Ant-based load balancing in telecommunications networks

Ruud Schoonderwoerd[1,2], Owen Holland[3],
Janet Bruten[1] and Leon Rothkrantz[2]

[1]Hewlett-Packard Laboratories Bristol,
Filton Road, Stoke Gifford, Bristol BS12 6QZ, United Kingdom

[2]Delft University of Technology,
Julianalaan 132, 2628 BL Delft, The Netherlands

[3]University of the West of England
Coldharbour Lane, Bristol BS16 1QY, United Kingdom

ruud@hplb.hpl.hp.com o-hollan@csd.uwe.ac.uk
jlb@hplb.hpl.hp.com roth@kgs.twi.tudelft.nl

Abstract

This paper describes a novel method of achieving load balancing in telecommunications networks. A simulated network models a typical distribution of calls between nodes; nodes carrying an excess of traffic can become congested, causing calls to be lost. In addition to calls, the network also supports a population of simple mobile agents with behaviours modelled on the trail laying abilities of ants. The ants move across the network between randomly chosen pairs of nodes; as they move they deposit simulated pheromones as a function of their distance from their source node, and the congestion encountered on their journey. They select their path at each intermediate node according the distribution of simulated pheromones at each node. Calls between nodes are routed as a function of the pheromone distributions at each intermediate node. The performance of the network is measured by the proportion of calls which are lost. The results of using the ant-based control (ABC) are compared with those achieved by using fixed shortest-path routes, and also by using an alternative algorithmically-based type of mobile agent previously proposed for use in network management. The ABC system is shown to result in fewer call failures than the other methods, while exhibiting many attractive features of distributed control.

1 Introduction

The notion of complex collective behaviour emerging from the behaviour of many relatively simple units, and the interactions between them, is fundamental to the field of artificial life. The growing understanding of such systems offers the prospect of creating artificial systems which are controlled by such emergent collective behaviour; in particular, we believe that the exploitation of this concept might lead to completely new approaches for the management of distributed systems, such as load balancing in telecommunications networks.

What is load balancing? For economic and commercial reasons, such networks are equipped not with a level of equipment which will guarantee successful call connection under all possible circumstances, but with some lower level which will give acceptable performance under most conditions of use. If there is some significant change in the conditions - for example, if the total call volume at any time is unusually high, or if some particular location is suddenly the origin or destination of an unusually large volume of calls - then these capacity limitations might lead to the system failing with calls unable to be connected.

Calls between two points are typically routed through a number of intermediate switching stations, or nodes; in a large network, there are many possible routes for each such call. It is thus possible to relieve actual or potential local congestion by routing calls via parts of the network which have spare capacity. Load balancing is essentially the construction of call-routing schemes which successfully distribute the changing load over the system and minimise lost calls. Of course it is possible to determine the shortest routes from every node to every other node of the network. In this way the average utilisation of nodes will be minimised, but this is not necessarily the ideal way to avoid node congestion, as this has to do with how the traffic on the network is distributed.

Controlling distributed systems like these by means of a single central controller has several disadvantages. The controller usually needs current knowledge about the entire system, necessitating communication links from every part of the system to the controller. These central control mechanisms scale badly, due to the rapid increase of processing and communication overheads with system size. Failure of the controller will often lead to failure of the complete system. There is the additional practical commercial requirement that centrally controlled systems may need to be owned by one single authority. Further, the nature of distributed systems like these is highly dynamic, complex and stochastic, and their behaviour can neither be predicted nor explained by reducing it to a single central controllable factor.

A good decentralized control mechanism will not have the problems mentioned above. The field of artificial life has given us inspiration for such a mechanism that will be completely distributed, and highly adaptive to changes in the network and traffic patterns. This solution makes use of the distributed processing capability already inherently present in the network in the form of network nodes. The distributed nature of such an approach may make the system very robust against failures of individual control entities.

Our approach is inspired by the work of biologists studying social insects, who have uncovered the mechanisms controlling the foraging behaviours of ants (Beckers, Deneubourg & Goss, 1992; Deneubourg & Goss, 1989; Goss et.al., 1990; Franks, 1989). The most important method is the laying and sensing of trails of pheromones - specialised chemical substances which are laid in amounts determined by local circumstances, and which by their local concentration subsequently directly influence an ant's choice of route.

In this paper we present a network model which is populated by artificial ants that make use

of the trail laying principle; at each node an ant encounters on the journey to its destination, it leaves an amount of simulated pheromone which is a function of the congestion of the node, and of the distance the ant has travelled from its source node; the ant then selects the next node on its journey on the basis of the local pheromone distribution. The routing of calls is then based on these pheromone distributions.

We compare this method with another decentralised network control mechanism, which is based on previous work carried out by British Telecom (Appleby & Steward, 1994). It makes use of mobile agents - computational processes moving from node to node, gathering information, and making decisions for rerouting on the basis of a minimum cost function.

Both approaches are compared to an approach which uses fixed, non-adaptive routing tables algorithmically optimised to yield the shortest paths.

2 A Network Simulation to investigate distributed control mechanisms

An application has been written to simulate traffic patterns on a model of a switch-based telecommunications network. Such a network is most naturally represented by an undirected graph. Each node in the graph corresponds to a switching station; the links between nodes correspond to communication channels. A given node will usually only be linked to a subset of other nodes, usually its geographical neighbours; links are intrinsically bidirectional. The network model used is a graph of n nodes, each of which has several attributes:

- A node identifier.
- A capacity. This is the number of simultaneous calls that the node can handle.
- A routing table with (n-1) entries, one for each node in the network. Each entry tells us which node is the next node on the route to the destination node concerned.
- A probability of being the end node (either source or destination) of a call.
- A spare capacity. This is the percentage of the capacity that is still available on the node.

The links between the nodes are assumed to have infinite capacity, so that the geographical distance between linked nodes is of no account. The node capacities will therefore be the only bottlenecks in the network.

2.1 The simulation

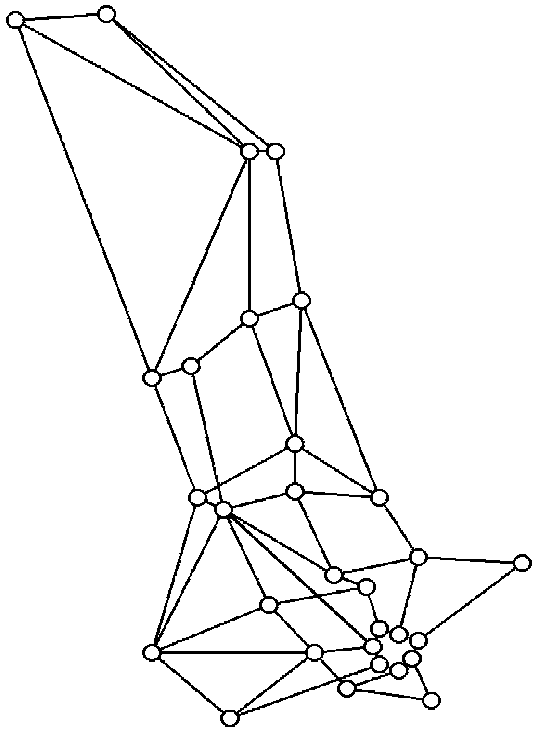
Every time step of the simulation proceeds as follows. First, calls that have expired are removed, releasing capacity at the nodes. Next, calls are generated by a traffic generator. These generated calls involve a source node, a destination node and a duration, measured in time steps. When a call is generated, its route is determined by the current routing tables, and the call is placed on the network, reducing the spare capacity of each node on the route. If there is no spare capacity available on a node on the route of the call, this call will fail. The expected number of calls generated in one time step follows a Poisson distribution, and the duration of each call an exponential distribution. The source and destination nodes are randomly chosen as a function of each node's probability of being an end node; this is both convenient and reasonable.

2.2 Test details

The 30-node network topology of Figure 1 was chosen because this is a realistic intercon-

nection structure of a possible switch-based network. It is also the same topology as was used in (Appleby & Steward, 1994), and is in fact the structure of the British Synchronous Digital Hierarchy (SDH) network. The interconnection structure is an irregular mesh that is interesting because it makes traffic management a complex and difficult task.

FIGURE 1. This network topology is the same as the interconnection structure of the SDH network of British Telecom and provides a realistic network topology.



