Self-Organisation in Ant-based Peer-to-Peer systems

Richard Paul

Doctor of Philosophy
Institute of Computing Systems Architecture
School of Informatics
University of Edinburgh
2008

Abstract

Peer-to-peer systems are a highly decentralised form of distributed computing, which has advantages of robustness and redundancy over more centralised systems. When the peer-to-peer system has a stable and static population of nodes, variations and bursts in traffic levels cause momentary levels of congestion in the system, which have to be dealt with by routing policies implemented within the peer-to-peer system in order to maintain efficient and effective routes.

Peer-to-peer systems, however, are dynamic in nature, as they exhibit churn, i.e. nodes enter and leave the system during their use. This dynamic nature makes it difficult to identify consistent routing policies that ensure a reasonable proportion of traffic in the system is routed successfully to its destination. Studies have shown that churn in peer-to-peer systems is difficult to model and characterise, and further, is difficult to manage.

The task of creating and maintaining efficient routes and network topologies in dynamic environments, such as those described above, is one of dynamic optimisation. Complex adaptive systems such as ant colony optimisation and genetic algorithms have been shown to display adaptive properties in dynamic environments. Although complex adaptive systems have been applied to a small number of dynamic optimisation problems, their application to dynamic optimisation problems is new in general and also application to routing in dynamic environments is new. Further, the problem characteristics and conditions under which these algorithms perform well, and the reasons for doing so, are not yet fully understood. The assessment of how good the complex adaptive systems are at creating solutions to the dynamic routing optimisation problem detailed above is dependent on the metrics used to make the measurements.

A contribution of this thesis is the development of a theoretical framework within which we can analyse the behaviours and responses of any peer-to-peer system. We do this by considering a peer-to-peer system to be a graph generating algorithm, which has input parameters and has outputs which can be measured using topological metrics and statistics that characterise the traffic through the network. Specifically, we consider the behaviour of an ant-based peer-to-peer system and we have designed and implemented an ant-based peer-to-peer simulator to enable this.

Recently methods for characterising graphs by their scaling properties have been developed and a small number of distinct categories of graphs have been identified (such as random graphs, lattices, small world graphs, and scale-free graphs). These graph characterisation methods have also enabled the creation of new metrics to enable measurements of properties of the graphs belonging to different categories.

We use these new graph characterisation techniques mentioned above and the associated metrics to implement a systematic approach to the analysis of the behaviour of our ant peer-to-peer system. We present the results of a number of simulation runs of our system initiated with a range of values of key parameters. The resulting networks are then analysed from both the

point of view of traffic statistics, and also topological metrics.

Three sets of experiments have been designed and conducted using the simulator created during this project. The first set, equilibrium experiments, consider the behaviour of the system when the number of operational nodes in the system is constant and also the demand placed on the system is constant. The second set of experiments considers the changes that occur when there are bursts in traffic levels or the demand placed on the system. The final set considers the effect of churn in the system, where nodes enter and leave the system during its operation. In crafting the experiments we have been able to identify many of the major control parameters of the ant-based peer-to-peer system.

A further contribution of this thesis is the results of the experiments which show that under conditions of network congestion the ant peer-to-peer system becomes very brittle. This is characterised by small average path lengths, a low proportion of ants successfully getting through to their destination node, and also a low average degree of the nodes in the network. This brittleness is made worse when nodes fail and also when the demand applied to the system changes abruptly.

A further contribution of this thesis is the creation of a method of ranking the topology of a network with respect to a target topology. This method can be used as the basis for topological control (i.e. the distributed self-assembly of network topologies within a peer-to-peer system that have desired topological properties) and assessing how best to modify a topology in order to move it closer to the desired (or reference) topology. We use this method when measuring the outcome of our experiments to determine how far the resulting graph is from a random graph. In principle this method could be used to measure the distance of the graph of the peer-to-peer network from any reference topology (e.g. a lattice or a tree).

A final contribution of this thesis is the definition of a distributed routing policy which uses a measure of confidence that nodes in the system are in an operational state when making calculations regarding onward routing. The method of implementing the routing algorithm within the ant peer-to-peer system has been specified, although this has not been implemented within this thesis. It is conjectured that this algorithm would improve the performance of the ant peer-to-peer system under conditions of churn.

The main question this thesis is concerned with is how the behaviour of the ant-based peer-to-peer system can best be measured using a simulation-based approach, and how these measurables can be used to control and optimise the performance of the ant-based peer-to-peer system in conditions of equilibrium, and also non-equilibrium (specifically varying levels of bursts in traffic demand, and also varying rates of nodes entering and leaving the peer-to-peer system).

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Richard Paul*)

This thesis is dedicated to S.C and W.P Paul, they did more good than harm

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The main question of this thesis

The main question this thesis is concerned with is how the behaviour of the ant-based peer-to-peer system can best be measured using a simulation-based approach, and how these measurables can be used to control and optimise the performance of the ant-based peer-to-peer system in conditions of equilibrium, and also non-equilibrium (specifically varying levels of bursts in traffic demand, and also varying rates of nodes entering and leaving the peer-to-peer system).

1.1.1 Overview of the thesis work and rationale

Peer-to-peer systems are a form of distributed computing that is highly distributed. As such peer-to-peer systems have advantages of robustness and redundancy over more centralised systems. Variations and bursts in traffic levels cause momentary levels of congestion in the system, which have to be dealt with by routing policies implemented within the peer-to-peer system in order to maintain efficient and effective routes. This is true even when the population of nodes in the peer-to-peer system is static.

During their operation peer-to-peer systems exhibit churn i.e. nodes enter and leave the system. The dynamic nature of the peer-to-peer system means the problem of identifying consistent routing policies that ensure a reasonable proportion of traffic in the system is routed successfully to its destination is very difficult. Furthermore, from empirical studies we see that churn in peer-to-peer systems is difficult to model and characterise.

The task of creating and maintaining efficient routes and network topologies in dynamic environments, such as those described above, is one of dynamic optimisation. Complex adaptive systems such as ant colony optimisation and genetic algorithms display adaptive properties in dynamic environments. The application of complex adaptive systems to dynamic optimisation problems in general is new. In addition the application of complex adaptive systems to routing in dynamic environments is only now being examined. The problem characteristics and conditions under which these algorithms perform well, and the reasons for doing so, are not yet fully understood. The assessment of how good the complex adaptive systems are at creating solutions to the dynamic routing optimisation problem detailed above is dependent on the metrics used to make the measurements.

There are two possible categories of metrics that can be used in this instance. Firstly, traffic metrics, which define the characteristics of the traffic handling capability of the peer-to-peer system. Secondly, a way of measuring the topology of the network produced by the peer-to-peer system is required. While it is true that the traffic statistics are sufficient when considering the ability of the system to move data from source to destination node, traffic statistics alone do not give an adequate picture of the redundancy and resilience properties of the network of the peer-to-peer system. Both categories of measurements can be used in conjunction with one another in order to form a comprehensive view as to the behaviour of a peer-to-peer system.

The model of preferential attachment [3] has been used to describe how networks grow, and their topology evolves, over time. As we explain in this thesis, the preferential attachment model is contingent on two main assumptions being true:

- The network has to be growing in terms of the number of nodes within it.
- The probability of an existing node forming a new connection to another node must be proportional to the degree of the node from which the connection is being formed.

These assumptions have limited validity in the normal operation of peer-to-peer systems. This raises the question, what is the best way of describing how peer-to-peer systems self-organise in response to external stimuli, and their own internal states of congestion? In this thesis we address this.

