta-heuristic", in New Ideas in Optimization, Corne, D., Dorigo, M. and Golver, F. (eds), McGraw-Hill, pp. 11-32.
6. Dorigo, M., Maniezzo, V. and Colorni, A. (1996) "The Ant System: Optimization by a Colony of Cooperating Agents", IEEE Transactions on Systems, Man and Cybernetics - Part B, 26, pp. 29-41.
7. Glover, F. and Laguna, M. (1997) Tabu Search, Kluwer Academic Publishers, Boston: MA, 442 pages.
8. Goldberg, D. (1989) Genetic Algorithms in Search, Optimization and Machine Learning, Addison Wesley: Reading, MA, 412 pages.
9. Reinelt G. TSPLIB95. Available from http://www.iwr.uni-heidelberg.de/iwr/comopt/soft/TSPLIB95/TSPLIB95.html
10. Stützle, T. and Dorigo, M. (1999) "ACO Algorithms for the Traveling Salesman Problem", in Evolutionary Algorithms in Engineering and Computer Science, Miettinen, K., Makela, M., Neittaanmaki, P. and Periaux, J. (eds), Wiley.
11. van Laarhoven, L. and Aarts, E. (1987) Simulated Annealing: Theory and Applications, D Reidel Publishing Company: Dordecht, 186 pages.