Available online at www.sciencedirect.com
SCIENCE DIRECT®
Nuclear Instruments and Methods in Physics Research A 502 (2003) 364–368
NUCLEAR INSTRUMENTS & METHODS IN PHYSICS RESEARCH Section A
www.elsevier.com/locate/nima

Swarm intelligence in optimisation problems

B. Denby[a,*], S. Le Hégarat-Mascle[b]

[a] LISIF/UPMC, 4 Place Jussieu, F-75252 Paris Cedex 05, France
[b] CETP/CNRS, 10 Avenue de l'Europe F-78140 Vélizy, France

Abstract

Swarm intelligence is a name given to a new set of nature-inspired computing paradigms which are being successfully applied to optimisation problems in a variety of fields. The technique is introduced via the application of an algorithm that mimics the behaviour of ants to routing in a satellite-based telecommunications network. The algorithm used is situated within the context of agent-based models, and parallels to other nature-inspired computing techniques are drawn.

PACS: 07.05.Mh; 07.05.Dz; 07.05.Tp; 84.40.Ha

Keywords: Telecommunications; Optimisation; Swarm intelligence; Artificial intelligence; Satellites

1. Introduction

In 1959 entomologist Pierre-Paul Grassé [1] showed that the mound-building behaviour of the termite *Bellicositermes natalensis* could be explained by individual members of the colony following a set of simple rules:

1. Make masticated pulp balls and carry them about.
2. Drop one on a raised, open area if there is one.
3. Otherwise, sniff out an existing pile and affix one on top of that.
4. If the tower gets too high, attach a ball in the direction of the nearest neighbouring tower if there is one; otherwise go elsewhere.

This simple algorithm results in complex nest structures complete with arches and interior passageways. Scientists have found 'intelligent' behaviours in colonies of other 'social' insects such as bees, wasps, and ants as well. The phenomenon has come to be called 'swarm intelligence' (SI), as it seems to associate a higher level of intelligence to the colony than is manifested by its individual members.

Since the early 1990s, a significant amount of work has been done using social insect-inspired algorithms to solve both 'toy' and 'real' problems. There are today yearly international conferences on SI of various types, for example, ANTS'2002—*From Ant Colonies to Artificial Ants: Third International Workshop on Ant Algorithms*, Brussels, 11–14 September 2002. Applications include the travelling salesman problem, quadratic assignment, graph colouring, optimisation, network routing, cluster finding, job scheduling, search

*Corresponding author.
E-mail address: denby@ieee.org (B. Denby).

0168-9002/03/$ - see front matter © 2003 Elsevier Science B.V. All rights reserved.
doi:10.1016/S0168-9002(03)00444-3

engines, and load balancing [2–7]. SI thus seems to adapt well to problems involving combinatorial complexity.

Ant colony optimisation (ACO) [8] is currently a popular algorithm. The classic application is to routing in telecommunication networks, on which a number of articles have appeared [9–13]. The nature-inspired basis of the technique is as follows. While searching for food, ants deposit trails of a chemical substance called pheromones to which other ants are attracted. As shorter paths to food will be traversed more quickly, they have a better chance of being sought out and reinforced by other ants before the volatile pheromones evaporate. Other ants then follow suit, and so on. Thus using pheromones and random search procedures, the colony is able to rapidly find the shortest paths to food.

In what follows, an application of ACO to routing in a satellite telecommunications network is presented [14]. The simulation strives to be realistic, containing 72 LEO satellites and 121 ground stations, with results compared to those obtained with the more standard link state technique SPF.

2. ACO applied to routing in a satellite network

2.1. The network model

The satellite network model contains 72 LEO satellites equally distributed among nine orbits of radius 1603 km and 50° equatorial inclination, with a minimum elevation of 17.5°. The orbital period is 118.5 min and satellite footprints are 5100 km in diameter. Each satellite is equipped with 155 Mbit/s uplink and downlink transceivers, and four bi-directional intersatellite links (ISL) also of 155.5 Mbit/s, each with its own queue, enabling it to communicate with its two nearest inter- and intra-orbit neighbours. This creates a uniform, space-based network to which earth stations can connect for multimedia applications. In the terrestrial sector, the earth's surface is divided into a grid of 12×24 cells, each containing a single-gateway station which handles all the traffic of the cell (Fig. 1).