The peer-to-peer system we consider in this thesis uses Ant Colony Optimisation (ACO) as the basis for the traffic routing calculations within the nodes of the system. Although, ACO has been applied to many different scenarios, the underlying dynamics of the algorithm are not yet fully understood. In this thesis, we take a systematic approach to the measurement of the topology of the network as it evolves within the peer-to-peer system, from a range of starting conditions. By so doing, we are able to derive insight as to the effect of driving parameters on the behaviour of the ACO algorithm.

The effect of churn [1] in the system) in peer-to-peer systems is not fully understood. Relevant literature presents insight as to productive and unproductive mechanisms for recovery from a node's failed states within the peer-to-peer system. Further observation is made about the catastrophic consequences of unproductive recovery. Specifically, the distinction between productive and unproductive mechanisms is couched in terms of relative rates of node repair and failure. However, the literature presents no insight about the underlying dynamics of churn

[1] churn is the process by which nodes enter and leave the peer-to-peer system

on peer-to-peer systems — particularly in terms of the effect of varying the relative rates of repair and failure.

Finally, we compare the response of the ant-based peer-to-peer system to both performance failure (where nodes, fail to process traffic due to congestion), and fault failure (where nodes fail to process traffic due to the node entering a non-operational state).

1.1.2 Justification of a simulation based approach

An alternative approach to the analysis of ant peer-to-peer systems is the use of process algebras. Here the process algebra is used to describe the possible decisions and actions that the ant in an ant system might make. Wighted process algebras take into consideration that the different possible decisions and actions that an agent might take vary in terms of probability. An example of a weighted process algebra is WSCCS, the interested reader is referred to [69] for an introduction to WSCCS, and an application of WSCCS to the analysis of task allocation between ants is presented in [70].

The main aim of this thesis was to explore the underlying dynamics of the ant swarm based routing in the context of peer-to-peer systems. The justification for using a simulation based approach in this analysis, was to most directly see the effect of varying parameters on the emergent behaviours of the swarm peer-to-peer system. In order to do this, a thorough understanding of how the calculations made as the ant passed through the system was required. Furthermore, an exploration of how changing the mechanism by which the ants routed themselves through the system, on the emergent properties of the system was to be carried out. The development of a simulator and simulation based approach was most conducive to these goals.

1.2 Contribution of the thesis

This thesis focuses on the underlying dynamics of a peer-to-peer system using swarm routing based on ACO for internal traffic routing. The first task has been to develop a theoretical framework that can be used to describe the relationships between the values of the output variables, and the values of the driving parameters. The main challenge has been to develop a model that does justice to the complexity of the factors involved in routing within a peer-to-peer system. A further challenge has been the choice of the input driving parameters and output measurable parameters that we have used to document the behaviour of the peer-to-peer system. The simulation model that has been developed supports the following functions:

- Discovery of new destination and intermediate routing nodes by source nodes through the information collected by ants as they traverse through the peer-to-peer system.

- Queueing of traffic at nodes in the peer-to-peer system.

- Parametrisation of routing decisions at nodes using both long and short term criteria.
- Input of user-defined traffic profiles (and demand functions) during the course of the simulation.
- The operation of a reliability model enabling the simulation of nodes entering and leaving the peer-to-peer system.
- A reporting function that enables the collection of traffic and topological statistics from the peer-to-peer system.

Using this simulation model the main contributions of this thesis are as follows:

- The development of a theoretical framework that can be used to describe the topological and traffic characteristics of the peer-to-peer system in response to different driving parameter values of the simulation. This theoretical framework is independent of any underlying assumptions about the mechanism by which the network grows.
- The definition of a novel method of measuring the distance of a reference network topology from a topology generated by the ant peer-to-peer simulator.
- Insight into the effects of variations of driving parameter values on the ant peer-to-peer system under conditions of equilibrium, bursty traffic, and nodes entering and leaving the system.
- The specification of a routing algorithm that uses a measure of confidence in the information about the operational state of the nodes in the peer-to-peer system as part of the routing calculation.

In order to develop the model stated above and also generate the experimental results, a number of assumptions have had to be made about the operation of the peer-to-peer system. These are:

- The nodes in the peer-to-peer system fail independently of one another.
- There is a random distribution of source and destination nodes in the traffic profile that the peer-to-peer system has to cope with.
- Ants arriving at nodes are processed on a first come first served basis.

1.3 Thesis overview

This thesis is comprised of four main sections in addition to this introductory chapter. Firstly, we deal with the *background material* necessary to understand the main simulation study. The background material is described in the literature survey, Chapter 2, and also the chapter on theoretical concepts, Chapter 3, where the main conceptual ideas are presented. Secondly, in the *tools* section the simulation tool that has been developed is described in Chapter 4. The *experimental* section is comprised of three chapters. Equilibrium behaviours (Chapter 5) explore the behaviour of the system under constant loadings. The chapter on burstiness (Chapter 6) investigates the behaviour of the system in response to bursts in the demand placed on the system. Finally, damage behaviours (Chapter 7) indicate the response of the system to nodes entering and leaving the peer-to-peer system at varying rates. Finally, the main *conclusions* are drawn from the study in Chapter 8.

The structure of each of the individual chapters, and the constituent material is described below.

1.3.1 Thesis structure

In Chapter 2 we first survey the literature relevant to this thesis. We consider condensed matter formulations of network topologies, where the issues of preferential attachment, and other growth models of networks are considered. This treatment of network topologies assists in the understanding of how networks grow and adapt. The wider issues of self-healing and autonomic systems are also considered. Literature on Ant Colony Optimisation (ACO), and the application of ACO to routing in peer-to-peer systems are reviewed. Literature regarding *churn* in peer-to-peer systems is reviewed and presented where the main open questions regarding churn are highlighted. Mobile Ad Hoc Networks (MANETs) are examined as an example of a solution to robust routing in dynamic environments. AntHill is then cited as an example of more elaborate ant-based peer-to-peer systems. Finally, the ant-based approaches to routing are contrasted with cognitive packet networks, where genetic algorithms are used as the main learning algorithm for learning good routes.

The main theoretical concepts used in this thesis are introduced in Chapter 3. A contribution of this thesis is the definition of a theoretical framework within which the main mechanisms of a peer-to-peer system can be placed. The workings of ACO are explained both in general and also in particular applications to network routing in peer-to-peer systems. We also explain the experimental approach taken when assessing the behaviour of the ant peer-to-peer system. We divide the experiments into two categories; equilibrium, and non-equilibrium. In equilibrium experiments we measure the behaviour of the peer-to-peer system in response to constant external stimuli (in this case demand function). Under these circumstances any form of self-organisation or structure in the peer-to-peer system is formed as a consequence of optimisation decisions being made internal to the peer-to-peer system. By contrast, non-equilibrium experiments consider the behaviour of the peer-to-peer system in response to the demand function and also changes in the environment such as changes in the demand function, or the entering or leaving of nodes in the system. These two examples of non-equilibrium behaviours can cause the system as a whole to fail for different reasons (node congestion in the first case and insufficient number of nodes being in the operational state in the second case). Both non-equilibrium scenarios are contrasted in this thesis as both compromise the operation of the peer-to-peer system. A further contribution of this thesis is presented as an analytical framework for the categorisation of graph topologies, and the definition of a metric (the KL distance) that enables the measurement of the distance of a given graph topology from a reference topology (such as a lattice, small world, tree, or random graph).

The two versions of the software simulation tool that have been developed in order to conduct the experiments, the ant peer-to-peer simulator, are described in Chapter 4. Specifically, the main modes of operation of the simulation tool, and the use of configuration files to enable this, are described. We then focus on the application architecture of the simulation tool, where the function of each object in the application architecture is described, and also the main control loop where the messages that get passed between the objects during the course of the operation of the simulator are described. We also state the reasons for the production of a second version of the simulation tool. Finally, the methods for testing and validation of the different versions of the simulation tool are stated, and the outcomes of the regression tests between the two main software versions are documented.

