AGENT-BASED PROBLEM SOLVING: THE ANT COLONIES METAPHOR

Stefan Wagner[1], Michael Affenzeller[1], Ismail Khalil Ibrahim[2]

[1]Institute of Systems Science
Johannes Kepler University Linz
Austria
{sw,ma}@cast.uni-linz.ac.at

[2]Department of Telecooperation
Johannes Kepler University Linz
Austria
ismail@tk.uni-linz.ac.at

Abstract

In this paper the analogy between biological swarms and artificial multiagent systems is pointed out. As an example the steps required to model the artificial optimization technique called Ant Colony Optimization starting from the foraging behaviour of natural ant colonies are explained in detail. During the development of the model the authors use the language of multiagent systems to highlight the suitability of such an approach.

1. Introduction

It is known from the field of complexity theory the effort needed to solve a problem is not equal for all different kinds of problems. Generally it is necessary to distinguish between two different classes of problems: On the one hand there are the so called P (or polynomial) problems whose complexity is determined by a polynomial function. On the other hand there is the class of NP (or non-polynomial) problems which are not determined by polynomials. In other words the number of steps that is necessary to solve an instance of the NP-problems increases exponentially (or even worse) with the size of the instance itself. Consequently it is not possible to solve such problems exactly in acceptable time with the algorithms and computing power which are available nowadays.

However, the difficulty is that many practical problems like route planning, scheduling, creating of timetables, etc. belong to the NP-class. As these problems need to be solved anyhow the only possibility is to use approximation techniques instead which are unable to find the optimal solution to a given problem but a solution that is good enough for the specific application.

When designing such optimization techniques nature has proven to be a very good archetype. Many of today's state-of-the-art algorithms are directly inspired by natural processes (e.g. Evolutionary Algorithms, Simulated Annealing, and Particle Swarm Optimization). In the last years especially biological swarm systems have attracted the interest of various research groups. Many of these natural collectives that consist of rather simple individuals are able to deal with very complex problems.

Furthermore, if one looks closer to the developments in the area of systems design and software development, it is obvious that a new paradigm is emerging. In recent years agent-based approaches were intensively studied and applied to many different systems ([6]). Due to the structural similarity of multiagent systems with biological swarms it is consequently rather proximate to try to build optimization algorithms by modelling natural systems in the context of multiagent systems (e.g. [7]).

In this paper, we are going to try to follow this trend by taking a closer look at the Ant Colony Optimization, an optimization algorithm inspired by the foraging behaviour of ant colonies. Thereby the main focus lies on the aspect of how this technique can be modelled in terms of multiagent systems.

2. Ant Colonies – The Natural Archetype

When observing natural ant colonies it can easily be seen that the complexity of each single ant is rather low. Due to its size the ant's cognitive abilities are very limited and it can perceive only a very local area of the environment. When an ant has discovered a food source there is no mechanism like e.g. the bees' dance to communicate its location. Moreover, the ant itself is also not able to perform any length optimization of the path between the found food source and the ant nest.

However, in nature it is observable that a whole ant colony is able to discover the shortest paths between the nest and food sources very efficiently ([1],[5]). So the question is how this implicit optimization is done by the whole colony, if the abilities of the single colony individuals are so limited.

The answer to that mystery is rather simple: While walking ants drop a substance called *pheromone*. Other ants are able to smell this leftover of their fellows and tend to choose a path with a higher pheromone concentration with higher probability. Consequently pheromone trails are established and after some time all ants of the colony will walk on these trails due to the high pheromone concentration.