The number of parameters that can vary in this network model is large. This implies that some arbitrary choices have to be made for testing:

- When a call is set up, each node in the model of Figure 1 has a certain probability of being an end point of the call. These probabilities are generated by selection from a suitable distribution at the start of every run, and lie between 0.01 and 0.07. After generation these probabilities are normalised to sum to 1.
- The capacity of each node is 40 calls, so that every call using a node increases the utilisation (or decreases the spare capacity) of that node by 2.5%.
- During every time step of the simulation, an average of 1 call is generated with an average duration of 170 time steps. This means that the average number of calls on the network will be 170, once the traffic pattern has built up.
- The average length (total number of nodes) of the shortest route between two nodes is 4.07. With fixed shortest-path routing tables, each of the 170 calls will use 2.5% of the capacity of an average of 4.07 nodes. As there are 30 (number of nodes) times 40 = 1200 'capacity units', the average utilisation of the nodes will be 170 x 4.07 / 1200 = 57.7%.

To facilitate seeing exactly what is going on in the network simulation, the traffic in the network is represented visually during run-time by changing the colour of the nodes to indicate their spare capacities. A large number of features to support visualisation and usability have been added to the program, such as displaying the changes with time of one particular route, running the simulation step-by-step, and inspecting every part of the network during the simu-

lation. The software was written in Smalltalk, running in the VisualWorks™ [1] environment on a Hewlett-Packard 9000 Series 700 workstation.

3 Ants in nature

Individual ants are behaviourally very unsophisticated insects. They have a very limited memory and exhibit individual behaviour that appears to have a large random component. Acting as a collective however, ants manage to perform a variety of complicated tasks with great reliability and consistency. A few examples of collective behaviour that have been observed in several species of ants are (Hölldobler & Wilson, 1994; Franks, 1989):

- regulating nest temperature within limits of 1ºC;
- forming bridges;
- raiding particular areas for food;
- building and protecting their nest;
- sorting brood and food items;
- cooperating in carrying large items;
- emigration of a colony;
- complex patterns of egg and brood care;
- finding the shortest routes from the nest to a food source;
- preferentially exploiting the richest available food source.

These behaviours emerge from the interactions between large numbers of individual ants and their environment. In many cases, the principle of stigmergy (Grassé, 1959) is used. Stigmergy is a form of indirect communication through the environment. Like other insects, ants typically produce specific actions in response to specific local environmental stimuli, rather than as part of the execution of some central plan. If an ant's action changes the local environment in a way that affects one of these specific stimuli, this will influence the subsequent actions of ants at that location. The environmental change may take either of two distinct forms. In the first, the physical characteristics may be changed as a result of carrying out some task-related action, such as digging a hole, or adding a ball of mud to a growing structure. The subsequent perception of the changed environment may cause the next ant to enlarge the hole, or deposit its ball of mud on top of the previous ball. In this type of stigmergy, the cumulative effects of these local task-related changes can guide the growth of a complex structure. This type of influence has been called sematectonic (Wilson, 1975). In the second form, the environment is changed by depositing something which makes no direct contribution to the task, but is used solely to influence subsequent behaviour which is task related. This sign-based stigmergy has been highly developed by ants and other exclusively social insects, which use a variety of highly specific volatile hormones, or pheromones, to provide a sophisticated signalling system. Some of the above behaviours have been successfully simulated with computer models, using both sematectonic stigmergy (Theraulaz & Bonabeau, 1995), and sign-based stigmergy (Stickland, Tofts & Franks, 1992) and also on robots (Beckers, Holland & Deneubourg, 1994; Russell 1995; Deveza et.al., 1994).

A type of sign-based stigmergy is used in our network model. It is based on the way ants find short routes from their nest to a food source, and also on the way they select between food

1. VisualWorks is a trademark of ParcPlace Systems, Inc.

sources of different value. The way ants organise these routes has inspired us to investigate a new approach for congestion avoidance in telecommunications networks.

3.1 Basic principles of trail laying

Depending on the species, ants may lay pheromone trails when travelling from the nest to food, or from food to the nest, or when travelling in either direction. They also follow these trails with a fidelity which is a function of the trail strength, among other variables. Ants drop pheromones as they walk by stopping briefly and touching their gaster, which carries the pheromone secreting gland, on the ground. The strength of the trail they lay is a function of the rate at which they make deposits, and the amount per deposit. Since pheromones evaporate and diffuse away, the strength of the trail when it is encountered by another ant is a function of the original strength, and the time since the trail was laid. Most trails consist of several superimposed trails from many different ants, which may have been laid at different times; it is the composite trail strength which is sensed by the ants. The principles applied by ants in their search for food are best explained by an example as given in (Beckers, 1992):

FIGURE 2. Ants have a decision to make

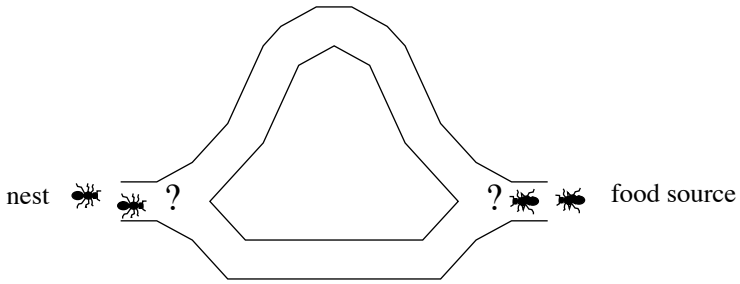
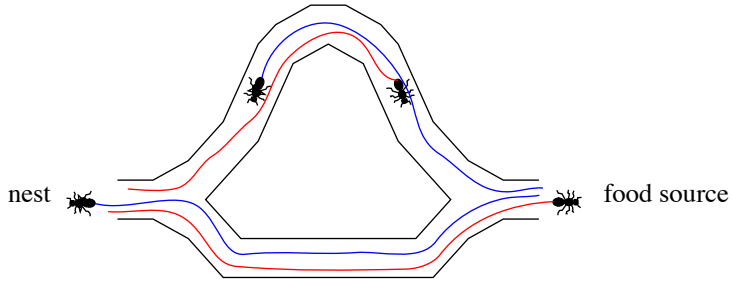


Figure 2 illustrates two possible routes between nest and food-source. Initially, an ant arriving at a T-crossing (choice point), makes a random decision with a probability of 0.5 of turning left or right. Now suppose there are two ants leaving the nest, looking for food, and two ants returning from the food source to the nest. Let the ants be of a type such as *Lasius Niger* which deposits pheromones when travelling both to and from the nest. If one ant from each pair turns left, and the other turns right, after a while a situation occurs like that in Figure 3. The lines on the paths represent the pheromone trails. The ants that chose the shorter branch have arrived at their destination, while the ones that chose the longer branch are still on their way. Ants initially select their way with a 0.5 probability for both branches, as there is no pheromone on the paths yet. If there is pheromone present, there is a higher probability of an ant choosing the path with the higher pheromone concentration, i.e. the path where more ants have travelled recently. If at the moment of the situation in Figure 3 other ants arrive and have to choose between the two paths, they are more likely to choose the shorter path, because that is where the concentration of pheromone is higher. This means that the amount of pheromone on the shorter path is more likely to be reinforced again. In this way, a stronger pheromone trail will arise on the shorter path, and so the path will be selected by an increasing proportion of ants As fewer ants choose the longer path, and the existing pheromone slowly evaporates, the trail on the longer path will weaken and eventually disappear.

FIGURE 3. Situation several moments later



Although this is essentially self-organisation rather than learning, ants have to cope with a phenomenon that looks very much like overtraining in reinforcement learning techniques. There are two main issues: the blocking problem and the shortcut problem (Sutton, 1990). The blocking problem occurs when a route previously found by the ants is no longer available. It can then take a relatively long time for the ants to find a new route. The shortcut problem occurs when a new, shorter route suddenly becomes available. In this case the new route will not easily be found, because the old trails are so strong that almost all the ants choose them.

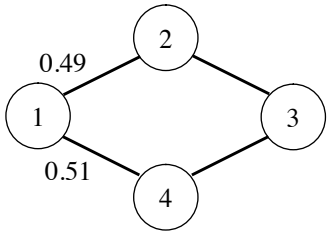
4 Ant-Based Control (ABC) for network management

How could this trail laying and following behaviour be applied to something like a telecommunications network? And can we overcome the blocking problem and the shortcut problem? This section describes how we implemented an artificial ant population on the network model. Further details may be found in (Schoonderwoerd, 1996).