A contribution of this thesis is documented in the first experimental chapter (chapter 5) which focuses on the results of simulation experiments conducted when the simulator is in equilibrium conditions, and a series of constant levels of demand (or levels of traffic) are placed on the system. The effects of changes in values in the driving parameters of initial range of awareness of the node, the pheromone persistence rate, and the percentage of nodes sending ants, on the traffic and topological measurements have been analysed. The Kullback-Leibler distance (as defined in Section 3.5.2, page 47) is also used to measure the effects of altering the initial radius of awareness, pheromone persistence rate, and the percentage of nodes sending ants on the overall network topology. We see that each of the driving parameters has very different effects on the overall ant peer-to-peer system.

The second experimental chapter (Chapter 6), considers the effect of changes in the demand function on the system. In order to do this we introduce the concept of the *demand multiplier*, which allows independent control of the number of nodes sending ants and the total number of ants sent each simulation iteration. A wavelet model is used to generate synthetic traffic data. Using the wavelet model it is shown that the burstiness characteristics of the traffic data can easily be varied using a simple parameter — the Hurst exponent. Experiments are performed in order to assess the effect of different levels of burstiness in traffic on the resultant network topology of the ant peer-to-peer system, and also the traffic statistics. In this way the effects of system degradation through network congestion can be examined. The results of the experiments show that in some cases the effects of the burstiness in the traffic profile (such as congestion) can be smoothed by a dynamic variation of the pheromone persistence rate.

The third experimental chapter (Chapter 7), concentrates on the effect of nodes entering and leaving the ant peer-to-peer system or churn. We use a simple two state discrete time Markov model in order to describe the behaviour of each individual node within the peer-to-peer system, which in turn aggregates to an overall churn process within the peer-to-peer system. We formulate a configuration space for the two state discrete time Markov model, and use the configuration space to specify a range of experiments to examine the effect of varying magnitudes of churn on both topological metrics, and traffic metrics. The contribution of this chapter is that the experiments show that there is a maximum level of churn that the system can handle. This is indicated by a collapse of all traffic and topological metrics at values of churn greater than this. A further consequence of churn is described in terms of the effect it has on the confidence in routing information in the routing tables of the nodes. We show also how the measurement of confidence of information can be used in conjunction with pheromone level in order to make more accurate decisions about how to route ants around the ant peer-to-peer system. This method is encoded in an algorithm which is also defined in this chapter and is presented as a conjecture, as the algorithm has not been substantiated by experimental evidence.

Finally, in Chapter 8 we see the main conclusions for the thesis by contrasting the effects of burstiness in traffic and damage on the peer-to-peer system. We also describe possible further work originating from the work presented in this thesis.

Chapter 2

Literature survey

2.1 Overall Aim

The overall aim of this chapter is to survey the literature relevant to the work presented in this thesis. We also use the literature to justify the approaches taken in doing the research work of the thesis. The literature survey considers work from a number of related disciplines in order to address adequately the different aspects of the problem of describing self-organisation in peer-to-peer networks. The literature survey is also intended to prepare the reader for the material detailed in the theoretical framework.

This chapter is organised as follows. Firstly, in Section 2.2 we present a review of condensed matter formulations of network topologies. Section 2.4 then reviews the relevant literature on ant colony optimisation. The literature on self healing networks and wutonomic systems is review in Section 2.3. A general treatment of Ant Colony Optimisation is given in Section 2.4, where both the basic ideas are presented and also the literature on the theoretical aspects of the algorithm. We pay specific attention to failure and reliability models in peer-to-peer systems in Section 2.5. Further we then focus on mobile ad hoc networks and peer-to-peer simulators in Section 2.5.2. All of the previous literature is bought together when considering Bison and AntHill in Section 2.6, which are two frameworks for using biologically inspired computing in distributed peer-to-peer systems. Alternative models of self-regulating networks are examined in Section 2.7. Each of these sections has a sub section indicating how the literature that has been reviewed relates to the main work in this thesis.

2.2 Condensed matter formulations of network topologies

In this section in the literature review we consider the relevant work on evolution of networks and also some of the theoretical foundations behind the evolution of networks from the point of view of condensed matter physics.

The theory of small world graphs and their theoretical properties is detailed in [51] where it is explained that the average path length of small world graphs scale with the log of the number of nodes within them, not linearly, when there is a large number of nodes in the system. Conversely, when there is a small number of nodes in the system the scaling of the path length should be linear. The method of constructing small world graphs by subjecting regular lattice type graphs to random re-wiring is also detailed. A scaling function and scaling variable is detailed for the small world graphs. The interpretation of the scaling variable is given as two times the number of short-cuts on the graphs for a given value of p (where p is the probability that a connection in a lattice graph will be randomly rewired to another node), and the scaling function is given as the average fraction by which the vertex-vertex distance is reduced for a given value of the scaling variable. It is also shown that calculating the path length distribution analytically from this model is very difficult. The method of rewiring regular lattice graphs randomly to make small world graphs is explored further in [1], where it is shown that when many networks grow they do so exhibiting *preferential attachment*, where the probability of a new node in the network attaching to an existing one is dependant on the number of connections the existing node already has. Through analysis it is shown that this mechanism of preferential attachment creates a scale-free stationary state, where the degree distribution of the graph follows a power law $P(k) = k^{\gamma}$. A scale free-network can be constructed by progressively adding nodes to an exisiting network and introducing linkts to existing nodes with proeferrential attachment so that the probability of linking to a given node i is proportional to the number of existing links k_i that the node has i.e.,

$$P(linkingtonodei) = \frac{k_i}{\sum_j K_j}$$

where the degree distribution of the graph follows a power law.

Two models for the development of scale-free graphs are examined in detail. The first model examines what happens when the network grows at a constant rate and the probability of attachment to nodes is independent of the degree of the node being attached to. It is found that under these circumstances no scale-free portions of the network are generated. The second model is similar to the first, in the respect that there is a preferential attachment rule. In this second model however, there is no growth as the number of nodes in the system remains constant throughout the simulation. It is found that in neither the first or the second case scale-free graphs are generated. This leads to the conclusion that both growth and preferential attachment are needed in order to create scale-free graphs.

Models of preferential attachment have been applied to real networks. Specifically, these models have been used to measure the scale-free topology of the world wide web [3]. Here a simple preferential attachment model is fitted to topological data from the world wide web. Using this the average path length and diameter for the world wide web are derived. In a related topological measurement study [30], we see a simple preferential attachment model applied to a range of networks including those of academic citations, the Internet, and actor collaboration.

The underlying statistical physics for preferential attachment is treated in [2] where the connection between condensation processes in Bose-Einstein systems[1], and condensation processes in networks, is made. Here a simple mapping is used to make a correspondence between the degree of a node in a network and a corresponding energy level in a Bose-Einstein system. A further observation about the phases of the network topology is made, by considering the limiting cases of the Bose-Einstein model. The underlying statistical physics is explained in great detail in [2]. Firstly, a number of empirical topological studies are presented of common networks such as the Internet, citation networks, and science collaboration networks. After looking at the requisite theory for random graphs, and percolation theory results concerning the topological properties of both small-world, and scale networks are derived from the associated models. The evolution of networks are then considered in scenarios of growth, competition for growth, the ageing of nodes, and also network attack and failure, using preferential attachment rules to form and re-form the network.

