ASGA: Improving the Ant System by Integration with Genetic Algorithms

Tony White, Bernard Pagurek, Franz Oppacher[1]
Systems and Computer Engineering, Carleton University,
1125 Colonel By Drive, Ottawa, Ontario, Canada K1S 5B6
email: {tony, bernie}@sce.carleton.ca

ABSTRACT

This paper describes how the Ant System can be improved by self-adaptation of its controlling parameters. Adaptation is achieved by integrating a genetic algorithm with the ant system and maintaining a population of agents (ants) that have been used to generate solutions. These agents have behavior that is inspired by the foraging activities of ants, with each agent capable of simple actions. Problem solving is inherently distributed and arises as a consequence of the self-organization of a collection of agents, or swarm system. This paper applies the Ant System with Genetic Algorithm (ASGA) system to the problem of path finding in networks, demonstrating by experimentation that the hybrid algorithm exhibits improved performance when compared to the basic Ant System.

1. Introduction

The notion of complex collective behavior emerging from the behavior of many simple agents and their interactions is central to the ideas of Artificial Life [Langton, 87]. There are many examples in Nature of social systems where individuals possess simple capabilities which, when compared to their collective behaviors, are much more complex. Such systems span many levels of evolutionary complexity, from simple bacteria [Shapiro, 88], to ants [Goss et al, 90], [Franks, 89], caterpillars [Fitzgerald and Peterson, 88] and beyond.

The continuing investigation and research of naturally-occurring social systems offers the prospect of creating artificial systems that are controlled by emergent behavior and promises to generate engineering solutions to distributed systems management problems such as are found, for example, in telecommunications networks.

In this paper we describe the essential principles of Swarm Intelligence (SI) and, in particular, how an understanding of the foraging behaviors of ants [Beckers et al, 92] has led to new approaches to control and management in telecommunications networks and a new combinatorial optimization technique, Ant Search.

Ant Search applications [Kunz and Snyers, 94; Colorni et al, 92] have identified the need for appropriate choice of control parameters. It is this observation that motivates the research reported in this paper. This paper proposes the integration of Genetic Algorithms with an Ant Search algorithm for the purpose of enhancing the latter.

This paper consists of six major sections. In the next section, a brief overview of Swarm Intelligence and Ant Colony search is presented with a brief review of applications of Ant Search (AS) being provided. The next section describes the Ant System with Genetic Algorithms (ASGA) system and provides an overview of the algorithm used. The following section describes the three path-finding problems that are the subject of investigation in this paper. Experimental setup and results are then detailed for the three network path finding problems described. The final sections provide a summary and a statement of future work that is planned with this system and derivatives of it.

2. Swarm Intelligence and the Ant Colony

Swarm Intelligence [Beni and Wang, 89] is a property of systems of unintelligent agents of limited individual capabilities exhibiting collectively intelligent behavior. An agent in this definition represents an entity capable of sensing its environment and undertaking simple processing of environmental observations in order to perform an action

[1] Email: oppacher@scs.carleton.ca

chosen from those available to it. These actions include modification of the environment in which the agent operates. Intelligent behavior frequently arises through indirect communication between the agents, this being the principle of stigmergy [Grassé, 59]. It should be stressed, however, that the individual agents have no explicit problem solving knowledge and intelligent behavior arises (or emerges) because of the actions of societies of such agents.

Individual ants are behaviorally simple insects with limited memory and exhibiting activity that has a stochastic component. However, collectively ants manage to perform several complicated tasks with a high degree of consistency. Examples of sophisticated, collective problem solving behavior have been documented [Frank, 89; Hölldobler and Wilson, 94] including:

- Forming bridges
- Nest building and maintenance
- Cooperating in carrying large items
- Finding the shortest routes from the nest to a food source
- Regulating nest temperature within a one degree Celsius range
- Preferentially exploiting the richest source of food available.

In the examples listed above, two forms of stigmergy have been observed. Sematectonic stigmergy involves a change in the physical characteristics of the environment. Ant nest building is an example of this form of communication in that an ant observes a structure developing and adds its ball of mud to the top of it. The second form of stigmergy is sign-based. Here something is deposited in the environment that makes no direct contribution to the task being undertaken but is used to influence the subsequent behavior that is task related.