4.1 Pheromone tables

We replaced the routing tables in the network nodes by tables of probabilities, which we will call 'pheromone tables', as the pheromone strengths are represented by these probabilities. Every node has a pheromone table for every possible destination in the network, and each table has an entry for every neighbour. For example, a node with four neighbours in a 30-node network has 29 pheromone tables with four entries each. One could say that an n-node network uses n different kinds of pheromones. The entries in the tables are the probabilities which influence the ants' selection of the next node on the way to their destination node. Figure 4 shows a possible network configuration and a pheromone table. For example, ants travelling from node 1 to node 3 have a 0.49 probability of choosing node 2 as their next node, and 0.51 of choosing node 4. 'Pheromone laying' is represented by 'updating probabilities'.

FIGURE 4. Using ants for network management



The three pheromone tables for node 1:

		next node	
		2	4
destination node	2	0.95	0.05
	3	0.49	0.51
	4	0.05	0.95

At every time step during the simulation, ants can be launched from any node in the network. Each ant has a random destination node. Ants move from node to node, selecting the next node to move to according to the probabilities in the pheromone tables for their destination node. Arriving at a node, they update the probabilities of that node's pheromone table entries corresponding to their *source* node i.e. ants lay the kind of pheromone associated with the node they were launched from. They alter the table to increase the probability pointing to their previous node. When ants have reached their destination, they die.

Take as an example an ant in the network of Figure 4 that is launched at node 3 with destination node 2, and has just travelled from node 4 to node 1. This ant will first alter node 1's table corresponding to node 3 (its source node) by increasing the probability of selection of node 4; it will then select its next node randomly according to the probabilities in the table corresponding to its destination node, node 2. (Note that just for the purpose of illustration this particular ant travelled an ineffective route.)

In this way, ants moving away from their source node can only directly affect those ants for which it is the destination node. This is unlike the trails of bidirectional trail laying ants, in which a trail laid in one direction can directly affect ants travelling in either direction. However, the ants which can be directly influenced by an ant travelling from a source node S to a destination node D will include those travelling from D to S; these are the very ants which could be expected to have most influence on ants travelling from S to D, and so ants travelling from S to D may have a strong influence on ants subsequently travelling that route via their effect on the ants travelling on the opposite route. The system may thus achieve similar effects to the biological bidirectional trail layers, but through an indirect form of interaction.

This way of directly updating probabilities differs from the way ants lay pheromones, but is functionally equivalent. We feel that using probabilities instead of absolute pheromone quantities helps us to understand the behaviour of the artificial ants better. The tables we use give the probabilities of alternative choices between paths directly, whereas the pheromones of real ants are basically a code that is effectively converted into probabilities by the ant's nervous system.

The method used to update the probabilities is quite simple: when an ant arrives at a node, the entry in the pheromone table corresponding to the node from which the ant has just come is increased according to the formula:

$$p = \frac{p_{old} + \Delta p}{1 + \Delta p}$$

Here p is the new probability and Δp is the probability (or pheromone) increase. The other entries in the table of this node are decreased according to:

$$p = \frac{p_{old}}{1 + \Delta p}$$

Since the new values sum to 1, they can again be interpreted as probabilities. Note that a probability can be reduced only by the operation of normalisation following an increase in another cell in the table; since the reduction is achieved by multiplying by a factor less than one, the probability can approach zero if the other cell or cells are increased many times, but will never reach it. For a given value of Δp the absolute and relative increase in probability is much greater for initially small probabilities than for those which are larger. This has the effect of weighting information from ants coming from nodes which are not on the currently preferred route, a feature which may assist in the rapid solution of the shortcut problem.

4.2 Ageing and delaying ants

A primary requirement of this work was to find some simple methods of encouraging the ants to find routes which are relatively short, yet which avoid nodes which are heavily congested. Two methods are used. The first is to make Δp, the value used to change the pheromone tables, reduce progressively with the age of the ant. When the ant moves at one node per time step, the age of the ant corresponds to the path length it has traced; this biases the system to respond more strongly to those ants which have moved along shorter trails. The second method, which depends on the first, is to delay ants at nodes that are congested with calls to a degree which increases with the degree of congestion. This delay has two complementary effects:

- it temporarily reduces the flow rate of ants from the congested node to its neighbours, thereby preventing those ants from affecting the pheromone tables which are routing ants to the congested node, and allowing the probabilities for alternative choices to increase rapidly.
- since the ants are older than they otherwise would have been when they finally reach the neighbouring nodes, they have less effect on the pheromone tables.

It is of course possible to achieve the second effect alone by increasing the parameter representing the age of the ant without actually delaying the ant; this essentially reduces the effect on pheromone tables of an ant which has passed through a congested node. This has a biological parallel: in some species of ants, those returning from a richer food source tend to drop more pheromone than those from a poor source (Beckers, 1993). In the case of our network simulation, however, the combination of delay and age-related penalty seems to be particularly effective. An added advantage of this formulation is that the manipulation of the parameter relating delay to the degree of congestion can be used to control the relative weighting which the system gives to preferring the shortest route (which maximises spare capacity), as against preferring the least congested route.

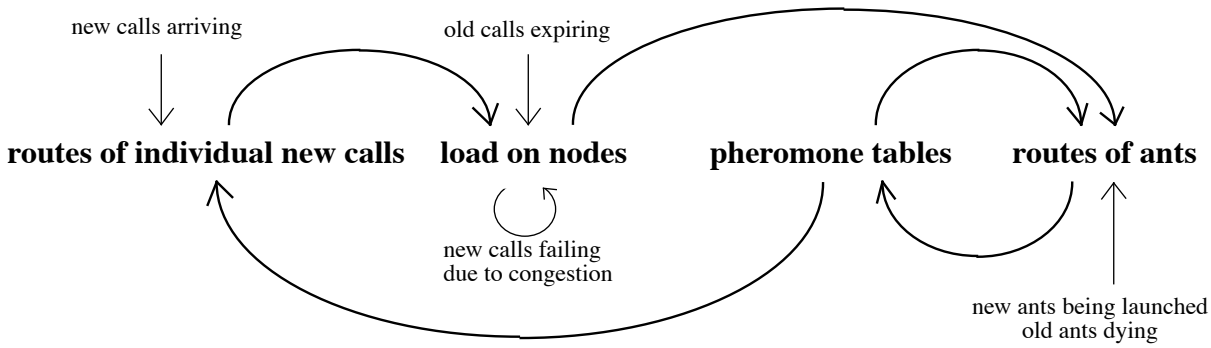
4.3 How calls are routed

Having explained how ants ‘choose’ their routes through the network, let us consider the calls. Calls operate independently of the ants. To determine the route for a call from a particu-

lar node to a destination, the largest probability in the pheromone table for this destination is looked up. The neighbour node corresponding to this probability will be the next node on the route to this destination. The route is valid if the destination is reached, and the call is then placed on the network, unless one of the nodes on the route is congested; in that case the call fails to be placed on the network.

In this way, calls and ants dynamically interact with each other. Newly arriving calls influence the load on nodes, which will influence the ants by means of the delay mechanism. Ants influence the routes represented by the pheromone tables, which in their turn determine the routing of new calls. These relationships are illustrated in Figure 5. One needs to realise that the pheromone table by which an individual ant is influenced, is a different table than the pheromone table that will be updated by this ant. The load on the network at any given time influences which calls can subsequently be placed on the network and which calls will fail; which of course determines the load at a later stage.

FIGURE 5. Relationship between calls, node utilisation, pheromone tables and ants. An arrow indicates the direction of influence



4.4 Initialisation

A network initialised with random or uniform entries in the pheromone tables will not initially contain any useful information about consistent (i.e. non-circular) call routes, let alone good routes. It therefore makes little sense to examine network performance during this phase. However, even in the absence of calls on the network, the ants will bias towards shortest paths. There are three mechanisms contributing to this:

- The shortest routes will be completed first, and will subsequently direct other ants to their sourcing nodes first.
- Shorter routes involve less branching, so the number of ants travelling over these routes and laying pheromones will be larger than on longer and more branched routes.
- Ants travelling over shorter routes will be younger when they arrive, and will therefore exercise more influence on the pheromone tables.