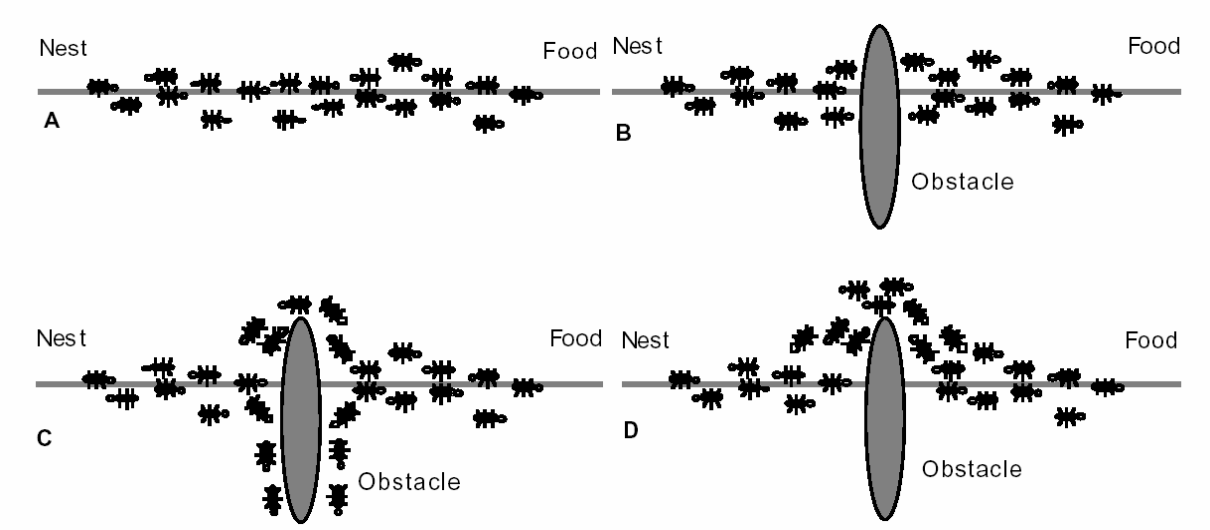

Figure 1: Path optimization performed by the foraging behaviour of ant colonies.

This rather simple biological mechanism enables the ant colony to find the shortest way between two points. In Figure 1 this process is graphically visualized. Assume that the ants have already established a continuous pheromone trail between their nest and the food source (A). If any

obstacle is placed on that trail, it is interrupted and the ants standing right in front of the obstacle don't have any idea where to go (B). Consequently they try to find a way around the obstacle and as they have no idea which of the two possible paths is shorter, approximately half of the ants will randomly choose the upper (shorter) path and the other half will take the longer lower way (C). As the upper path is the shorter one the ants that have chosen it will arrive on the other side of the obstacle much faster than their mates on the lower path. Therefore the number of ants that walk over the shorter path will be higher and therefore also the pheromone concentration on the shorter path will increase more quickly than on the longer one. As a consequence after some time all ants arriving at the obstacle will choose the shorter path due to the higher pheromone concentration (D). So the colony as a whole was able to find the shortest path between the nest and the food source again.

This simple but very effective technique inspired M. Dorigo and led to the development of *Ant Colony Optimization* (ACO), which he introduced in his Ph.D. thesis in 1992 [2]. In the following chapters the reader is guided through the most essential modelling steps starting from a natural ant colony and ending up with the ACO-algorithm. However, in contrast to Dorigo's approach, we look at the whole model from a multiagent based system perspective.

3. Creating an Agent Model Inspired by Nature

The first necessary step when trying to simulate the behaviour of real ants for solving optimization problems is to think concretely about the problem that should be solved. Certainly the problem of shortest paths between the ants nest and some food source is very relevant for an ant colony. However, in a technical and scientific meaning the solving of combinatorial, NP-hard optimization problems is the interesting task. Therefore it is necessary to transform the problem in a way that the natural mechanisms are still applicable and the achieved solution is appropriate. Such transformation can be done in the following way:

Solutions to combinatorial optimization problems are mostly composed of some kind of iterative sequence of steps or components. Consequently, the stepwise procedure of constructing a valid solution can be seen as the path of a virtual ant that is starting from the nest (an empty solution), walking iteratively from one step to another assembling a solution and trying to reach a complete and valid solution to the problem.

After having reformulated the problem according to the field of combinatorial optimization the next step is to generate a model for the natural archetype by identifying the appropriate counterparts of the natural components. To do so the PAGE-description is used. PAGE stands for *Percepts, Actions, Goals and Environment* and is a possible way to describe the most important characteristics of an agent. As we are thinking of natural ants as agents it seems appropriate to start with the PAGE-description of a real ant (Table 1).

As the main aspects of the real ant are quite clearly stated in Table 1 the next step requires some adaptation in order to model a virtual agent capable of finding solutions to combinatorial optimization problems.

Percepts	pheromone concentration at the current position → food at the current position
Actions	walk → drop pheromone
Goals	find food
Environment	real world

Table 1: PAGE-description of a real ant.

3.1. Environment

According to the modification of the main objective from finding the shortest paths to constructing optimal solutions we have to consider a very abstract environment. Virtual agents are situated on a graph of solution components. Each node of this graph represents a certain component of a solution and the nodes are connected among each other to describe the step of appending the next component to the current solution fragment of an agent. Moreover, there is also a special node representing no solution at all which is the starting point for all agents and can therefore be seen as the agents nest.

By walking around in this environment the agents can iteratively construct a solution by assembling the various components. Additionally, the virtual agents also must have a possibility to communicate with each other. This can be realized very analogically to real ants: Each edge of the graph is weighted by a number representing the current amount of pheromone dropped on that edge.