Sign-based stigmergy is highly developed in ants. Ants use highly volatile chemicals called pheromones (a hormone) to provide a sophisticated signaling system. Ants foraging for food lay down quantities of pheromone marking the path that it follows with a trail of the substance. An isolated ant moves essentially at random but an ant encountering a previously laid trail will detect it and decide to follow it with a high probability and thereby reinforce it with a further quantity of pheromone. The collective behavior which emerges is a form of autocatalytic behavior where the more the ants follow the trail the more likely they are to do so. The process is characterized by a positive feedback loop, where the probability that an ant chooses any given path increases with the number of ants choosing the path at previous times.

The use of ant foraging behavior as a metaphor for a problem-solving technique is generally attributed to Dorigo [Dorigo et al, 91]. In the AS, problem solving can loosely be described as finding an optimal path through a network. Since his early work on the Travelling Salesman Problem (TSP) and Asymmetric TSP, the technique has been applied to several other problem domains. These include the Quadratic Assignment Problem (QAP) [Maniezzo et al, 94; Taillard et al, 97], graph coloring [Costa and Hertz, 97], vehicle routing [Bullnheimer et al, 97] and communications network routing. [Schoonderwoerd et al, 97] and [Di Caro and Dorigo, 97] have applied AS to routing problems in communication networks. Both applications of AS in this domain have been in a control setting with the former group using AS for circuit switched path routing while the latter have used it for packet switched routing. Both groups report improvements in network throughput by application of ant-inspired algorithms.

Variations on the basic AS have also been proposed. Local search operations have been proposed [Stützle and Hoos, 97], as has the integration of reinforcement learning [Leerink et al, 95], or Q-learning [Gambardella and Dorigo, 95].

3. The ASGA System

The AS is characterized by a number of controlling parameters such as the number of iterations of the algorithm, the number of agents (ants) used per iteration, the sensitivity to the cost of a link in the network and the sensitivity to pheromone concentration. The ASGA system allows the parameters controlling the sensitivity to the cost of a network link and the sensitivity to pheromone concentration to adapt as the search algorithm proceeds.

The ASGA system integrates a genetic algorithm with an ant system for the purpose of controlling the adaptation process. Each agent in the ASGA system encodes the sensitivity to link and sensitivity to pheromone parameters. The ASGA system algorithm can be described in pseudo-code as follows:

```
Procedure ASGA
Begin
  InitializePopulation.
  For n := 1 to numberOfIterations do:
  Begin
    For k :=1 to populationSize do:
      UseAgentToSolveProblem(k)
    For k := 1 to populationSize do:
      ReinforcePathFollowedByAgent(k)
    GenerateAgentParameters
  End
End
```

A population of agents is maintained in ASGA, with each member of the population having been used to solve the problem in question using an AS algorithm. Each agent

has a fitness value associated with the solution found. For example, in the case of the problem of finding a simple route in a network or a travelling salesman problem, this might be the inverse of the cost of the route found.

The *InitializePopulation* function randomly generates values for pheromone and cost sensitivities and assigns them to agents; i.e. the first iteration of agents is constructed. The *GenerateAgentParameters* function is used to generate the cost and pheromone sensitivity values for the next iteration. This function follows the standard mechanism of a genetic algorithm [Goldberg, 89]. Agents from the current generation (P) are sampled, with replacement, to generate an intermediate population (Q). This intermediate population is then manipulated genetically, using crossover and mutation operators to form a second intermediate population (R). Finally, the merging of the populations P, Q and R produces the next generation of agents, (S). The *UseAgentToSolveProblem* function uses a typical AS migration algorithm for movement of agents from node to node in the network. The migration decision function is given by:

$$p_k(i,j) = [T(i,j)]^{\alpha_k}[C(i,j,u)]^{-\beta_k} / N_k \qquad (1)$$

$$N_k = \Sigma_{j \in S_k(i)} [T(i,j)]^{\alpha_k}[C(i,j,u)]^{-\beta_k}$$

Where:

- i and j are vertex indices,
- u(i,j) is the utilization of the link between i and j,
- T(i,j) is the quantity of pheromone present on the link between the ith and jth nodes,
- C(i,j,u) is the cost associated with the link between the ith and jth nodes with utilization u(i,j),
- S_k (i) is set of nodes not yet visited for the kth agent that can be reached from the ith node,
- p_k(i,j) is the probability that the kth agent will choose the edge from the ith to the jth node as its next hop,
- α_k is the sensitivity to pheromone for the kth agent,
- β_k is the sensitivity to the cost of the network link for the kth agent.