After a short time the highest probabilities in the pheromone tables of each node will define relatively short routes, and circular routes will have been eliminated (for example after 500 time steps the average route length is typically 4.10, where 4.07 is the minimum possible), and when the routes are sufficiently short, calls can safely be put on the network; subsequent adaptation will then be influenced by any congestion caused by calls. All ant networks were therefore initialised with equal probabilities for neighbour nodes in each pheromone table, and allowed to run for a fixed period before calls were applied.

4.5 Noise

So far, we have not considered the blocking problem and the shortcut problem. We need to avoid 'freezing' of the routes in situations that remain static for a long time and then suddenly change. One way of doing this is by adding an exploration probability, or noise, to the random walk of the ants; this will ensure that even apparently useless routes are used occasionally, so that at least some information about them is present in the system to give a head start when a route is blocked (e.g. by extreme node congestion or node failure). Noise might also encourage the more rapid discovery of a better route which suddenly appears.

A convenient implementation is to arrange that a noise factor of *f* means that at every time step an ant has probability *f* of choosing a purely random path, and probability (1-*f*) of choosing its path according to the pheromone tables on the nodes. The possibly beneficial effects of the addition of noise to ant-based algorithms were noted in (Deneubourg, 1990): 'Rather than simply tolerating a certain degree of error, it can even be desirable to deliberately add error where none or little exists.'

4.6 General framework for ant-based control systems

The basic principles for ant-based control (ABC) systems are applicable to a wide variety of problems, and can be characterised as follows:

- Ants are regularly launched with random destinations on every part of the system.
- Ants walk randomly according to probabilities in pheromone tables for their particular destination.
- Ants update the probabilities in the pheromone table for the location they were launched from, by increasing the probability of selection of their previous location by subsequent ants.
- The increase in these probabilities is a decreasing function of the age of the ant, and of the original probability.
- This probability increase could also be a function of penalties or rewards the ant has gathered on its way.
- The ants get delayed on parts of the system that are heavily used.
- The ants could eventually be penalised or rewarded as a function of local system utilisation.
- To avoid overtraining through freezing of pheromone trails, some noise can be added to the behaviour of the ants.

4.7 Parameters

There is a large number of parameters to tune for this system - choices are based on experience with a variety of previous simulations:

- The speed of the ants is one node per simulation time step (unless they are delayed on a particular node).
- We chose to let every node launch an ant with a random destination on every time step of the simulation.
- The probabilities are updated as explained in Section 4.1, and according to the following formula, where *age* stands for the number of time steps that passed since the launch of the ant:

$$\Delta p = \frac{0.08}{age} + 0.005$$

- The *delay* in time steps that is given to the ant is a function of the spare capacity s of the node:

$$delay = \lfloor 80 \cdot e^{-0.075 \cdot s} \rfloor$$

- The initialisation period, that is the period during which the ants initialise the routes on the network without traffic, is between 250 (no noise) and 500 (5% noise) time steps.

Before we give some results of our simulations, we will describe briefly another distributed network management approach based on work by researchers from British Telecom. We compared the ABC method with this method and with a fixed routing scheme.

5 Mobile Software Agents

A different approach for a distributed control mechanism is provided by the mobile agents developed by researchers from British Telecom in (Appleby & Steward 1994). In addition, these agents are an example of a routing scheme that optimises routes in the network according to a least cost criterion. We implemented two modified versions of their scheme on our network simulation model. Further details may be found in (Schoonderwoerd, 1996).

In this mobile agents approach, there are two 'species' of agents: load management agents and parent agents. The lowest level of control is provided by the load management agent. Each such agent is launched from a particular node, and then searches algorithmically for better routes from nodes in the network to the node where it was launched. Parent agents provide the second level of control. According to heuristics and information gathered on the network, a parent agent can decide that network management at certain locations is needed to relieve congestion, and therefore launches load agents at those locations.

5.1 The Load management agent

A load agent is launched on a particular node and optimises the routes from all other nodes to that source node. It does this by visiting every node in the network, recording the current spare capacity, and amending the routing tables. We investigated two methods, each with a different kind of load agent. In the first, we made load agents find the route with a minimum bottleneck i.e. they maximise the minimum spare capacity on the route, as in the BT work. In the second we implemented a version in which they minimise the sum of squared node utilisations. The reason for this will become clear later.

The algorithm that is used to find the new routes is a version of Dijkstra's shortest path algorithm (Dijkstra, 1959). As a criterion for 'shortest path' the minimum spare capacity on the route is used in the original algorithm; and the sum of squared node utilisations for our version of the load agent.

5.1.1 The algorithm of the Load agent

The load agent using the minimum spare capacity on the route as a shortest path criterion works as follows:

Travelling over the network the load agent makes a distinction between two kinds of nodes: permanent and temporary labelled nodes. The agent maintains lists with records of both kinds of nodes, with the following data-fields:

- The identifier of the node
- The largest spare capacity of the route from that node to the agent's source node. This is the route where the minimum spare capacity is maximal.
- the neighbour of the node on this route.

The goal of the load agent is to update the routes from every other node in the network to the agent's source node. Having done that, the agent is finished and terminates. The algorithm is as follows:

1. When a load agent is launched on a node, the node sets a flag ‘load agent working’. The agent starts by creating a permanent label for its source node, with spare capacity infinity and no contributor to the best route. The agent does not have records yet for any other nodes. Hereafter it visits all neighbour nodes, to create temporary records for each neighbour. In this first step the spare capacities in these records are equal to the spare capacities of the nodes themselves. The entries in the routing tables of the nodes will then be updated. This part of the initial step is also trivial: the next node in the route to the source node is the source node itself.
2. The next step is to promote one node from temporary to permanent. This will be the temporary node with the largest spare capacity in the agent's records. The agent goes to this node and moves the record for it to the list of permanent nodes. The agent then visits all of this node’s neighbour nodes that it does not have records for, and creates temporary records for them. For each of these records the contributing neighbour is the newly promoted node, and the spare capacity is the least of the neighbour nodes' own spare capacity and that recorded for the newly promoted node. Here the routing tables are updated again: the next node in the route to the source node is the new permanent node. At this stage the list of temporary records now consists of temporary records from this step, and from the earlier steps.
3. Again the temporary node with the largest spare capacity on its route to the source node is visited and made permanent. Its neighbours become temporary, their routing tables are updated, so that the agent is ready to promote another node. Steps 2 and 3 are repeated until the agent has visited all nodes in the network. At this point all routing tables have been updated, so the agent goes back to its starting node to undo the flag ‘load agent working’, and dies in peace.

The new routes that are found by a load agent are the optimal routes given the information gathered by the agents: the route from every node on the network to the agents’ starting node has a maximised minimum spare capacity. However, traffic patterns change while the agent is working, and so the information gathered by the agent need not accurately reflect the situation at a given moment - i.e. the routes are not guaranteed to be optimal. Note further that the agent only looks at the minimum spare capacities on the routes, and does not take the spare capacities of other nodes in the route into account. (When two temporary nodes are labelled with the same spare capacity available, the load agent makes an arbitrary choice which of these nodes to promote.)

5.1.2 Difference from the original approach

There is a major difference from the approach in the BT work: we changed the direction in which routing tables were updated. In (Appleby & Steward, 1994) load agents did not update the routing tables in the direction of their source node, but in the direction of the newly visited nodes, and from all nodes on the route between this node and the source node. In this way two load agents may at the same time do updates of routes to the same node. As these agents might

have different data, because of constant network changes, we suspected that circular routes might occur in the network. In early simulations we observed such circular routes. By changing the direction in which updating occurs we avoided this problem. However, we lose the possibly beneficial effect that load agents also update routes to nodes in the network other than their own source node.

5.1.3 Another criterion for 'shortest route'

We made an improvement to the load agent by storing the total sum of squared utilisations of all nodes on the route from that node to the agent's source node, instead of the largest spare capacity of the route. (The node utilisation is the percentage of the node's capacity that is occupied by calls.) The load agent then promotes temporary nodes with the smallest sum of squared node utilisations in its records. In this case the routing tables can only be updated when the node is being promoted, because when a node is still temporary it may be possible to find a better solution for that node later. The agent finishes as soon as all nodes have been promoted.