2.2.1 Relevance of condensed matter formulations of network topologies

In this thesis we consider the organisation of peer-to-peer systems in terms of an evolution of a network. The models detailed in the literature cited above, give insight into how networks evolve, and therefore also how self organisation in peer-to-peer systems might occur. Specifically we consider peer-to-peer systems to be graph generating algorithms. The models of preferential attachment and condensed matter physics give insight into useful metrics to use in order to measure the properties of the resultant graphs. A further advantage of these models, and the metrics that they present, is the categorisation of graphs into sets with differing topological properties (e.g. small world, random, and lattice). The existing literature, however, presents no continual metric for the categorisation of these graphs. In other words no way of measuring the distance of a given graph from each of the underlying categories has been presented.

2.3 Self-Healing Networks and Autonomic systems

In this section we review the relevant literature on self-healing networks and autonomic systems. This literature is relevant because our Ant peer-to-peer system exhibits self-regulation.

[1] A Bose-Einstein system is a system comprised of Boson particles. These Boson particles organise themselves, and the energy of the physical system they constitute by adhering to particular statistical constraints, namely Bose-Einstein statistics in which there is no upper limit to the number of particles occupying a given quantum state. The analogy with networks is created by considering a given quantum state to correspond to a node with a given number of connections or degree. The Bose-Einstein statistical framework can then easily be modified to describe the distribution of connections between nodes in a network.

An autonomic system is one where the system self-regulates, specifically this means that the system self-configures, self-heals, self-protects, and self-optimises in response to changing environmental conditions. An architecture for an autonomic networking system is proposed in [64] where we see a three tier architecture proposed in order to enable event correlation (where related fault events and alarms are correlated and categorised). Specifically, the events in question are fault events. Tier one of the architecture relates to visualisation correlation where new alarm correlations are created from visualising the fault management data. Tier two is concerned with knowledge acquisition or rule-based correlation where rules are used to anticipate downstream effects and consequences. Finally, in tier three we see data mining used as the main tool for the discovery of fault alarm correlations. This architecture is then automated in a tool which discovers correlations between faults in the system being monitored and proposes corrective courses of action in response to these. The different fault detection and recovery mechanisms are the focus of [65]. Two methods proposed here are firstly, to react to the events which faults cause, and, secondly, to monitor operating conditions against expectations. However, irrespective of the method of fault detection it is the root cause analysis of the problem that is key to the development and effectiveness of autonomic systems [63]. The relevance of autonomic systems to overall system dependability is addressed in [62] where it is observed that the self-regulation properties of autonomic systems increase system dependability.

A variety of general schemes for self-healing in networks are addressed in [52] where three schemes are proposed. Firstly, the use of dedicated resources assures that fixed capacity backup routes are available throughout the network which can be used when failures occur. Secondly, semi-dedicated resources ensure that backup paths can be constructed from a set of shared resources when faults occur. Finally, thirdly, on-demand resource schemes are tailored to the scenario where neither the route nor capacity is fixed. In this scenario topology update algorithms are employed to find extra capacity when routes fail. The conclusion of this paper [52] is that the method of self-healing used in a network must be tuned to the amount of spare resources in the network and also the intended recovery time. Comparing the three recovery schemes the authors show that there is an inverse relationship between the recovery time and also the amount of resources needed in the system. In [22] we see a self-healing routing scheme (SHAODV) based on Ad-Hoc on demand distance vector routing (AODV) in which the following processes are described in order to achieve the resilience in networks. We enumerate the steps here.

1. Construction of a stable path — this is where a stable source destination route through the network is discovered.

2. Stability detection — here every node tests the stability of the links incident to them

3. Local repair of unstable path — during the repair time routing is directed around the

break to a temporary directional path.

4. Route update — once the repair has been done the relevant entries in the routing table are updated.

5. Path reduction — the local repair may increase the length of the path, meaning that the repaired path may not be the shortest one. The path reduction step re-optimises the paths in the network.

Simulation tests are run using this routing scheme and the results show that SHAODV constructs routes more efficiently, and also with lower average route overhead than associated AODV schemes.

2.3.1 Relevance of self-healing networks and autonomic systems to this work

In this thesis we are concerned with the self-organising properties of peer-to-peer systems, in particular under conditions of varying traffic characteristics, and also churn in the system — where nodes enter and leave the system. Any peer-to-peer system needs to maintain functioning under these conditions. The literature cited above we see a number of schemes for self-healing and self-correcting in networks. We see that many of the existing schemes rely on elaborate tiered architectures to collect and filter fault information, and then issue repair actions. The Ant Colony Optimisation (ACO) Algorithm (introduced below) as applied to networking protocols, has the property that decisions about routing and repair of networks are made in a highly decentralised and dynamic manner. The aim of this thesis is to test and measure the properties of ACO in a set of dynamic conditions.

2.4 Ant Colony Optimisation

In this section we review the relevant literature on ant algorithms. However, we begin with a definition of the Ant Colony Optimisation Algorithm. Broadly the ant colony optimisation algorithm is a meta-heuristic (or a framework) for constructing optimisation solutions as described in [18]. This is done by the interplay of three procedures

- Construct Ant Solutions — this is where a number of ants concurrently visit adjacent states of the considered problem by moving through the problem graph. As the ants are routed through the problem graph they lay down pheromone on the edges of the graph.

- Update Pheromones — this is the process by which the pheromone trails are modified on the different edges of the problem graph.

It is this process of updating of pheromone trails and routing ants probabilistically on the problem graph according to the strength of the pheromone trails that enables the refinement of the solutions and the continual optimisation of the solutions produced by ACO.

There is a great deal of literature, as ant algorithms have been applied to lots of different problems. A good overall introduction to the topic of ant algorithms is given in [18], where many of the main aspects of the meta-heuristic are described in detail. In this section we survey the trends for the future research for the Ant Colony Meta-Heuristic and consider what is known about the theory behind Ant Colony Optimisation. We then consider a specific application namely routing in telecommunications networks and survey the relevant literature. The existing work on applications of Ant Colony Optimisation to peer-to-peer systems is surveyed and finally we look at attempts that have been made to parallelise the algorithm.

The trends for the future development of the ant colony meta-heuristic indicate that the there are several new areas of application available in chapter 7 of [18]. Multiple objective problems is a new area of application for ACO, and they occur where the goal is to find the best compromise between multiple objectives. Dynamic problems occur where there is a dynamic nature to the problem in which the optimisation is to be performed. Routing in telecommunications networks is an example of dynamic optimisation where points of congestion due to traffic flow cause the constraints in the optimisation problem to change from moment to moment, as points of congestion form and clear.

Despite the large number of applications of Ant Colony Optimisation(ACO), and also the wealth of new application areas for the meta-heuristic, very little is known about the underlying dynamics of the system. A review of the current theory of ACO is presented in chapter 3 of [18]. Here it is shown that ACO can be cast into a model based search framework, where candidate solutions are constructed, and then evaluated. Using this technique we see that the model parameters are the pheromone trails associated with the construction graph, and the model update rules are used to update the pheromone trails.

Another piece of theoretical insight regarding ACO comes in the form of a convergence proof, this is stated in chapter 4 of [18], that states in broad terms that

- When using a fixed lower limit on the pheromone trail (or lower limit on the level of pheromone in the pheromone trail) ACO is guaranteed to find the optimal solution.
- When converged all the ants will construct the optimal solution over and over again.

However, these proofs say nothing about the time required to find the best solution and neither is any insight offered into the relationships between the optimal solution of a system that has been perturbed in comparison to the non-perturbed system.