To re-iterate, the values α_k and β_k are encoded genetically in the agent's description.

Once all agents have been routed for a given iteration the *ReinforcePathFollowedByAgent* function is used to update the T(i,j) values based upon the routes found by the agents. The update rule is given by:

$$T(i,j) \leftarrow T(i,j) + \Sigma_k \Delta T_k(i,j) \qquad (2)$$

Where:

$$\Delta T_k(i,j) = \begin{cases} f(Cost_k) & (i,j) \in R_k \\ 0 & otherwise \end{cases}$$

- R_k is the path found for agent k,
- $Cost_k$ is the cost of path found for agent k,
- $f(Cost_k)$ is the function used to update pheromone concentrations on the edges in the path computed by agent k.

4. Path Finding with ASGA

The ASGA system has been applied to three classes of problem, all of which are related to routing in networks. First, the problem of finding a shortest path between two points in a network has been solved. While this is not a hard problem for a single route, with Dijkstra's algorithm [Dijkstra, 59] often being used for shortest path calculations, it is much harder when multiple paths are considered and balanced loading of the network is required. The figure below highlights a simple shortest path between the vertices n1 and n5.

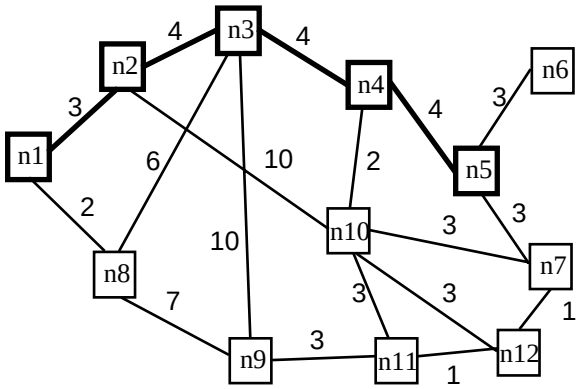

Figure 1: A example simple path

Second, the problem of computing an optimal path connecting a subset of the points in the network has been solved. This problem requires that a spanning tree be computed that connects the subset of the network vertices that are on the path. In this problem, the algorithm has to identify Steiner vertices, that is vertices that are to be included in the minimum spanning tree along with the vertices to be connected. These vertices represent the multi-cast nodes in point to multi-point routing. This is an NP hard problem and previous research has applied genetic algorithms for its solution [Mann et al, 95]. In the aforementioned work, the genetic algorithm identifies the multi-cast nodes (a node is either a multi-cast node or not) and then uses the Prim-Kruskal algorithm in order to compute the minimum spanning tree. While effective, the solution provided by the augmented ant system *directly* computes the minimum spanning tree without further algorithmic involvement. The figure on the next page shows a potential path between vertices {n1, n10, n4 and n7}. In the example shown in the figure the vertices n2 and n3 have been included as Steiner vertices.

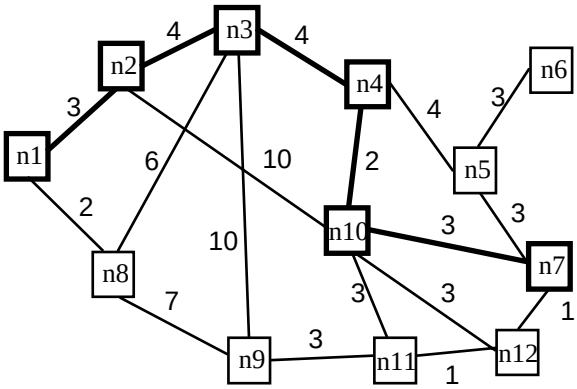

Figure 2: An example multi-point path

Finally, the problem of finding the shortest cycle between two points in the network has been solved. In this problem the algorithm has to compute two independent routes between the source and destination points whose combined cost is minimal across all possible routes. This problem can be described another way by identifying the nodes that are to be included in the cycle -- Steiner vertices -- and computing the shortest cycle possible. Clearly, this problem is related to the TSP. Described as above the problem is known to be NP-complete. The figure below gives an example of a cycle computed between vertices n1 and n10.