Furthermore, it is necessary that the environment provides some information about the costs of each possible step in terms of solution quality to prevent the agents from performing mere random search, if no pheromone information is available. So it is necessary to label the edges also with a so called heuristic value.

To visualize the typical structure of an ACO-environment Figure 2 shows an example for a TSP instance consisting of 5 cities.

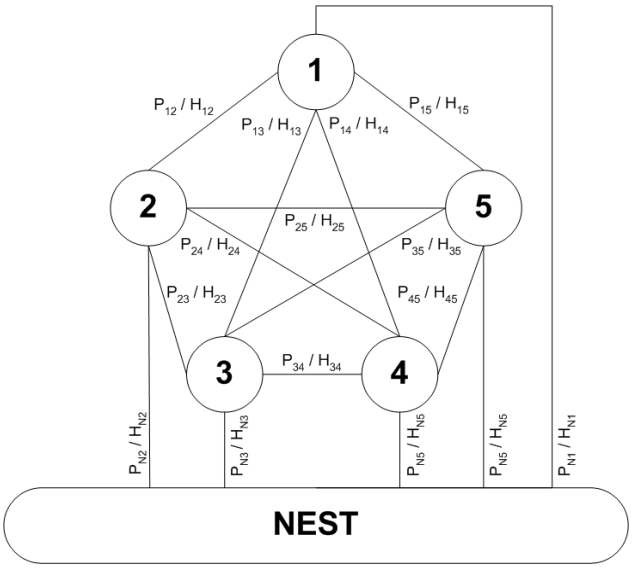

Figure 2: A typical ACO-environment for a 5 cities TSP.

Finally, in the context of multiagent systems the ACO-environment has the following properties: It is *locally accessible* as any agent can read the pheromone and the heuristic value of each edge that is leaving from the node where it is currently located. The environment is *deterministic* as changing the pheromone weights is the only way agents manipulate the environment. Further it is *not episodic* because the pheromone weights at the edges of the graph are constantly modified and their value has a great influence on the movement of the agents. Moreover, it is also *discrete* as it is in the nature of combinatorial optimization problems to not having an infinite number of solution components. Finally it is not enjoined whether it has to be *static or not*. This depends on the problem instance itself. As ACO is able to react quite well on environmental changes it was already applied on problems of both categories.

3.2. Percepts

Like real ants virtual agents are able to perceive the environment in a very local area around their current position i.e., they are able to read the pheromone and the heuristic value of each edge adjacent to their current node.

3.3. Goals

Each agent tries to assemble a complete and valid solution of the problem by passing a necessary amount of nodes in the graph. Note that the individual agents are not trying to construct high-quality solutions. Cost aspects are only considered in the form of the heuristic value which is most important at the beginning of the search process to provide the agents with an initial orientation. During further development of the search pheromone trails are formed and the importance of the heuristic value decreases. So just like in a real ant colony the optimization capability is an emergent property of the whole group of agents.

3.4. Actions

To achieve its goal of constructing a valid solution every agent is repeatedly performing the following sequence of actions: At first it takes a look at the pheromone and heuristic weight of each edge that is adjacent to its current position. Then based on that value it chooses a certain edge that seems to be promising because either a high pheromone concentration or a good heuristic value moves over it onto a new node and increases the pheromone value of the chosen edge usually by a fixed value. Furthermore, each agent needs an internal memory to store all nodes that have already been visited to prevent the agent from going in circles and to guarantee that a valid solution is built.

3.5. The Virtual Agent Model

The main functionality of a virtual agent is given in Algorithm 1. In each iteration the agent is at first perceiving its surrounding environment via the methods *getAdjacentNodes*, *getPheromoneValues* and *getHeuristicValues*. Then it chooses the next edge based on the sensed values and its internal memory. Note that this is an autonomous step which can be considered as somewhat intelligent. Finally it updates the pheromone weight on the chosen edge and moves to the next node. This procedure is repeated until a complete solution was generated.

After having specified the most important parts, the virtual agent model can be summarized by the PAGE-description (Table 2).

```
procedure VirtualAgent()
  resetMemory(M)
  currentNode = Nest

  while (solution not complete)
    N = getAdjacentNodes(currentNode)
    P = getPheromoneValues(N)
    H = getHeuristicValues(N)
    edge = chooseEdge(P, H, M)

    updatePheromoneConcentration(edge)
    currentNode = getNextNode(edge)
    updateMemory(currentNode)
  end while
end procedure
```

Algorithm 1: Functionality of a virtual agent in pseudo-code.