We now focus on the literature associated with a specific application of ACO, that is routing in telecommunication networks. We start by considering a taxonomy of routing algorithms by way of background. This is presented in chapter 6 of [18]. The main categories presented are

- Centralised routing — where a central controller is responsible for the updating of the node's routing tables and making routing decisions. This solution does not scale well and is not fault tolerant, but may be suitable for small networks.
- Static routing — where the routing is determined only by the source and destination of the packet concerned.
- Adaptive routing — where the routing policy can be adapted to local traffic conditions.
- Optimal routing — where the routing is performed according to a network wide perspective, and the goal is to optimise all individual link flows.
- Shortest path routing — where the goal is to optimise the shortest path between the greatest number of source/destination pairs of nodes.
- Distance Vector — make use of routing tables consisting of estimated distance, destination and next hop. Routing decisions at node *i* are made choosing the next hop node using local information.
- Link-state algorithms — these use a distributed replicated database which is a dynamic map of the entire network.

In some cases the categories can be combined for instance centralised routing may be both optimal and shortest path. However, for the purposes of being exhaustive we see stick to the categorisations above.

The AntNet algorithm is described in chapter 6 of [18] and [11, 10] as being a routing scheme which incorporates a learning mechanism that uses ACO. AntNet is tested against other common routing algorithms such as OSPF(static link state), SPF(adaptive link state), BF(adaptive, distance vector), and also Q-R(adaptive, distance vector). It is shown to compare favourably using delay distribution as the main metric. Three traffic profiles were used to test the routing algorithms. The first, *uniform Poisson* defined an identical Poisson process at each node for the sessions arrivals. The second *simulated hot spots* where a subset of the nodes in the network had a high rate of traffic. Sessions are opened from the hot spot nodes to all other nodes. Lastly, *temporary hot spots* were simulated by allowing hot spots to turn on momentarily during the simulation. In these studies the topology of the underlying substrate networks remains fixed. In summary, AntNet is shown to perform better in terms of throughput and delay than the other proposed algorithms. There are two reasons for this superior performance. The first reason is better distribution of global information than in other schemes through the dissemination of ants in the network enables coordinated decisions to be made. The second reason is more efficient mechanisms for updating of information in the routing tables enables less efficient routes to be pruned more effectively.

Having established the advantages of the AntNet routing scheme over many of the more widely used routing schemes we now consider the literature regarding improvements to the basic AntNet scheme. Specifically, in [4] two mechanisms for improvement are proposed. Firstly, schemes for the intelligent initialisation of the routing tables which uses information about the network topology by increasing the probability that traffic will flow between neighbouring nodes. A scheme for more intelligent update after network node failure is proposed where a measure of the ability of the failed node to handle traffic before failure is used to update the routing tables of neighbouring nodes that may send traffic to the failed node. In this way when reinitialising a node's routing table after a failure has occurred, learned information about the network topology prior to the failure can be used. Noise in the routing table is also used in order to encourage ants to explore new routes in the network. Finally, a mechanism for controlling the total number of ants in the network is proposed in order to avoid congestion in the network and improve throughput. Experimental results showed that these enhancements to the routing scheme improved the overall performance of AntNet both with respect to the original version and also the more common routing schemes stated above.

A study investigating the effect of the topological properties of the substrate graph on the overall performance of AntNet is presented in [16]. Four networks are generated with topologies that vary according to the total number of nodes, the maximum degree, the minimum degree, and the average degree. AntNet was then run using these four networks as substrates and the average packet transfer time and loss rates were measured. The effect of asserting that the routes traced by the ants in the network were loop-free was investigated. The conclusion of this work is that the loop-free feature is effective in the routing table when the underlying network is sparse (i.e. there are few cross links). A suggestion for further work is to develop an algorithm which allows the loop free feature to be turned on and off according to the local topological features of the graph.

How loops are formed in ant-based routing systems is the main focus of [9]. This is important because the formation of loops is a hazard in all distributed control routing schemes, as once formed the loops will waste network resources, by preventing the ants from reaching their destination. After analysing the scenarios in which loops are formed it is observed that, in ant systems, Canwright [9] concludes that loops are unstable in ant systems. The reason for this instability can be understood by considering the scenario where a loop is already formed and subsequent ants enter into it. In order to reinforce a route an ant must successfully make a journey from its source node to the destination node and back again. If part of that journey skirts the loop that is already formed in the network, then the reinforcement mechanism stated above states that the arc exiting the loop will also be re-inforced (otherwise the journey would not have been reinforced in the first place). This reinforcement process will have the consequence that ultimately the loop will be broken as the exit from it will be strengthened over and above those arcs which lie within the loop.

Another aspect of this self-regulating behaviour of ants is load balancing, which is treated in the context of AntNet in [31], and formulated as an optimisation problem. In this paper an algorithm for the dynamic load balancing for transmitting the data packets is proposed. A cube topology is used to evaluate the algorithm which is shown to have good network utilisation and also a low rate of bandwidth blocking. Specifically the algorithm works by using a combination of a measure of pheromone and also a measure of cost for the routes on the cube in order to determine onward routing decisions. Pheromone levels in the routing tables of the nodes are updated upon arrival of the ants at the nodes. The adaption procedure of the routes in the system occurs by four mechanisms. Firstly, calculation of the cost of transmitting the packet once it has arrived at its destination. Secondly, once the packet has arrived back at its source node calculate the cost of the overall round trip and use this to adjust the entries in the routing table of the source node. Thirdly, continually evaporate the pheromone levels in the system. Finally, fourthly, cleansing of the routing tables, by removing entries in the routing tables that point traffic to nodes that are in a failed state.

The original work introducing the use of ants in telecommunications routing was performed in [60]. Here we see a simple scheme for the use of laying of pheromone as a means of modifying the routing probabilities in routing tables in a peer-to-peer network. A general framework for ant-based control systems is defined. One detail of this mechanism that is defined is that the age of the ants are taken into account when calculating the impact they have on the routing tables of the nodes they are incident at. Older ants increment the pheromone values by smaller amounts than the related younger ones. Using this method it is shown that fewer packets are lost than in the corresponding fixed shortest path route schemes.

A further treatment of routing using ant based schemes is given in [8] where we see the use of ants in conjunction with a dynamic programming capability to determine the routes through the network. We also see that the inclusion of dynamic programming yields better performance than without. The dynamic programming scheme is crafted so that all the rows in the routing table that an ant visits are updated. The amount of pheromone deposited is calculated using relative age not absolute age. That is to say, we compute the number of iterations that have passed between the ant visiting the current node and the node whose row in the routing table is being updated.

More recently, the literature presents an application of ant systems to membership overlay design in peer-to-peer systems. In [43] we see the problem of maintaining a consistent and efficient overlay in dynamic conditions cast as a dynamic optimisation technique, where the challenge is to cope with time varying input data. The argument is made that ACO is particularly suitable for dynamic optimisation problems. The basis for the argument is that the ability of ant algorithms to construct an internal representation of the essential elements of the instance to solve makes ant algorithms appropriate for dynamic optimisation problems. The problem of dynamic overlay design in peer-to-peer systems is one where the peers have to select which other peers to communicate with. The goal of an overlay is to achieve a fair distribution of load in order to maximise the throughput of the network. Firstly, the static sub problem, where nodes do not enter and leave the system, is formulated in terms of a mixed integer problem. This is solved by Langrangian relaxation. The dynamic case, where nodes do enter and leave the overlay, is then considered. The observation is made that the frequency with which the re-optimisation occurs in the overlay network must be greater than the frequency with which the changes in the overlay network occur. However, no further analysis of this problem is preformed in terms of what the optimal ratios of these frequencies are. Three algorithms are run concurrently. The first algorithm updates Lagrangian penalties, the second is the Lagrangian heuristic, and finally the Ants (Membership overlay design problem) MOP algorithm is executed which optimises the overlay network using the sending out of ants. This paper presents evidence that ant techniques are applicable to dynamic problems, and also details a method of hybridising a mathematical technique (Lagrangian optimisation) with ant procedures.