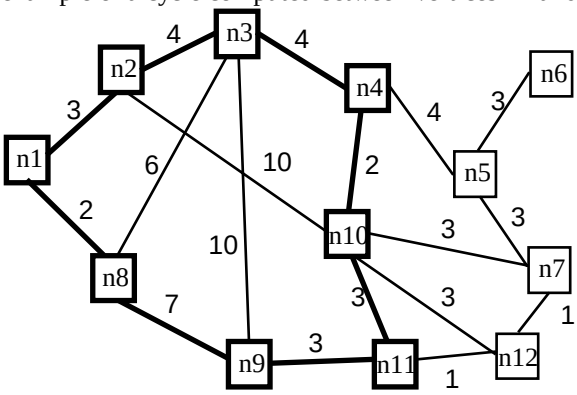

Figure 3: An example cyclical path

4.1. AS solutions for path finding problems

The following three paragraphs describe the migration algorithms and reinforcement terms used by the AS to solve the three path finding problems described above.

Problem 1: Point to point path finding

In this problem agents start at the source node and use the migration decision function in equation (1) to move from the source to the destination node. The fitness (f_k) of the path found by the agent is the inverse of the sum of the costs of the edges used in the path from source to destination. The reinforcement term $\Delta T_k(i,j)$ of equation (2) is computed for the edges used by the kth agent as Qf_k, where Q is a constant.

Problem 2: Point to multi point path finding

Consider for a moment the problem of connecting m+1 nodes, i.e. one source and m destinations. This can be viewed as m point to point requests that must be satisfied simultaneously. The migration decision function in equation (1) is again used to define movement from one node to the next. However, the fitness of the paths found for the individual agents use the inverse of the sum of the costs of the edges with edges counted only once. The reinforcement term $\Delta T_k(i,j)$ of equation (2) is computed for **all** edges used by the agents as Qf_k, i.e., reinforcement is equal for the agents associated with the m point to point requests.

Problem 3: Cyclical path finding

The cyclical path finding problem is similar to the point to point path finding problem except that path finding continues beyond the destination and back to the source node. The migration decision function of equation (1) is used to move agents from one node to the next and the fitness of the path found is the inverse of the sum of the costs of the edges used in the source destination and destination to source path fragments. The reinforcement term $\Delta T_k(i,j)$ of equation (2) was, once again, given by Qf_k.

4.2. Experimental Setup

Two graphs were used during experimental investigation of the ASGA system for the three problems outlined above. These are shown in figures 4 and 5 on the next page. The numbers associated with the edges in these networks represent the costs of the edges at zero edge utilization. Each edge is considered to have a capacity of 63 units.

For problem one, the point to point path finding scenario, ten randomly generated traffic profiles were created for all source-destination pairs with bandwidth requirements sampled uniformly from the set {0, 2, 4, 6, 8, 10} bandwidth units. A bandwidth requirement of zero units was taken to mean that no path need be calculated for the source-destination pair. Paths were calculated using ASGA, with the utilization of the network increasing by the bandwidth requirements of the traffic as paths emerged. All paths were computed in parallel. Initial network edge utilizations of 0, 30, and 50% were considered in order to test the effects of four different cost functions. For problem three, the same randomly generated traffic profiles were used for experimentation. For problem two, ten randomly generated traffic profiles were created with either 2,3 or 4 destinations. Bandwidth requirements for the point to multi point requests were identical to problem one.

A population size of 50 was used with path emergence considered to have occurred when 90% of the population follow a given path. A maximum of 100 cycles (or generations) of the ASGA algorithm was allowed before

path calculations were stopped and 20 agents per cycle were sent out into the network for path finding. The value of α_k was allowed to vary in the range -0.25 to 3 and the value of b_k was allowed to vary in the range -0.125 to 1.5. A total of 8 bits was allowed for the encoding of α_k and also for b_k. When ASGA was contrasted with AS with constant α_k and b_k, values of 2 and 1 respectively were used. These constant values were found to be a reasonable compromise for AS path finding. A value of 10 was chosen for Q. An indirect representation was used with mapping of bit strings into floating point values in the above ranges in such a way as to cover the ranges uniformly. Values of 0.8 and 0.01 were used for the probabilities of crossover and mutation respectively. Two-point crossover was used as the crossover operator.