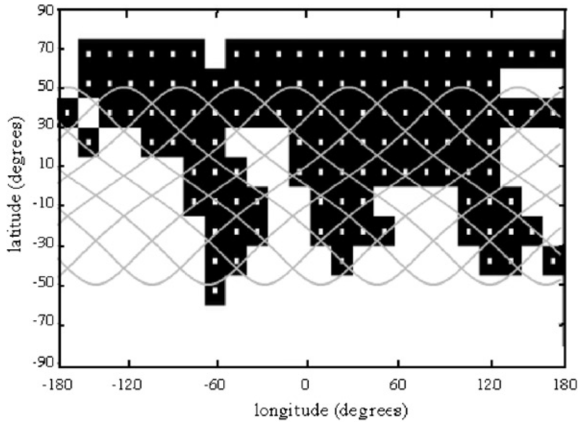

Fig. 1. Mercator view of earth's surface on a 12×24 grid. Land cover is constrained to lie within gridlines and a gateway station assigned to each populated square. Fixed-earth projections of the satellite orbital paths are also shown.

2.2. The traffic model

Traffic was considered to consist of both circuit-switched voice calls and packet mode data transmissions. For voice traffic, each gateway was assigned a traffic level in calls/s based on population and industrialisation figures projected to the year 2005 [15]. With a fixed 64 kbit/s rate for each call, and call duration selected from a decreasing exponential distribution of mean 3 min, voice traffic amounts on average to 35 000 calls in the system (35 kErlangs), or about 2.24 Gbit/s system voice load. For data traffic, the number of sessions was based on values from Ref. [15], and system data load obtained as the product of the number of open data sessions times the number of 40 byte packets to be sent per session. In all simulations presented here, the number of data sessions per hour per gateway was 100 000.

Although experiments with differing amounts of data traffic were run, only a representative result, called 'normal', is presented here. To show the behaviour of the system for bursty data, an additional experiment called 'bursty' is also presented, in which 50% of session sizes were multiplied by 10 while retaining an overall traffic level of 'normal'. Table 1 gives the parameters for the two experiments.

Table 1
Mean values of traffic parameters for representative simulation scenarios

Traffic model	Packets per data session	% of 10 × augmented data sessions	System data load (Gbit/s)	System total load (Gbit/s)
Normal	2000	—	2.15	4.39
Bursty	2000	50	2.15	4.39

In all experiments, the total calls bandwidth was 2.24 Gbit/s, and the number of data sessions per hour per gateway is 100 000. The ACO and SPF routing algorithms add 230.4 or 408 kbit/s, respectively, of routing bandwidth to the values shown.

2.3. The ACO algorithm

The version of ACO used was adapted from the one in Ref. [10] and functions as follows:

- Once every 100 ms, each satellite node emits an ant with a random destination.
- The ant follows the routing tables to the destination, except for a 1% 'exploration' probability, waiting in queues and memorising trip times en route.
- When the destination is reached, it follows the same path back, jumping all queues, and updating routing tables along the way.

The routing tables are updated according to the following algorithm:

- First calculate $r = \min[T/\bar{T}; 1]$ where T is the current ant trip time and $\bar{T}$ is the mean time for the path in question.
- Next, modify the probability of the link that is part of the ant's path according to

  $$P_{\text{ant ISL}} = P_{\text{ant ISL}} + (1 - r)(1 - P_{\text{ant ISL}}), \tag{1}$$

  and decrement the other three ISLs as

  $$P_{\text{ISL}(i)} = P_{\text{ISL}(i)} - (1 - r)P_{\text{ISL}(i)}. \tag{2}$$

Two generic improvements to this 'baseline' model have been cited in the literature: (1) replacing r by the so-called 'squashed' value r^s (s here was chosen to be 0.2); (2) using the 'fuzzy' routing technique of the ant packets for normal data packets as well. All results presented use 'squashed'/'fuzzy' ACO.