2.4.1 Relevance of Ant Colony literature to this work

This thesis concerns peer-to-peer systems, and their self-organising properties. Specifically the peer-to-peer system being analysed uses an ant colony optimisation algorithm as the basis for the routing of traffic. As stated above there are a number of new application areas in ant colony optimisation. The dynamic optimisation of an overlay network in a peer-to-peer system under conditions of both churn and also varying traffic profile, as presented in this thesis, is an example of this. Studies regarding the topological properties of the solutions produced by ACO have up until now been confined to the analysis of loops and their properties. In this thesis we take a wider view of the topology of the solution produced by an ant based peer-to-peer system, and consider aspects of the degree distribution [2] of the nodes in the graph produced by the ant peer-to-peer system as it evolves.

The literature does present some precedent for the use of ant routing in optimisation of overlays in peer-to-peer systems. While these studies do establish that it is possible to use ant techniques in combination with langrangian relaxation and other techniques to dynamically optimise overlay networks, no results regarding the underlying dynamics (in terms of the optimal frequencies ratios of the optimisation operations during the simulation) of these systems have

[2] The degree of a node is the number of connections to it. These connections can either originate from the node in question (in the case of the out degree) or terminate at the node in question (in the case of the in degree). The degree distribution is the distribution produced when considering the proportion of the total number of nodes that have a given degree.

been presented. Nor have these frequency ratios been related to frequencies of the changes in the environment (such as the rate of churn [3] in the system).

2.5 Failure, Reliability, and Churn models and issues in Peer-to-Peer systems

During the course of their operation peer-to-peer systems exhibit *churn* in the peer membership. At the time of writing the analysis of churn within peer-to-peer systems is a relatively new topic with little in the literature. However, defining churn is difficult, and there are a number of fundamental open questions associated with this elaborated upon in [66] such as

- what are the fundamental properties of churn?
- how similar or different are churn characteristics across different peer-to-peer platforms?
- what is the correct model for churn?
- how do we best handle churn?

For the purposes of this thesis we adopt a working definition of churn to be

- The entering and leaving of nodes within a peer-to-peer system

In order to characterise churn we must find a definition of availability. One such definition is given in [56] where the ratio of the sum of the node's session times to its overall lifetime is calculated. Studies of the Gnutella peer-to-peer system have shown that the availability of the nodes within the system oscillate in a given 24 hour period [12], and the distribution of the node availability follows a log quadratic curve [12]. Further studies of peer lifetime in Gnutella, BitTorrent, and Kad have indicated that session times follow a power law distribution [66]. More theoretical models of peer lifetime have included modelling using a Pareto distribution [37] which is used to model a range of peer lifetimes and to incorporate these in the model. A further property of the Pareto distribution is that it is heavy tailed which means that nodes connected to the system for a long time have a greater probability of still being connected.

Techniques for availability management and prediction have been used to create fault tolerant distributed system management systems such as Total Recall [5]. This system has been tested under a number of repair policies primarily eager and lazy policies, where it is observed that eager policies (where the system is proactive about self-healing) will always outperform lazy policies (where the system repairs only in response to an external stimuli), in terms of the number of successfully answered queries in a distributed database. The observation made here

[3] churn is the process by which nodes enter and leave the peer-to-peer system

is that availability management is too complex for humans and software systems need to be put in place to effect this, by predicting the level of availability, determining the required level of redundancy, and initiating repair actions accordingly.

Further solutions to churn are proposed and compared in [56]. These are:

- Reactive where a node which enters a system publicises itself to the rest of the system by sending a copy of its entire leaf set neighbours to every node within the leaf set.
- Periodic, where broadcasting of new nodes occurs on a regular basis, not when a node enters the system.
- Super-peers, where structure is imposed on the peer-to-peer system, so the super-peers act as a broadcasting network for the rest of the system.

One of the main differences between proactive recovery and reactive recovery is the effect that this has on the surrounding network. It is observed in [40] that reactive recovery creates a positive feedback situation. This is because a node reacts to the loss of a neighbour by sending a copy of a new leaf set to every node within the leaf set. This broadcast causes network traffic, which in turn causes congestion, which finally gives the node originating the broadcast the impression that its neighbour nodes are in a failed state, which causes a second broadcast of the new leaf set. The solution to this feedback is observed in [40] to be to decouple the rate of recovery from the rate at which failures are discovered in the system.

In simulations of churn it is found that requests are routed to the most reliable peers [40]. Also churn in systems can cause systems to incur particular phase transitions [14]. The exact position of the threshold at which the phase transition occurs depends on the maintenance scheme in place. However, beyond this threshold the system is unstable and is not capable of routing information effectively.

2.5.1 Relevance of failure, reliability and churn models to this work

Peer-to-peer systems exhibit churn, and part of this thesis has been a study of the effect of churn on the ant system and how it copes. From the literature we see that there are a number of open questions about churn and how best to model it. Furthermore, the literature presents conflicting evidence about the nature of churn. Although observations have been made about the phase transitions that churn can cause in systems, there have not been any systematic studies about how varying the churn parameters fundamentally effects the system. In this thesis we use a simple model, with a small number of control parameters to simulate churn. Furthermore, we take a structured approach to churn, by performing a set of experiments which span a range of parameter values for churn. We then examine the trends in the data that this continuum has produced.

2.5.2 MANETS and peer-to-peer simulators

In this section we review existing simulators for peer-to-peer systems, which have been mostly motivated by the need to understand Mobile AdHoc Networks (MANETs). Both [77] and [55] define MANETs as an autonomous collection of mobile nodes which communicate over wireless links. MANETs also have the property that the topologies dynamically rearrange as nodes enter and leave the collective system. It is this dynamism in the system that has driven the evaluation of various peer-to-peer routing schemes. Applications run over MANETs tend to have similar characteristics of maintaining one or more multi-cast trees in order that peers can run multiple services, and connect to multiple peers [55]. Further, many of these applications (such as resource discovery and group messaging) will have bursty traffic profiles associated with them [55].

In order to accommodate the applications stated above a number of routing protocols have been implemented within simulators. These are as follows.

- Ad Hoc Demand Distance Vector (AODV) — where routes are obtained by a process of discovery, but routing information is stored as one entry per destination. A node satisfies a given routing request by sending a route reply back to the source node or by increasing the hop count and re-broadcasting to its neighbours.

- Destination Sequenced Distance Vector (DSDV) — where each node lists the next hop and the hop count for all reachable destinations. Two mechanisms can be implemented within DSDV., either each node exchanges routing tables in full or in increments.

- Dynamic Source Routing (DSR)— is a multi-hop routing protocol for MANETs with two phases of operation — route discovery and route maintenance. The discovery process is implemented by broadcasting a route request packet across the network.

A taxonomy of each of these routing schemes is provided in [13] and [55]. The clearest taxonomy of routing schemes is presented by [73] where routing algorithms are categorised in two ways. Firstly, according to whether they are pro-active (i.e. searching packets are launched which seek new routes in the system) or reactive (i.e. routes are found only on demand). Secondly, how the routing is calculated, either source routing (where the entire route is specified at the beginning of the journey of the packet) or next-hop routing (where the packet is simply forwarded to a neighbour). DSR is a reactive source-based protocol, AODV is a reactive next hop protocol, DSDV is a proactive next hop-based protocol. It is the protocols listed above (AODV,DSDV,DSR) that have driven the development of some peer-to-peer simulators. Examples of simulators that implement these protocols are [41] and [77], and a discrete event simulator is presented in [50].

The key factor in determining the effectiveness of a given routing scheme in conditions of change in the system is in terms maintenance of the proactive and on-demand routing information [17].