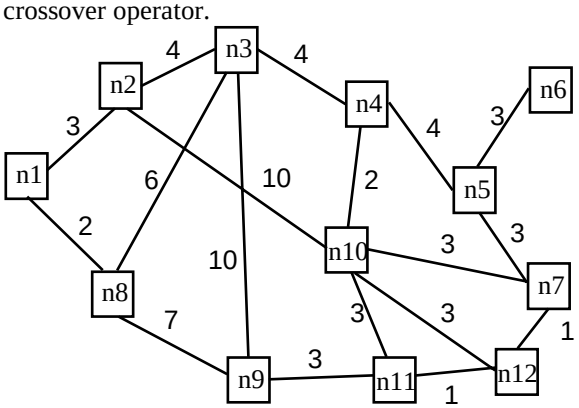

Figure 4: Experimental Graph I

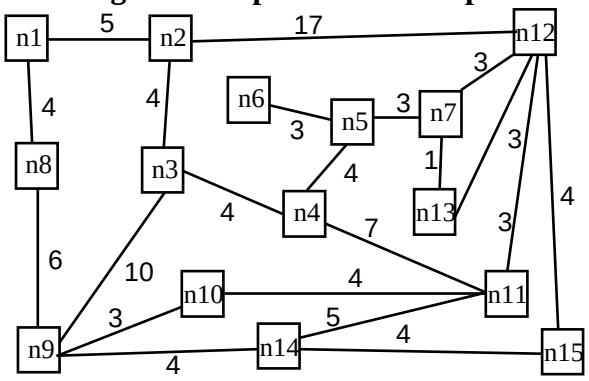

Figure 5: Experimental Graph II

Four cost functions were used in the experiments. These are shown in figure 6. These cost functions are all functions of the utilization of the capacity of the network edge. It should be noted that figure 6c is a constant implying that the cost is independent of edge utilization.

The equations for figures 6a to 6d are given by:

$$6a: \forall x \in [0,1],\ f(x) = 9x + 1$$

$$6b: 0.25 < x < 0.75,\ f(x) = (18x - 14/4)$$

$$6c: \forall x \in [0,1],\ f(x) = 1$$

$$6d: \forall x \in [0,1],\ f(x) = x^3(x+8) + 1$$

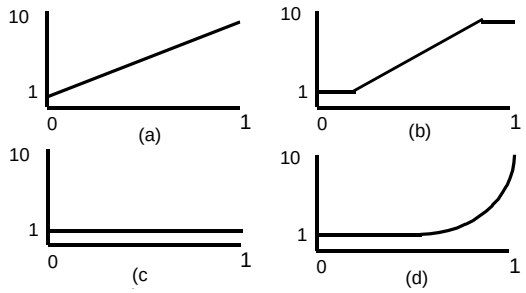

Figure 6: Cost functions

4.3. Results for Problem 1

Tables 1, 2 and 3 contain the results of experiments for constant α and β for the two experimental graphs with different initial edge utilizations for the path finding problem. Tables 4, 5 and 6 contain the results of experiments where α and β were allowed to adapt during search. The mean and standard deviations of run times are reported in tables 1 to 6.

Table 1: 0% Initial Utilization, $\alpha\beta$ constant

		graph1	graph2
	Mean (secs)	25.12	31.22
6a	std dev. (secs)	2.68	3.42
	Mean (secs)	32.67	37.94
6b	std dev. (secs)	4.11	4.62
	Mean (secs)	27.56	30.22
6c	std dev. (secs)	2.76	3.34
	Mean (secs)	22.18	25.89
6d	std dev. (secs)	2.07	3.01

Table 2: 30% Initial Utilization, $\alpha\beta$ constant

		graph1	graph2
	Mean (secs)	26.11	32.21
6a	std dev. (secs)	2.99	3.98
	Mean (secs)	27.33	31.43
6b	std dev. (secs)	3.22	4.76
	Mean (secs)	22.99	29.16
6c	std dev. (secs)	2.41	3.11
	Mean (secs)	22.18	25.89
6d	std dev. (secs)	2.07	3.01