The reason why we chose a different criterion for 'shortest route' was that we observed relatively long routes in simulations where load agents maximise the minimum spare capacity. A call on such a long route occupies more nodes and this additional demand on network resources may cause subsequent calls to fail; other load agents may then amend the route to follow an even longer path. Our improved algorithm counteracts this by taking the spare capacities of all nodes in the route into account, and leads to shorter routes (routes with fewer nodes). Note further that by squaring the utilisation, the relative influence of heavily utilised nodes is increased.

5.2 The Parent Agent

The next level of control in the network is provided by parent agents. They travel around the network and launch load agents where network management is needed. The decision to launch a load agent is made on the basis of information gathered, and a set of heuristic rules.

The parent agent travels randomly around the network to gather information about the nodes. At each node the agent visits, it records the following data fields:

- The traffic destination rate. This is the number of calls that have the node as their destination
- The utilisation of the node.
- The destination-rate history, which is the average destination rate of the last d visits to the node.

Further it records the following global information, when stepping around the network:

- The utilisation history, which is the average of node utilisations of the last m visited nodes.
- The number of nodes it has visited so far.
- A destination rate ranking table. This table contains a ranking of nodes according to their destination rate history.

When the agent encounters a node with a higher utilisation value than the agent's utilisation history, it assumes that traffic management is needed. The ranking table tells the agent which node is most used as a destination node of calls. The traffic to this node is likely to be the cause of some network overload. The agent then goes to this node and launches a load agent, unless the node has already got a load agent working for it. After reaching this node and launching a load agent, the parent agent continues its cycle of gathering information, and detecting where

new traffic management is needed. Note further that our parent agents look at the destination rate, whereas the original mobile agents looked at the sourcing rate of nodes. This is due to the difference of direction in which our load agents update the routing tables.

In real networks, the parent agent could also be programmed to solve problems like crashed load agents. This is not modelled in our simulation.

5.3 Parameters

As with the ants, the space of possible parameter settings is large; the values used in the simulations reported here Section 2.2 were those found to be best according to our experience with previous experiments:

- The speed of the agents is basically the same as the speed of the ants: every time step of the simulation an agent performs its task on its current node and moves to the next node.
- The parent agent takes the last 4 visits to each node into account to calculate the destination rate history.
- The global utilisation history is 60; the last 60 visits count in the calculation of the average utilisation.
- The destination ranking table has size 15. This means that if this table is smaller than 15, the parent agent will gather more information around the network
- The number of parent agents is 2.

The routing tables are initialised so that the length of every route, i.e. the number of nodes on the route, is minimal.

6 Results of the simulations

The simulation presented us with some practical problems. Because of the initial random selection of call probabilities, the random generation of calls, and the random lengths of calls, the variability between runs was very high. In order to overcome this, many runs would have to be averaged; since each run was very time consuming, an adequate number of runs would take a prohibitively long time. We therefore decided to use a repeated measurements experimental design, in which each condition is tested on the same dataset; because of the reduction in variablility, pairwise comparisons between conditions can then yield good information from a relatively small total number of runs. We proceeded as follows:

- 10 sets of call probabilities were generated using the method of Section 2.2.
- Each set of call probabilities was used to generate a call sequence lasting 15000 time steps.
- Each call sequence was split into two blocks A + B, each of 7500 time steps.

TABLE 1. Generation of call sequences for adaptation and test

call probability set	adaptation (0-7500)	test (7500-15000)
1	1A	1B
2	2A	2B
:	:	:
10	10A	10B

- Each run consisted of an adaptation period (block A) and a test period (block B). During the adaptation period the load balancing system was allowed to adapt to the call statistics. We have found that any adaptation which takes place is subsantially complete after 7500 time periods.
- During the test period (time steps 7500 to 15000) we recorded the network performance in terms of call failures. The experiments have been performed for: a fixed routing scheme without load balancing; the mobile agents; improved mobile agents as explained in Section 5.1.3; ants without noise; and ants with 5% noise. The experimental design is illustrated by Table 2 and the results in Table 3.

TABLE 2. Experimental design 1

adaptation (0-7500)	**test (7500-15000)**
1A	1B
2A	2B
:	:
10A	10B

TABLE 3. The mean percentages (ten experiments each) and standard deviations of call failures for unchanging call probabilities

	Mean	**Standard dev.**
Without load balancing (fixed, shortest routes)	12.57%	2.16%
Original mobile agents	9.19%	0.78%
Improved mobile agents	4.22%	0.77%
Ants (0% noise)	1.79%	0.54%
Ants (5% noise)	1.99%	0.54%

In order to find out the extent to which the results are due to the dynamic actions of the load balancing system, rather than to the convergence to a good set of routing tables specific to a particular set of call probabilities, we repeated some of these experiments, but this time we froze the routing tables for each method after the adaptation period; i.e. we stopped launching mobile agents or ants. The results are shown in Table 4.

TABLE 4. The mean percentages and standard deviations of call failures after stopping load balancing for unchanging call probabilities

	Mean	**Standard dev.**
No improved mobile agents after 7500	6.43%	2.17%
No ants (0% noise) after 7500	2.11%	0.60%
No ants (5% noise) after 7500	2.48%	0.69%

We were also interested in how well the load balancing systems will deal with a sudden change in call probabilities. We examined this by allowing the system to adapt to an adaptation block from one set of call probabilities, and then testing it with a test block from a different set of call probabilities.

The experiments as presented above were repeated, but now at time step 7500 the call probabilities and call patterns of run x changed to the call probabilities and call patterns of run x+1 in the former experiments; for example, after adaptation on block 1A, the system would be tested not on block 1B, but on block 2B. The experimental design is illustrated in Table 5.

TABLE 5. Experimental design 2

adaptation (0-7500)	test (7500-15000)
1A	2B
2A	3B
:	:
10A	1B

The mean percentages of call failures in these experiments were:

TABLE 6. The mean percentages (ten experiments each) and standard deviations of call failures for changed call probabilities

	Mean	Standard dev.
Without load balancing (fixed, shortest routes)	12.53%	2.04%
Original mobile agents	9.24%	0.80%
Improved mobile agents	4.41%	0.85%
Ants (0% noise)	2.72%	1.24%
Ants (5% noise)	2.56%	1.05%

And the same experiments where agents and ants were no longer launched after time step 7500 (see Table 7):

TABLE 7. The mean percentages and standard deviations of call failures after stopping load balancing and changing call probabilities

	Mean	Standard dev.
No improved mobile agents after 7500	8.03%	2.88%
No ants (0% noise) after 7500	4.29%	2.06%
No ants (5% noise) after 7500	4.37%	2.27%