Evaluations of these protocols under different conditions of mobility have been conducted according to a number of different metrics. Throughput, latency and control message overhead are measured in [39]. Power and throughput are measured in [32]. Finally, packet delivery ratio, average packet delay and normalised routing load are measured in [38]. However, it is noted that many protocols are optimised for simplistic traffic patterns. Further, it is noted in [77] that conventional protocols are not designed for dynamic changes in the environment.

The ideas listed in the main routing protocols above are extended to cover a swarm routing mechanism described in [77] where a comparison of ant-based routing is made with the mechanisms listed above, and an argument for swarm-based routing being scalable, fault-tolerant, adaptive, and parallel is made.

2.5.3 Relevance of MANETs and peer-to-peer simulators to this work

MANETs as defined above, address the problem of routing between peers in dynamic environments, where the dynamism is mainly created by the movements of the nodes in the peer-to-peer network. In this thesis we are also concerned with the problem of routing between peers in a peer-to-peer system under dynamic conditions, which in this case is created by both bursts in traffic, or churn in the peer-to-peer system. From the literature we see that for MANET schemes there is a fine balance between the extent to which a routing protocol is proactive (i.e. goes out there and gets new routes) or is reactive (only searches for new routes when existing ones have proven to have failed, or some other criteria is fulfilled). We also see that many of these schemes are not scalable due to issues of central coordination. The ant-based protocol addressed in this thesis is an example of a self-regulating proactive routing protocol, which is scalable due to the decentralised properties of the routing decisions, and also the update mechanism of the routing tables. From the MANET literature we see that ant-based protocols are a promising direction in which to take MANET research.

2.6 AntHill and Bison

Extensions to the swarm routing paradigm in [77] can be found in a European project Bison. Bison, as described in [46, 47], is a large project focusing on the use of complex adaptive systems (CAS) in optimisation in peer-to-peer systems, by exploiting their properties of self-organisation, adaption, and resilience. Examples of CAS's are simulated annealing, neural networks, genetic algorithms, and ant systems. The goal of Bison is to try to understand why given CAS's perform well under certain application areas, and to apply this understanding in the development of controllers for these particular CAS's.

Anthill, as described in [46], is a run-time peer-to-peer environment based on JXTA[4], in which a collection of interconnected nests intercommunicate. each nest is a peer entity sharing its computational and storage resources. The interconnection network is characterised by a lack of fixed structure, and genetic algorithm routines can be used for tuning the parameters of the ant algorithms in order to optimise given parameters. The JXTA framework allows the use of different transport layers and communications.

The most relevant application for ants to our study is that of telecommunications routing. In [29] we see a swarm-based optimisation algorithm capable of finding paths of resources in a complex network environment where there is an overall good utilisation of network resources. Agents cooperate in the search when they have overlapping search profiles, thus minimising re-work. Further efficiency is gained by using the cross-entropy method for enhancing the search mechanism. This works by enhancing the probabilities of certain routes using importance sampling. Further, a performance function weighs the path qualities, so that high quality paths have a greater influence on the alteration of the pheromone matrix encoding the problem. Extensions to this method can be generated by distributing the processing, so that the evaluation of the fitness of the path is performed at each step in the ants tour. A suggestion for further work is made in this paper, of deriving a mechanism for the separation (in routing terms) of ants searching for efficient primary and secondary routes. We see the cross entropy (CE) approach use d in [49] for route management in telecommunication networks where an implementation of the CE routing algorithm is placed in a small test network carrying real traffic.[5]

A control parameter, temperature, is used in order to determ ine how much emphasis should be placed on the best paths. This mechanism works as follows. A low temperature tends to represent a stable system and therefore an emphasis on the current best solution found. As the temperature increases the probability that that the system will find other solutions increases. During the study a number of paths were found in the network by the ants prior to real traffic

[4] JXTA (Juxtapose) is an open source peer-to-peer platform created by Sun Microsystems in 2001. This platform is defined as a set of XML based protocols that allow any device connected to a network to exchange messages and collaborate in spite of the network topology. JXTA is a mature P2P framework currently available and was designed to allow a wide range of devices - PCs, mainframes, cell phones, PDAs - to communicate in a decentralised manner.

As JXTA is based on a set of open protocols, it can be virtually ported to any modern computer language and these implementations are called bindings. The Java binding is the most mature implementation up to now, and there is a C/C++ version, JXTA-C/C++, that is being actively developed as well.

JXTA peers create a virtual overlay network that allows a peer to interact with other peers directly even when some of the peers and resources are behind firewalls and NATs or use different network transports. In addition, each resource is identified by a unique ID, a 160 bit SHA-1 URN in the Java binding, so that a peer can change its localisation address while keeping a constant identification number.

[5] An explanation of the Cross Entropy method is as follows and according to [18], where we see there are broadly two phases to the method. 1. Generate a random data sample (trajectories, vectors, etc.) according to a specified mechanism. 2. Update the parameters of the random mechanism based on the data to produce a "better" sample in the next iteration. This step involves minimizing the Cross Entropy or Kullback-Leibler divergence.

Using this method, we can refine the parameter values of the distribution we are using to model the real data, in order to improve the sample next time round.

being added. The cost of these paths was measured, and then re-measured once the real traffic had been added. The cost metric was simply the sum of the delays on the link. However, each switch processed messages at a constant rate, and also it was assumed that the packets arrived according to a Poisson process.

The results show that the ant routing system is able to adapt the preferred path according to local traffic conditions, as evidence is presented for paths being switched in response to this.

AntHocNet is presented in [15] as an example of an ant-based routing scheme for MANETs. The main mechanism for updating the routing tables in the MANET scenario is the periodic sending out of hello messages, which re-establish presence information in the routing tables of the nodes in the MANET. Link failures are interpreted as failures in the unicasting of hello messages. Simulation results show that the average end-to-end delay for AntHocNet is half that of associated AODV schemes.

2.6.1 Relevance of AntHill and Bison to this work

The work of this thesis is close to the work in the Bison project, as the objectives of both endeavours are similar, namely to try to discover the conditions under which a given complex adaptive system performs well, and why this may be the case, in the context of peer-to-peer routing. However, the Bison project presents no systematic study of the effect of the driving parameter values on the routing behaviour of the Ant colony routing algorithm they use, instead a number of variants of the routing algorithm are proposed. This thesis addresses the effect of the change in the driving parameter values of the ant routing algorithm. In this thesis we explore this relationship from both a topological and traffic statistics point of view. The AntHill system described above is composed of a series of interconnected nests of ants. Ants perform tasks within the nests as well as intercommunication tasks. Genetic algorithms are used to tune the parameters of the nests in order to perform optimisation tasks. In this thesis we are concerned with the underlying dynamics of an ant peer-to-peer system, as such we choose a simpler model than the AntHill model where ants simply intercommunicate between the nests, and also do not use complex adaptive systems (such as genetic algorithms) for re-balancing individual nests within the ant colony, instead we use a simple set of driving parameters for the ant colony routing algorithm we implement.

2.7 Cognitive packet networks

The ideas of swarm routing have been further extended in the work on cognitive packet networks presented in [25, 26] in which a routing scheme using three packet types is proposed. The three packet types are smart, acknowledgement, dumb. Smart packets are sent out by nodes to retrieve information about the state of the network. Acknowledgement packets are returned from destination nodes to the source with the information about the state of the network requested in the smart packets. Finally, dumb packets are used to carry payload in the network. It is shown in [23] that using a neural network to learn the best way of routing packets using information feedback by acknowledgement of successful routing packets. In this study we see that the use of neural nets to find shortest path routes in a distributed network is possible, but further it is possible to use objectives other than shortest path (for instance longer paths) in order to minimise delay in congestion conditions. The observation is made [27] that routes do change as the system evolves, and also an analytical model is presented for the system. An alternative approach of using a genetic algorithm (GA) is documented in [24], where three goals (delay, hop count, and combination of delay and hop count) are optimised during route selection. It is found that the GA does reduce traffic loss rates at moderate network loads, but at high traffic loads the GA worsens the overall performance of the network. The explanation for this is that the dumb packets have an additive effect on the link traffic, which increases the observed delay and therefore end up reinforcing the wrong signals in the GA. A study of the time dynamics is performed in [21] where the average number of smart packets in order to find a valid path is plotted against the number of packets sent to the network. This analysis shows that after a threshold number of smart packets have been sent into the network there is no further advantage to be gained in terms of the routes found, by adding further smart packets in the network.