Table 3: 50% Initial Utilization, $\alpha\beta$ constant

		graph1	graph2
	Mean (secs)	25.55	32.28
6a	Std dev. (secs)	2.71	3.72
	Mean (secs)	33.69	38.96
6b	Std dev. (secs)	4.21	4.81
	Mean (secs)	33.11	35.65
6c	Std dev. (secs)	3.99	4.29
	Mean (secs)	22.18	25.89
6d	Std dev. (secs)	2.07	3.01

Table 4: 0% Initial Utilization, αβ adaptive

		graph1	graph2
	Mean (secs)	18.91	23.99
6a	Std dev. (secs)	1.52	2.98
	Mean (secs)	28.11	28.01
6b	Std dev. (secs)	2.31	2.41
	Mean (secs)	20.09	24.22
6c	Std dev. (secs)	1.55	2.11
	Mean (secs)	17.1	19.67
6d	Std dev. (secs)	1.39	1.65

Table 5: 30% Initial Utilization, αβ adaptive

		graph1	graph2
	Mean (secs)	19.11	24.11
6a	Std dev. (secs)	1.6	2.77
	Mean (secs)	21.2	25.22
6b	Std dev. (secs)	1.91	3.51
	Mean (secs)	17.01	22.11
6c	Std dev. (secs)	1.61	2.41
	Mean (secs)	17.1	19.67
6d	Std dev. (secs)	1.39	1.65

Table 6: 50% Initial Utilization, αβ adaptive

		graph1	graph2
	Mean (secs)	20.18	24.91
6a	Std dev. (secs)	2.01	2.22
	Mean (secs)	23.99	28.99
6b	Std dev. (secs)	2.31	2.54
	Mean (secs)	24.95	27.11
6c	Std dev. (secs)	2.88	3.11
	Mean (secs)	22.18	25.89
6d	Std dev. (secs)	2.07	3.01

By comparing tables 1 and 4, 2 and 5, 3 and 6, the results indicate that the ASGA system is clearly superior to the simple AS. ASGA is typically 25% faster than the simple AS in finding paths for the two experimental graphs across the range of initial utilizations chosen. The standard deviations for search times also decrease when comparing the adaptive to the non-adaptive systems, again indicating that adaptation of α,β values has improved predictability of the search process.

The results for 0, 30 and 50% initial edge utilization for cost function 6c being the same in tables 1, 2 and 3 can be explained by the fact that this cost function does not vary with the utilization of the network edges. By the same argument, tables 4, 5 and 6 show no variation in search times for cost function 6c.

The mean edge utilization results are not included here due to space restrictions. However, cost function 6d provided the best overall results when reviewing the standard deviation on edge utilization.

4.4. Results for Problem 2

Tables 7 and 8 contain the results of experiments for constant and adaptive α,β for the two experimental graphs respectively. In both of these tables an initial edge utilization of zero was used. Results for an initial edge utilization of 30% and 50% are not presented are they exhibit similar patterns to those provided for problem one and provide no further insight into the effects of the cost function chosen.

Table 7: 0% Initial Utilization, αβ constant

		graph1	graph2
	Mean (secs)	62.91	80.01
6a	std dev. (secs)	6.99	9.01
	Mean (secs)	82.01	99.11
6b	std dev. (secs)	11.1	11.91
	Mean (secs)	70.65	80.11
6c	std dev. (secs)	7.01	9.91
	Mean (secs)	56.99	65.85
6d	std dev. (secs)	5.61	7.91

Table 8: 0% Initial Utilization, αβ adaptive

		graph1	graph2
	Mean (secs)	45.31	55.91
6a	std dev. (secs)	3.41	7.15
	Mean (secs)	67.41	67.19
6b	std dev. (secs)	5.45	5.55
	Mean (secs)	47.34	56.67
6c	std dev. (secs)	3.51	5.01
	Mean (secs)	41.4	42.99
6d	std dev. (secs)	3.03	3.97

Once again, comparing tables 7 and 8, the adaptive system is clearly shown to be superior to the simple AS.

4.5. Results for Problem 3

Tables 9 and 10 contain the results of experiments for constant and adaptive α,β for the two experimental graphs respectively for the problem of finding a cyclical path between two nodes in the network. In both of these tables an initial edge utilization of zero was used. Results for an initial edge utilization of 30% and 50% are again not presented are they exhibit similar patterns to those provided for problem one and thus provide no further insight into the effects of the cost function chosen.