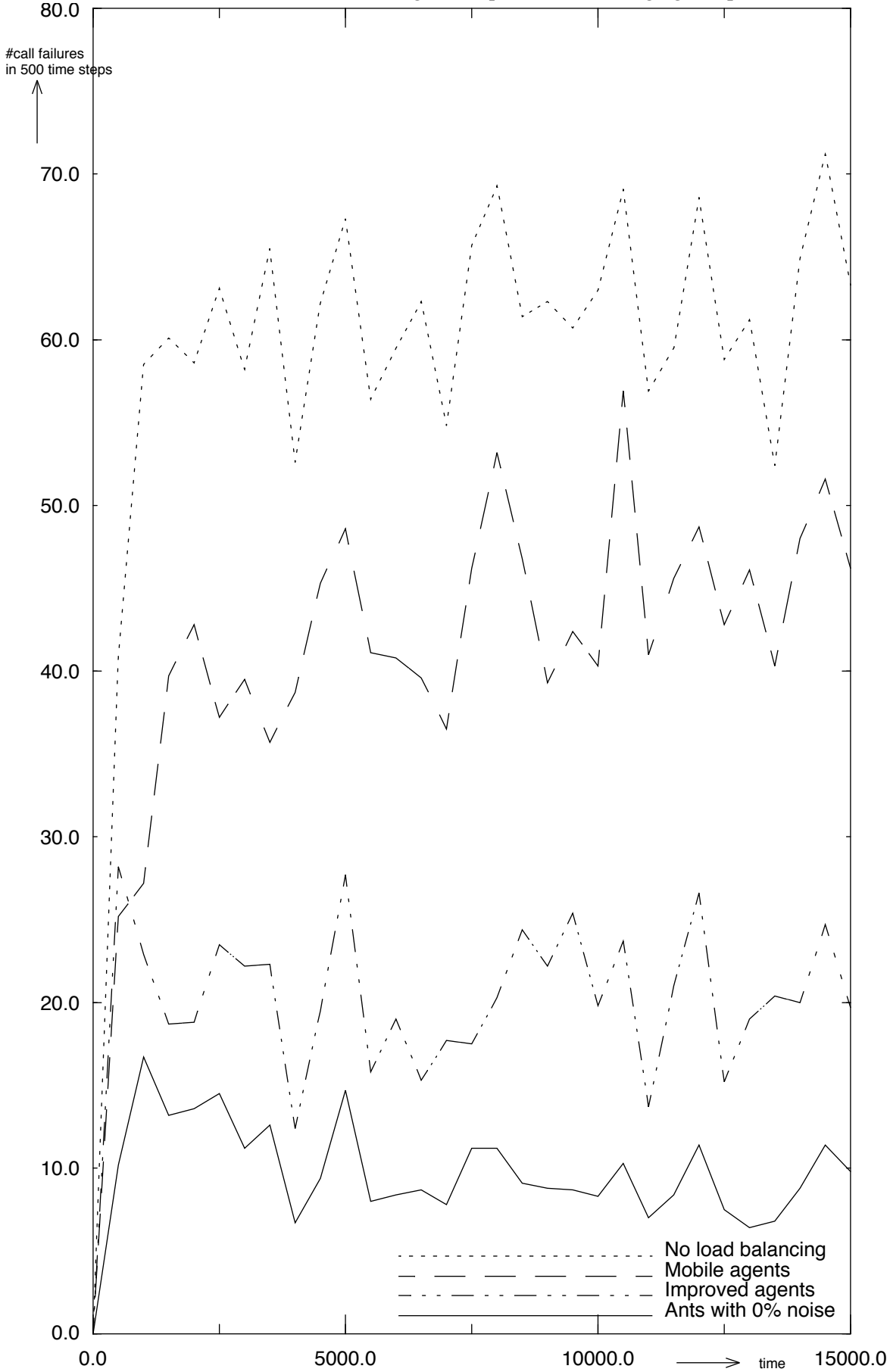
We performed the Student t-test (2-tailed) for related samples on the call failure data; we were able to pair observations derived from the same call sequences in the test period. 1B was paired with 1B, 2B with 2B, and so on. The following differences were found at the 0.01 significance level.

- Under all conditions the improved mobile agents gave significantly better results than the original mobile agents.
- All ant experiments gave significant better results than the corresponding experiments with the improved mobile agents.
- In the case of unchanging call probabilities, ants without noise gave better results than ants with 5% noise.

- All experiments with ants gave significantly better results in the simulations without changing call probabilities than in the simulations with changing call probabilities.
- All experiments with ants and agents were significantly better than the experiments where there was no dynamic load balancing.
- When the parent agents stopped launching load agents when the monitoring of call failures started (time step 7500), the results of the simulations were that significantly more calls fail in the experiments with and without changing call probabilities.
- Stopping the ants causes more call failures than leaving the ants working.
- In the situations with unchanging call probabilities, stopping the ants produces better performance than leaving both kinds of mobile agents working.
- We can not tell with sufficient significance whether noise is helping the ABC system to quickly adapt to the new sets of call probabilities.

To illustrate the performances of the different algorithms we also measured the number of call failures during every 500 time steps of each run for both the adaptation and test periods for the unchanging probability conditions. The averages over the ten runs under each condition are shown in the graph of Figure 6. Three of the graphs represent the situations where there is continuous load balancing, the fourth represents no load balancing. The graph for the ants with 5% noise is not depicted as it is very similar to that for the ants with 0% noise.

FIGURE 6. Performances of four load balancing techniques, with unchanging call probabilities.



6.1 Static solution or dynamic adaptation

One of the most interesting questions raised during the simulations on the 30-node network was whether the control systems were converging to a good static set of routing tables that was 'learned' from the statistics of the network, or whether they were constantly adapting to changing situations. A good static set of routing tables would combine information about the network topology and the call distribution statistics; routes would be sufficiently short, but would avoid the nodes likely to become congested with those particular call statistics. We think that three different forms of adaptation are possible:

- Adaptation to the network topology
- Adaptation to the call probabilities of the nodes
- Adaptation to temporary situations caused by the randomness of the call patterns

Adaptation to network topology. When we speak about adaptation to the network topology, we mean how well the control system adapts to the distribution of loads on the nodes that arises as a consequence of the specific topology. For the ABC system, the system converges to short paths during initialisation; for the mobile agents the system is initialised with the shortest paths. Due to the topology alone (and independent of a particular set of call probabilities) routing calls over these paths will already lead to congestion at certain nodes. The system will adapt to this congestion by changing its pheromone tables. Although adaptation to the distribution of network loads is a response to both the topology and the call probabilities, it is possible to get some insight into how well the system with an arbitrary set of call probabilities adapts to the topology alone. This insight can be obtained by inspecting the results of the experiments where dynamic load balancing is stopped at the same time as the call probabilities are changed. Any useful adaptation of the control system can then only be in relation to the topology, which is the only factor left unchanged.

For both the agents and the ants, the performance under these conditions is better than the experiments with fixed shortest path routing tables and no load balancing at all. Further one can clearly see that the ABC system performs better than the mobile agents (8.03% call failures for the improved agents versus 4.29% and 4.37% for the two ant experiments); this indicates that the ABC system adapts better to the the load distribution caused by the topology alone.

Adaptation to call probabilities. To see how well a method performs when adapting to a combination of topology and call statistics, one can consider the results where the launching of agents or ants is suddenly stopped, but the call probabilities remain the same. Here the ABC system performs better than the mobile agents (6.43% call failures for the agents versus 2.11% and 2.48% for both ant systems). The results are also better than those discussed above, involving only adaptation to the network topology. The size of the difference gives an indication of the influence of the call probabilities.

Adaptation to temporary situations. The performance of ants and agents on adapting to temporary situations is indicated by the differences in performance under unchanging call probabilities of the conditions where load balancing is either continued or stopped. The situations where ants are launched after 7500 time steps perform better than those in which launching is stopped. Although the ABC system has adapted to call probabilities and to the topology, routes are still changed frequently in response to temporary situations. Because this results in improved performance, we can take this to indicate that ants are dynamically adapting the

routes on the network to temporary situations as well. The same can be said about the agents, which seem in fact to be more sensitive to these temporary situations than the ants, as the difference in performance between the two agent experiments is relatively larger. Close observation of the network while running the simulation also confirmed our impression of useful reaction to temporary situations for both the ants and the agents.

Both type of systems thus appear capable of all three types of adaptation. The operation of the different components of adaptation for the ABC system may be seen in Figure 7, which shows the performance of ants with 5% noise (the graph for 0% noise looks similar and is shown in Figure 6). The corresponding graphs for the mobile agents are shown in Figure 8. All graphs are on the same scale.

- By comparing the figures with the upper graph of Figure 6, one can see that even if the call probabilities are changed and load balancing stopped, in both types of control system there is on average a better set of routing tables than with the fixed shortest paths determined without load balancing technique. This illustrates the adaptation to the network topology.
- It is obvious that as soon as the call probabilities are changed, the ABC system starts producing an increased number of call failures, after which the system adapts to the new set of call probabilities. The reaction of the mobile agent system to changing call probabilities is much faster.
- In both types of control system, the graph of the situation when there is continuous load balancing with no change in call probabilities lies lower than the one where load balancing is stopped. This indicates that some advantage of the systems comes from continuous dynamic adaptation to temporary situations.

FIGURE 7. Performance of ants with 5% noise. The average number of call failures during every 500 time steps are plotted.