Percepts	heuristic weights of adjacent edges → pheromone weights of adjacent edges
Actions	choose next edge → increase pheromone weight of chosen edge → move onto next node
Goals	construct complete and valid solution
Environment	graph of solution components

Table 2: PAGE-description of a virtual ant (agent).

4. The Ant Colony Optimization Meta-Heuristic

As the essential characteristics and the main functionality of the virtual agents have been defined in the previous chapter it is now time to combine these agents to a society. Thereby it has to be kept in mind that the main positive effect which leads to the optimization capability of real ant colonies is not present in the considered artificial environment. In the real world the lucky ants that are taking the shorter path arrive earlier at the food source. Consequently more ants walk over the short path in the same time which leads to a higher pheromone concentration and therefore to even more ants choosing that path.

However, when using virtual agents this principle is not applicable as time itself is not considered and the pheromone concentration on a better path is probably as high as the concentration on a worse one. Therefore it is necessary to update the pheromone concentrations manually after every generation of agents has finished its job. When the pheromone weights along the path of the agent that has performed best in the last run are increased by a certain value, it gets more likely that the following generations will also choose at least some of the edges. Consequently the search for good solutions is more and more concentrated in the area of the paths that have already proven their worth and the whole system is converging to a high-quality solution.

But besides this emphasizing of good paths it is also necessary to decrease all pheromone weights after each generation by a small value. In nature this process is called pheromone evaporation and its virtual counterpart is needed to prevent the pheromone values from increasing indefinitely. Furthermore, evaporation can be seen as the collective forgetfulness of the whole colony. It prevents the agents from prematurely getting stuck in a rather bad solution which is one of the most critical problems of neighborhood-based optimization techniques (cf. Simulated Annealing, Evolution Strategies) and at least according to its impact can be compared with the problem of premature convergence in the field of Genetic Algorithms.

So finally with the consideration of these two aspects the instruction sequence of a very generic form of the Ant Colony Optimization meta-heuristic is shown in Algorithm 2. A more detailed description of the ACO and various theoretical and practical aspects are given in [3].

```
procedure AntColonyOptimization()
  initializeEnvironment(E)

  while (termination criterion not met)
    A = createAgents()
    executeAgents(A, E)

    bestPath = getBestAgent(A)
    updatePheromoneConcentration(E, bestPath)
    performPheromoneEvaporation(E)
  end while
end procedure
```

Algorithm 2: The Ant Colony Optimization meta-heuristic in pseudo-code.

5. Conclusion

Agent-based approaches in systems design and software development became very popular in recent years. However, multiagent systems are not really new. If one takes a look at natural processes like bird flocks or fish schools a strong similarity to multiagent systems can be found very quickly.

In this paper, we show how a natural system consisting of many different individuals can be used as an archetype for an artificial multiagent system. As an example natural ant colonies and Ant Colony Optimization as their artificial counterpart were chosen. To point out the immense analogy between natural and artificial multiagent systems the main idea was to describe the whole modelling procedure in the language of multiagent systems and it turned out that this way of describing both systems is quite easy to understand and very suitable.

6. References

[1] DENEUBOURG, J. L., ARON, S., GOSS, S., PASTEELS, J. M., The Self-Organizing Exploratory Pattern of the Argentine Ant, in: Journal of Insect Behaviour, 3:159–168, 1990.

[2] DORIGO, M., Optimization, Learning and Natural Algorithms, Ph.D. Thesis, Politecnico di Milano, Italy, 1992.

[3] DORIGO, M., DI CARO, G., The Ant Colony Optimization Meta-Heuristic, in: D. Crone, M. Dorigo and F. Glover (editors), New Ideas in Optimization, McGraw-Hill, 11-32, 1999.

[4] DORIGO, M., GAMBARDELLA, L. M., Ant Colonies for the Traveling Salesman Problem, in: BioSystems, 43:73–81, 1997.

[5] GOSS, S., ARON, S., DENEUBOURG J. L., PASTEELS, J. M., Self-Organized Shortcuts in the Argentine Ant, in: Naturwissenschaften, 76:579–581, 1989.

[6] JENNINGS, N. R., WOOLDRIDGE, M., Applications of Intelligent Agents, in: N. R. Jennings and M. Wooldridge (editors), Agent Technology: Foundations, Applications, and Markets, Springer-Verlag, Heidelberg, Germany, 1998.

[7] TSUI, K. C., LIU, J., Multiagent Diffusion and Distributed Optimization, accepted to be published in: Proceedings of the Second International Joint Conference on Autonomous Agents & Multiagent Systems, 2003.