2.4. Dijkstra and SPF algorithms for comparison

The Dijkstra algorithm finds the absolute shortest path according to a cost function involving propagation delays and queue lengths. It assumes global, instantaneous knowledge and is not realisable. Our version of Dijkstra ignored queue lengths and thus corresponds to a true absolute minimum (though unrealisable) delay, i.e., propagation delay only.

In our version of the well-known Shortest Path First (SPF) algorithm, each satellite sends a list of its queue lengths to every node in the network once per second. The receiving node then updates its routing table based on this delayed information, using Dijkstra shortest path with a cost function

$$\text{cost} = t_{\text{propagation}} + 0.6t_{\text{queue}} + 0.4\bar{t}_{\text{queue}}. \tag{3}$$

The SPF update rate chosen gives an average routing bandwidth of about 408 kbit/s, i.e., roughly twice that of ACO (230.4 kbit/s).

2.5. Results

In Fig. 2, ACO delays are compared to those of infinite bandwidth Dijkstra for the cases of 'normal' and 'bursty' traffic conditions. As ACO results lie near those of Dijkstra, we may conclude that ACO routing gives near-optimal packet delay distributions. Furthermore, ACO mean packet delays are tens to hundreds of milliseconds lower than those of SPF over a range of traffic conditions. Meanwhile, the additional routing bandwidth introduced by ACO is negligible compared to the system load of several Gbit/s, and only about half that needed by SPF.

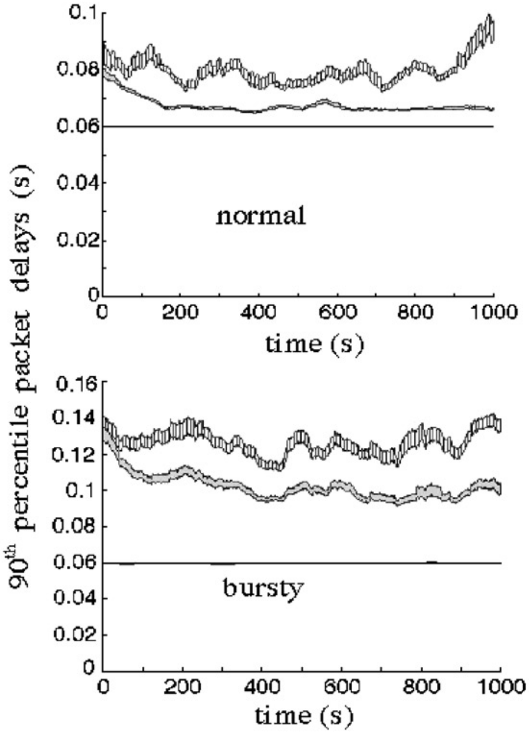

Fig. 2. Ninetieth percentile packet delays (in s) versus time (in s) for SPF (stripes), 'squashed'/'fuzzy' variant of ACO (gray), and infinite bandwidth Dijkstra (black) for 'normal' and 'bursty' traffic conditions.

3. Comparing ACO to other nature-inspired algorithms

A number of other modern optimisation and/or computing techniques are modelled upon natural phenomena. In simulated annealing, an analogy is made between the thermodynamic behaviour of solids and the solution of large combinatorial optimisation problems. Points in solution space are attributed values of an 'energy' function whose probabilities follow a Boltzmann distribution in a fictitious temperature. The act of choosing the next step in the optimisation probabilistically rather than deterministically introduces randomness which allows to escape from local minima and find globally optimal solutions. In genetic algorithms, problem solutions are mapped onto individuals in a pool. Goodness of solutions, determined by comparison to a fitness function, can be shared between individuals via genomes which can undergo crossover between individuals or mutation. In addition to randomness, there is also a notion of sociality in such algorithms, as well as one of self-organisation, in the sense that the steps taken in obtaining the solutions are not pre-programmed but rather determined stochastically during the running of the program. Neural networks are a well-known computer learning and classification technique in which distributed computing plays a role, i.e., the knowledge associated with a trained neural network is not stored in any specific location, but encoded in a distributed way across its weight matrix. Especially with regard to genetic algorithms, recurrent neural networks, and cellular automata, one also refers to the complexity exhibited in certain types of final states as being an emergent property, i.e., one not present in the individual elements of the systems, but rather emerging somehow out of the ensemble.