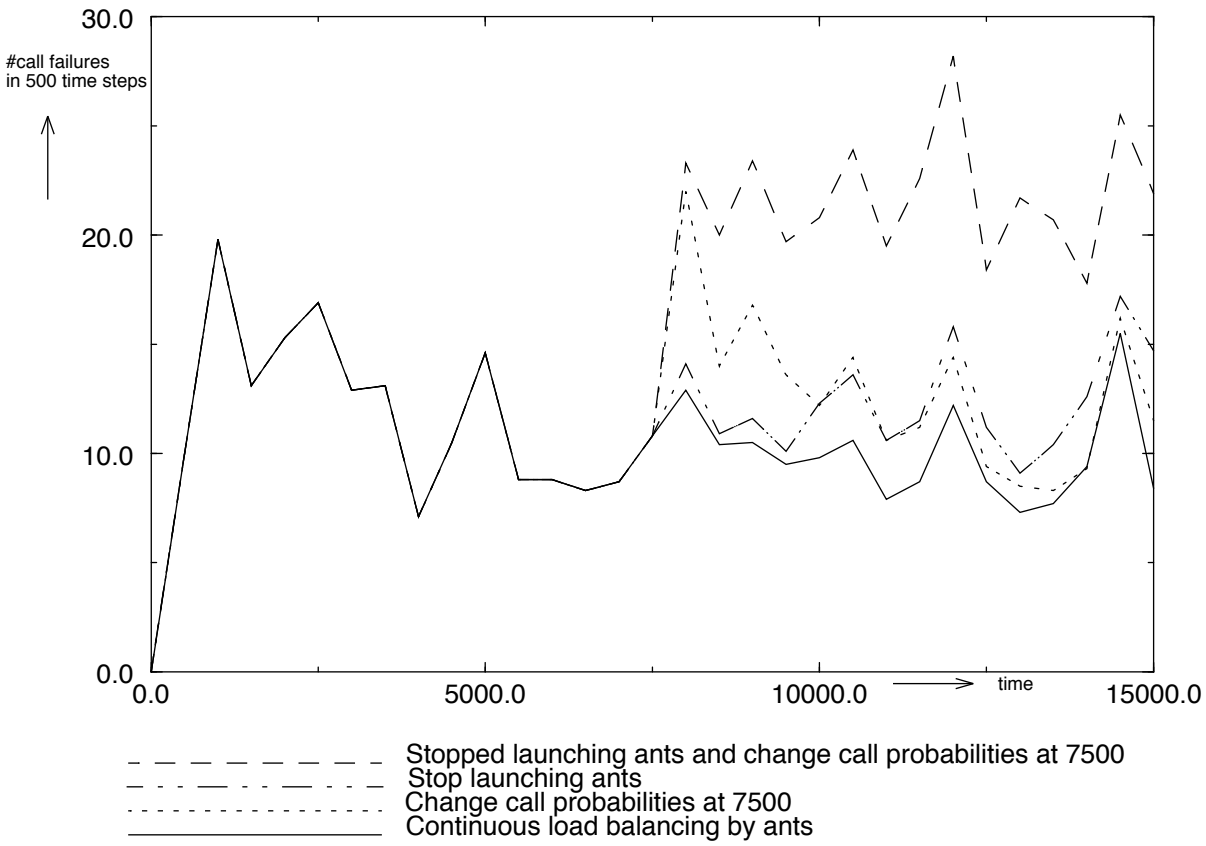
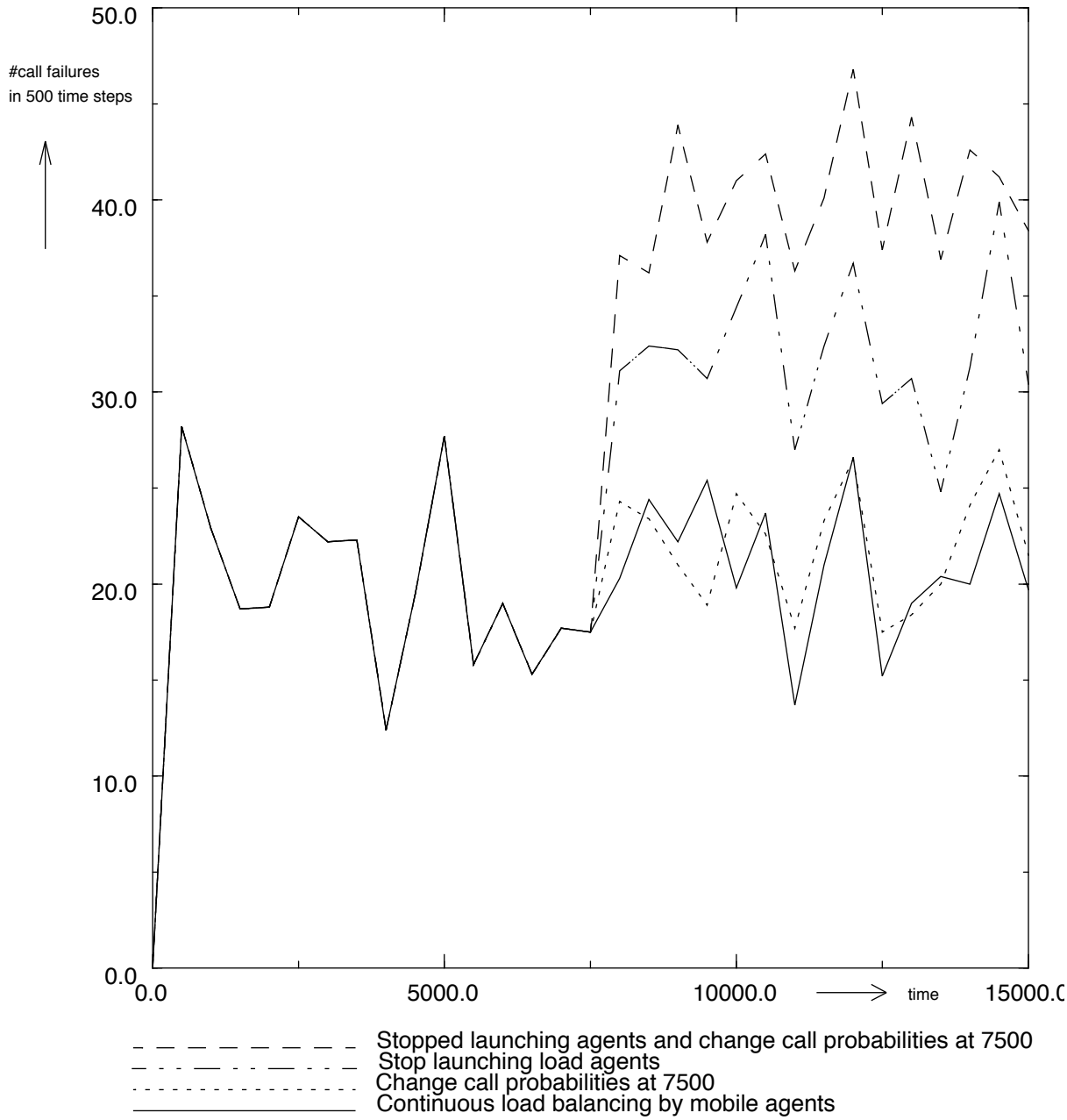


FIGURE 8. Performance of the improved mobile agents. The average number of call failures during every 500 time steps of the simulations are plotted



We note that the standard deviations are quite high for the situations when call patterns are changed and load balancing is stopped. We think this is due to the fact that the routing or pheromone tables at the moment of stopping agents or ants are partly adapted to the situation of that instant. This might be a temporary situation that is relatively exceptional, so that freezing the tables does not yield good static routing information. On the other hand it also might be a temporary situation more representative of an average situation, producing acceptable results when the tables are frozen.

In the ABC system, the balance between dynamic adaptation to temporary situations and finding good static routes could clearly be adjusted by manipulating the delay function and the pheromone update function. In our experiments we fixed these parameters on an empirical basis at levels which appeared to give low levels of call failures when adaptation was complete;

we did not systematically explore their effects on the dynamics of adaptation.

We believe that some of the power of ABC comes from the fact that the system stores information not only about good current routes, but also about good current and recent *alternative* routes. The mobile agents do not have a representation for alternative possibilities, and base their routing decisions only on temporary situations, which limits their capability to adapt to a more general, average pattern of node utilisations. We also suspect that with the mobile agents a particular problem occurs which has been recognised in the field of network routing before (Kelly, 1995): Making a new routing decision based on a temporary situation might result in a longer route that is only beneficial for a short period. Calls over this route put a higher demand on network resources and may cause a number of subsequent calls to be lost. As the demand for node capacity increases, more traffic has to be rerouted. This leads to a kind of cascade effect. In this way short term benefits are outweighed by the longer term costs. The ABC system naturally seems to find a good balance between adaptation over short and long time periods.

7 Ants versus mobile software agents

As well as the performance advantages presented in Section 6 and Section 6.1, ants have a number of other qualitative and quantitative advantages over mobile agents, as well as some disadvantages.