Table 9: 0% Initial Utilization, αβ constant

		graph1	graph2
	Mean (secs)	51.5	64.98
6a	Std dev. (secs)	5.94	7.11
	Mean (secs)	68.91	76.98
6b	Std dev. (secs)	8.97	10.01
	Mean (secs)	60.12	64.09
6c	Std dev. (secs)	5.92	7.98
	Mean (secs)	48.3	56.74
6d	Std dev. (secs)	4.91	6.71

Table 10: 0% Initial Utilization, αβ adaptive

		graph1	graph2
	Mean (secs)	38.56	49.02
6a	Std dev. (secs)	3.01	5.87
	Mean (secs)	57.6	55.37
6b	Std dev. (secs)	4.32	4.58
	Mean (secs)	43.09	49.41
6c	Std dev. (secs)	3.07	4.76
	mean (secs)	34.12	41.56
6d	std dev. (secs)	2.9	3.21

As tables 9 and 10 clearly show, the adaptive system outperforms the AS with constant α and β values and exhibits improvements that are similar to those observed for problems one and two.

It should be noted that when the search for a path starts, the cost element of the probability function dominates the calculation of $p_k(i,j)$, and an almost greedy heuristic comes into play, i.e. the least cost edge is probabilistically chosen. The actual value of α is relatively unimportant, as only a small amount of pheromone is present on the edges. Therefore, the main factor influencing choice of links is the actual cost of that edge. As routes are found, pheromone is laid on the edges that form that path. The amount of pheromone laid is inversely proportional to the total cost of the path found, and acts as a global measure of 'goodness'. As the quantity of pheromone rises, this brings the reinforcement part of the probability function, $p_k(i,j)$, into play. The sensitivity to pheromone, α, influences the choice of edges, and edges with higher pheromone concentrations are more likely to be chosen. It can easily be seen, therefore, that the importance of edge cost and pheromone concentration varies throughout the search process and that adaptation of the appropriate sensitivity parameters likely to be desirable.

Finally, allowing negative values for α and β has introduced the ability for agents to choose low pheromone concentration, high cost, edges. This has improved the ability of the system to recover from situations where agents have laid down large concentrations of pheromone on a non-optimal path early on in the search process.

5. Conclusions

This paper has described an adaptive agent system that solves three distinct path-finding problems in networks. This paper has demonstrated that the AS can be used to solve hard combinatorial optimization problems as represented by Steiner vertex identification and shortest cycle determination. Our experimental results have clearly demonstrated that self-adaptation of control parameters has led to improvements in system performance.

6. Future Work

Work is ongoing to apply the ASGA system to the TSP and asymmetric TSP as proposed in [Dorigo et al, 96]. Also, [Gambardella and Dorigo, 95] have proposed Ant-Q, a family of combinatorial optimization algorithms that represent the synthesis of Q-Learning and AS. Future work will extend ASGA to incorporate Ant-Q algorithms in order to determine the utility of adaptation in this new family of search algorithms.

References

Beckers R., Deneuborg J.L and Goss S. 1992. Trails and U-turns in the Selection of a Path of the Ant Lasius Niger. In *J. theor. Biol*. Vol. 159, pp. 397-415.

Beni G., and Wang J. 1989. Swarm Intelligence in Cellular Robotic Systems, *Proceedings of the NATO Advanced Workshop on Robots and Biological Systems*, Il Ciocco, Tuscany, Italy.

Bullnheimer B., R.F. Hartl and C. Strauss. 1997, Applying the Ant System to the Vehicle Routing Problem. *2nd Metaheuristics International Conference (MIC-97)*, Sophia-Antipolis, France.

Colorni A., M. Dorigo & V. Maniezzo. 1992. An Investigation of Some Properties of an Ant Algorithm. Proceedings of the Parallel Problem Solving from Nature Conference (PPSN 92), Brussels, Belgium, R.Männer and B.Manderick (Eds.), Elsevier Publishing, 509-520.

Costa D. and A. Hertz. 1997. Ants Can Colour Graphs. *Journal of the Operational Research Society*, 48, 295-305.