ACO shares many of the properties of these other nature-inspired algorithms. Randomness is present in the 'fuzzy' routing used by the ants. Sociality and self-organisation are intrinsic to the method. ACO-trained networks are distributed in the sense that overall knowledge of the network status exists nowhere. Such knowledge nonetheless certainly emerges from the system, however, as it is capable of producing Dijkstra-quality solutions.

4. Agent-based computing

Agent-based computing techniques have become very popular in recent years. An autonomous agent can be defined as a system that senses its environment and acts on it over time in pursuit of an agenda, doing so in a way that can affect its future behaviour. They are usually considered to be reactive, i.e., they do not contain internal symbolic models, but simply react to the current state of the environment. Agents are different from traditional computer programs in that they are autonomous and adaptable to changes in the environment. In multi-agent systems, they may interact with other agents in the system as well as the end-users of the system, and may co-operate with other agents to carry out more complex tasks than they themselves can handle. Mobile agents are understood to be able to move from one system to another in order to access remote resources or meet other agents. We may conclude that 'ants' are reactive, mobile, autonomous multi-agent systems. Agent-based systems in fact have much in common with nature-inspired algorithms.

5. Conclusions

Nature-inspired algorithms are useful because they are based upon well-known models. The underlying physics of such models can act as a guide on how to structure algorithms, and may inspire the confidence to try them over other types of algorithms. Random selection, self-organisation, distributed computation, and emergent, shall we say 'swarm', intelligence are all attractive features of these algorithms. We have also seen that agent-based methods have many similarities to nature-inspired algorithms. ACO is a simple SI algorithm which gives near-optimal results in a realistic telecommunications network simulation, and performs better than the traditional SPF algorithm while using less routing bandwidth.

References

[1] P.-P. Grassé, Insectes Sociaux 6 (1959) 41.
[2] V. Maniezzo, A. Colorni, M. Dorigo, The ant system applied to the quadratic assignment problem, Technical Report IRIDIA/94-28, Univ. Libre de Bruxelles, 1994.
[3] A. Colorni, M. Dorigo, F. Maffioli, V. Maniezzo, G. Righini, M. Trubian, Int. Trans. Oper. Res. 3 (1996) 1.
[4] D. Costa, A. Hertz, J. Oper. Res. Soc. 48 (1997) 295.
[5] M. Dorigo, L.M. Gambardella, IEEE Trans. Evol. Comput. 1 (1997) 53.
[6] M. Dorigo, G. Di Caro, L.M. Gambardella, Artif. Life 5 (1999) 137.
[7] L.M. Gambardella, E. Taillard, M. Dorigo, J. Oper. Res. Soc. 50 (1999) 167.
[8] M. Dorigo, V. Maniezzo, A. Colorni, IEEE Trans. Systems, Man Cybern—Part B 26 (1996) 29.
[9] R. Schoonderwoerd, O. Holland, J. Bruten, Proceedings of the Agents'97, Marina del Rey, CA, USA, ACM, New York, 1997, p. 209.
[10] G. Di Caro, M. Dorigo, AntNet: a mobile agents approach to adaptive routing, Technical Report IRIDIA97-12, Univ. Libre de Bruxelles, 1997.
[11] G. Di Caro, M. Dorigo, Proceedings of the 31st Hawaii International Conference on Systems Sciences HICSS-31, Hawaii, January 1998.
[12] G. Di Caro, M. Dorigo, J. Artif. Intell. Res. 9 (1998) 317.
[13] E. Bonabeau, F. Heneaux, S. Guèrin, D. Snyers, P. Kuntz, G. Theraulaz, Routing in telecommunications networks with smart ant-like agents, Santa Fe Institute Working Paper 98-01-003, 1998.
[14] E. Sigel, B. Denby, S. Le Hégarat, Ann. Telecommun. 57 (2002) 520.
[15] M. Werner, G. Maral, Proceedings of the International Mobile Satellite Conference, IMSC'97, Pasadena, June 1997, p. 283.