Consuming network resources. An ant hardly requires any bandwidth on the network: It only holds its age, and its source and destination identifiers (together with the fact 'I am an ant'). The mobile agents hold a number of relatively large tables and therefore require much more bandwidth than ants.

A property of the mobile agents from BT is that more load agents tend to get launched as the load on the network increases, to do re-routing. This might be just the moment when you do not want any more agents on the network, as it is already congested with calls. In contrast, congestion-dependent delay of the ants temporarily reduces ant traffic at congested locations. Having said that, fewer mobile agents are required than ants. The numbers of mobile agents used is of the order of tens, whereas ants are used in hundreds.

Limits to the number of agents. In principle, there is no limit to the number of ants that can be used in a network, because ants do not interfere with each other. The number of load agents on a network is much more limited. Once a load agent is launched on a particular node, another one can not be launched until the first agent finishes its job, because otherwise they would interfere with each other.

Robustness. Malfunctioning in the system might cause a mobile process such as an ant or agent to crash. If an ant crashes, this will not have a significant effect on the performance of the algorithm at all. However, if a load agent or parent agent crashes, this affects the future launching of load agents. This therefore has to be detected, and special measures have to be taken to restore the damage.

Ants can be considered as simple mobile data elements being processed upon by the node, rather than mobile computational processes, and so no problems will occur due to corruptly programmed ants. As an individual ant has a small influence on the system, damage caused by a corrupt ant will be limited. However, a load agent on its own can change all the routes to its source node, and can therefore cause much more damage if its state is corrupted.

Computational issues. Ants are likely to require more computation on the nodes of the network than the mobile agents, due to the extensive use of random generators. Further, with ant-based control, nodes need to allocate more space for their pheromone tables than is needed when normal routing tables are used. However, these issues do not affect bandwidth or switching capacity, which is our main concern.

Bidirectional routes. During the simulations of the ant controlled system, the route from one node to another tends most of the time to be the same that in the opposite direction. This is probably due to our mechanism of trail laying, where ants from complementary source and destination nodes mutually reinforce one anothers' trails. The mobile agents do not have this property. At first sight, this property might seem to be disadvantageous for good load balancing, but we believe that this will only make a significant difference in small networks.

Circular routes. In principle, ABC systems have the potential to yield circular routes. However, this situation was not observed, except when the noise was extremely high, or the initialisation period much too short. The mobile agents as implemented here are guaranteed not to result in circular routes.

8 Conclusions and future work

We implemented a completely decentralized adaptive control system for telecommunications networks which made use of emergent collective behaviour arising from the interactions between mobile objects modelled on ants, and tables on network nodes.

The principles of the algorithm are simple and general. We believe that the general framework for ant-based solutions presented here is capable of solving load balancing problems in large networks, both circuit switched and packet switched. We do not yet know exactly how the statistical and topological properties of the network influence the ideal parameter settings. But as shown here, even tuning parameters by hand can lead to a well balanced system.

The balance obtained is a good example of emergent organisation. The individual ants are remarkably simple, but the resulting structures enable very efficient use of a resource like a telecommunications network. We have identified three possible types of adaptation to the characteristics of such networks, and ABC systems show themselves capable of good performance on all three types.

We believe that ABC systems can be used to solve a large variety of optimisation problems, eg:

- distributing loads of interconnected processors on parallel machines and managing interprocessor communication for complex programs;
- material flow in production environments;
- optimal routing on integrated circuit-boards;
- organising public transport schemes.

Much investigation of the basic principles of the ant algorithm remains to be done. So far, our experiments have not yet enabled us to make statements that are sufficiently supported by statistics about the influence of the number of ants used in the simulations. We also do not know exactly how variations in the ants' influence on the pheromones affect the system, or about the effects of the size of the delays imposed on the ants. Most choices in this work are based on a relatively small number of experiments.

It would also be useful to investigate the performance of the algorithm on extremely large or

very small networks; the large networks will tell us about scaling, and the small neworks might assist our understanding of the basic processes which make the algorithm work. Another intriguing possibility is to use 'probabilistic routing' of calls. Here routes of calls, or perhaps a proportion of calls, would not be chosen according to the largest probabilities in the pheromone tables, but randomly according to these probabilities. A mechanism that is assumed not to be used by natural ants, but could be useful here, is laying 'anti-pheromone'. One could let ants directly decrease probabilities in the pheromone tables in particular circumstances, rather than increase them.

The pheromone tables do not only represent the best routes, but also contain information about the relative merits of alternative possibilities if something goes wrong. Our simulations have so far been confined to examining the use of this information in dealing with node congestion; sudden node (or link) failure and restoration also needs to be simulated to examine the abilities of the ants to deal with these contingencies. The ability to cope with the insertion of new nodes and links during network extension is also a topic of interest.

Extending ant-like algorithms to situations like telecoms networks which are not found in nature will also increase our understanding of the abstract and general abilities of such algorithms over and above those applications found in nature. We hope that this increase of knowledge will in turn assist and inform biologists studying social insects.

References

Appleby, S., & Steward, S. (1994). Mobile software agents for control in telecommunications Networks. In *BT Technology Journal, Vol. 12, No.2*.

Beckers, R., Deneubourg, J.L., & Goss, S. (1992).Trails and U-turns in the Selection of a Path by the Ant Lasius Niger. In *J. theor. Biol. 159*, 397-415.

Beckers, R. Deneubourg, J.L., & Goss, S. (1993). Modulation of trail laying in the ant Lasius Niger and its role in the collective selection of a food source. *J. Ins. Behav.* (in press)

Beckers, R., Holland, O.E., & Deneubourg, J.L. (1994). From local actions to global tasks: Stigmergy and Collective Robotics. In R.A. Brooks, & P. Maes (Eds.) *Artificial Life IV*, Cambridge, MIT Press. p. 181-189.

Deneubourg, J.L., & Goss, S. (1989). Collective patterns and decision-making. *Ethology, Ecology & Evolution 1*, 295-311.

Deneubourg, J.L., Goss, S., Franks, N., Sendova-Franks, A., Detrain C., & Chrétien, L. (1990). The dynamics of collective sorting robot-like ants and ant-like robots. In J.-A. Meyer & S. Wilson (Eds.), *From Animals to Animats: Proceedings of the first international conference on simulation of adaptive behavior*. Cambridge, MIT Press.

Deveza, R., Thiel, D., Russell, A., & Mackay-Sim, A. (1994). Odor sensing for robot guidance. In *Int. J. Robotics Research, vol 13, no 3*, 232-239.

Dijkstra, E.W. (1959). A Note on Two Problems in Connexion with Graphs. In *Numerische Mathematik vol. 1*.

Franks, N.R. (1989). Army Ants: A Collective Intelligence. In *American Scientist, Volume 77*, March-April.

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 R.N. Hughes (Ed.) *NATO AS1 Series, Vol. G20 Behavioral mechanisms of food selection*, Springer Verlag.

Grassé, P.P. (1959). La reconstruction du nid et les coordinations inter-individuelles chez Bellicositermes natalensis et Cubitermes sp. La theorie de la stigmergie: Essai d'interpretation des termites constructeurs. In *Ins. Soc., 6,* 41-83.

Kelly, F.P. (1995). The Clifford Paterson Lecture. Modelling communication networks present and future. *Proc. R. Soc. Lond. A. 444*, 1-20.

Hölldobler, B., & Wilson, E.O. (1994). Journey to the Ants. Bellknap Press / Harvard University Press.

Russell, R.A. (1995). Laying and sensing odor markings as a strategy for assisting mobile robot navigation tasks. In *IEEE Robotics and Automotion Magazine,* September, 3-9.

Schoonderwoerd, R. (1996). Collective Intelligence for Network control. *Unpublished Ir.-thesis*, Delft University of Technology, Faculty of Technical Informatics, June '96.

Stickland, T.R., Tofts, C.M.N., & Franks, N.R. (1992). A path choice algorithm for ants. In *Naturwissenschaften 79,* 567-572.

Sutton, R.S. (1990). Reinforcement Learning Architectures for Animats. In J.-A. Meyer & S. Wilson (Eds.), *From Animals to Animats: Proceedings of the first international conference on simulation of adaptive behavior*. Cambridge, MIT Press. p. 288 - 296.

Theraulaz G., & Bonabeau, E. (1995). Coordination in distributed building. In *Science, 269*, 686-688.

Wilson, E.O. (1975). Sociobiology, Belknap Press / Harvard University Press.