Di Caro G. and Dorigo M. 1997. AntNet: A Mobile Agents Approach to Adaptive Routing. Tech. Rep. IRIDIA/97-12, Université Libre de Bruxelles, Belgium, 1997.

Dijkstra E.W. 1959. A Note on Two Problems in Connexion with Graphs In *Numerische Mathmatik* vol. 1.

Dorigo M., V. Maniezzo & A. Colorni. 1991. The Ant System: An Autocatalytic Optimizing Process. Technical Report No. 91-016, Politecnico di Milano, Italy.

Dorigo M., V. Maniezzo & A. Colorni. 1996. The Ant System: Optimization by a Colony of Cooperating

Agents. IEEE Transactions on Systems, Man, and Cybernetics-Part B, 26, 1, 29-41

Franks N.R. 1989. Army Ants: A Collective Intelligence, *Scientific American*, Vol. 77.

Fitzgerald T.D., Peterson S.C. 1988. Cooperative foraging and communication in caterpillars, *Bioscience*, 38, pp. 20-25.

Gambardella L.M. & M. Dorigo. 1995. Ant-Q: A Reinforcement Learning Approach to the Traveling Salesman Problem. Proceedings of ML-95, Twelfth International Conference on Machine Learning, Tahoe City, CA, A. Prieditis and S. Russell (Eds.), Morgan Kaufmann, 252-260.

Goldberg, D. 1989. Genetic Algorithms in Search, Optimization, and Machine Learning. Reading, MA: Addison-Wesley.

Goss S., Beckers R., Deneubourg J.L., Aron S., Pasteels J.M. 1990. How Trail Laying and Trail Following Can Solve Foraging Problems for Ant Colonies, in Hughes R.N. (ed.) *NATO ASI Series, Vol. G 20, Behavioural Mechanisms of Food Selection*, Springer Verlag, Berlin.

Grassé P.P. 1959. La reconstruction du nid et les coordinations inter-individuelles chez Bellicoitermes natalenis et Cubitermes sp. La theorie de la stigmergie: Essai d'interpretation des termites constructeurs. In *Insect Societies*, Vol. 6, pp. 41-83.

Hölldobler B. and Wilson E.O. 1994. *Journey to the Ants*. Bellknap Press/Harvard University Press, 1994.

Kuntz P. and D. Snyers. 1994. Emergent Colonization and Graph Partitioning. Proceedings of the Third International Conference on Simulation of Adaptive Behavior: From Animals to Animats 3, MIT Press, Cambridge, MA.

Leerink L.R., S.R. Schultz and M.A. Jabri. 1995. A Reinforcement Learning Exploration Strategy based on Ant Foraging Mechanisms. Proceedings of the Sixth Australian Conference on Neural Networks, Sydney, Australia, 1995.

Langton, C.G. 1987. Artificial Life, Proceedings of an Interdisciplinary Workshop on the Synthesis and Simulation of Living Things, Los Alamos, New Mexico, Addison Wiley.

Maniezzo V., A. Colorni and M. Dorigo. 1994. The Ant System Applied to the Quadratic Assignment Problem. Tech. Rep. IRIDIA/94-28, Université Libre de Bruxelles, Belgium.

Mann, J., White, T., and Turner, J. 1995. Optimal Route Finding in ATM Networks Using Genetic Algorithms, in Proceedings 7th BNR Design Forum, December, 1995.

Shapiro, J. A. 1988. Bacteria as multi cellular organisms, *Scientific American*, pp. 82-89.

Schoonderwoerd R., O. Holland and J. Bruten. 1997. Ant-like Agents for Load Balancing in Telecommunications Networks. *Proceedings of Agents '97*, Marina del Rey, CA, ACM Press pp. 209-216, 1997.

Stützle T. and H. Hoos. 1997. The MAX-MIN Ant System and local Search for Combinatorial Optimization Problems: Towards Adaptive Tools for Global Optimization. 2nd Metaheuristics International Conference (MIC-97), Sophia-Antipolis, France - July 21-24, 1997.

Taillard E. and L. M. Gambardella. 1997. An Ant Approach for Structured Quadratic Assignment Problems. *2nd Metaheuristics International Conference (MIC-97)*, Sophia-Antipolis, France.