2.7.1 Relevance of Cognitive packet networks to this work

The work on cognitive packet networks presents contrasting evidence that complex adaptive systems in the form of genetic algorithms, and neural networks can be used in network routing schemes in order to optimise the performance of a network. However, the evidence in the case of genetic algorithms is that in some cases (namely conditions of high traffic load and therefore congestion), their inclusion worsens network performance. the cognitive packet network scheme works by the implementation of a feedback mechanism whereby information about the fitness of the routes in the network is fedback to the originating node in the network. In this thesis we concentrate on an alternative adaptive algorithm for the implementation of routing in peer-to-peer networks, namely the ant colony optimisation algorithm. The feedback mechanism we use in our ant based peer-to-peer system is similar to the one used in cognitive packet networks in that information about successful routes is fedback to the source node, and alterations in the routing table entries are made in order to minimise congestion.

2.8 Summary

In this chapter a review of the literature relevant to the study of self-organisation in peer-to-peer systems has been presented. Network rearrangements have been considered from the point of view of condensed matter physics, and also preferential attachment, and we have reviewed the instances where these models have been applied to real world networks. The general literature on self-healing and self-regulating systems has been reviewed where we see that the focus is on fixed architectures to support the efficient filtering of fault alarm information in order to precipitate accurate and timely corrective actions. Normally, these actions concern the efficient switching of routes to pre-planned protection paths. An examination of the resource demands and time to recover profiles of different fault recovery mechanisms in networks.

The Ant Colony Optimisation literature has been reviewed where it is revealed that although there have been many applications of the ACO meta-heuristic little is known about the underlying dynamics of ACO. Dynamic optimisation problems have also been identified as new areas of application for the ACO meta-heuristic.

The literature on AntNet was examined where a number of studies that concern the behaviour of systems running the AntNet routing scheme which show encouraging results in comparison to more common routing algorithms are presented. Many of these are performed on fixed topology networks. The literature presents an open question in terms of how an AntNet system would self organise under varying environmental conditions. Preliminary investigations of the use of AntNet in overlay design and optimisation have been conducted but these rely on assumptions about the relationship between the frequency of change in the environment and frequency of change in the system. The existing literature does not present a thorough investigation of effects of varying these frequency ratios.

When considering churn in peer-to-peer systems we see that the literature presents a number of open questions about how best to model churn in peer-ot-peer systems, and also that in certain cases failure and recovery behaviour can lead to a destructive feedback in the system which causes the system to flood and congest. The main observation is that the best way to prevent this feedback is to decouple the recovery rate from the rate at which failures are detected in the system. However, the literature presents no insight as to what the underlying dynamics of the decoupling process may be.

In the section on MANETs we saw that there are a number of categories of routing algorithm that are appropriate to routing in dynamic environments. It is noted in the section on MANETs, that many routing protocols are optimised for simplistic traffic patterns. We reviewed the Bison project, where studies have shown that ant systems are able to effectively respond to dynamically changing environments. These studies have been performed on fixed topology networks where simulations of congestion and network failure have been performed. We also saw examples of Ant based routing algorithms that have been implemented on real networks again of fixed topology.

The AntHill platform, that has been developed as part of the Bison project, is a simulation environment which allows the developer to install a variety of routing protocols within an ant-based peer-to-peer environment. The platform is based on JXTA allowing different transportation protocols to be inserted. Again fixed topology networks are used as the basis for simulation of the Ant systems. While these fixed topologies can be supplied in the form of XML files they do not evolve as the ants explore the system.

Further examples of the use of evolutionary algorithms in network routing applications have been examined in the work on cognitive packet networks. Here we see that using a simple range of packets (smart, acknowledgement and dumb) the routes that the system as a whole chooses change as the system evolves. However, there are cases where the use of evolutionary algorithms to learn the best routing schemes under given traffic conditions does not necessarily improve the performance of the overall network. This observation supports further the need for a clear understanding of the underlying dynamics of distributed peer-to-peer systems where the routing decisions are made using evolutionary algorithms.

2.8.1 Synthesis of the literature and motivation for experimental work

As described above, the literature presented in this chapter raises a number of open questions. We now combine these open questions into the following scientific questions which we intend the experiments in this thesis to address (either in full or in part).

- We wish to contribute to a better understanding of the underlying dynamics of the ant colony optimisation algorithm, by investigating how changing different parameter values that drive the routing calculations of the ant peer-to-peer system affect the behaviour of the ant peer-to-peer system.

- Specifically we wish to investigate the conditions under which complex adaptive systems work well or badly in the context of routing in peer-to-peer systems.

- We wish to investigate how self-healing properties of self-regulating systems such as ACO increase the reliability of the overall peer-to-peer system.

- We wish to investigate descriptions of netowk growth and evolution of topology in networks that are not dependent on underlying assumptions about the mechanisms by which new connections are added to the network.

- We wish to investigate the behaviour of complex adaptive systems in dynamic optimisation problems, such as optimal routing when nodes are entering and leaving the system.

- We wish to approach these scientific questions by altering the values of the driving parameters of our peer-to-peer system, in a systematic way, and gathering accurate experimental data to document the response of the system under each new condition.

Chapter 3

Theoretical concepts

3.1 Introduction

In this chapter we present the main theoretical ideas and concepts that underpin the work in this thesis.specifically we propose a number of metrics and analytica techniques that can be used to specify the organisational state of any peer-to-peer system at any point in its evolution. We model a peer-to-peer system by a graph generating algorithm, and further use ant colony optimisation (ACO) as the main mechnism for routing traffic within the ant peer-to-peer system.

3.1.1 Introduction to ant peer-to-peer systems

Peer-to-peer architectures are different from client-server arhitectures as each peer can act as both client and server, and every node is as important as every other node within the system as a whole.In client server systems thsi is not the case as there is an implicit hierachy (servers are more important) which guides the network topology. Because of this lack of centralisation peer-to-peer systems have several advantages over client server systems, some of which are:

- Due to its decentralised nature the peer-to-peer system is resiliant against attack or failure in a way that a client server system is not.
- The scaling properties of peer-to-peer systems are more favourable than client server systems.
- Because of the scaling properties of peer-to-peer systems they are more appropriate for the dissemination of large quantities of data to a large dstributed user community as it is easier to place copies of high demand files and data at disparate nodes in the system to improve redundancy and assist in load balancing the system.

3.1.2 Motivation

As stated above peer-to-peer systems have no cetral point of coordiation, which suggests that organisation in the system is due to two mechanisms, individually or in conjuntion.

- Load balacning internal to the system either preventing or responding to congestion.
- Symetries, corelations, or other persistent spatio-temporal pattenrs in the demand function.

We term these *consequential forms* of organisation.

Other forms of organisation may be imposed on the system in order to modify its operation, such as shutdown or partial shutdown. These are intentional forms of organisation rather than the form of organisation consequential to use as stated above. Both of the consequential forms of organisation cause traffic to orginate from and terminate at certain nodes in preference to others in the system.

The routing of trrafic we are considering within this peer to peer system in this thesis is based on a scheme that uses ant colony optimisation (ACO). The ACO algorithm is used here for several reasons. It is an adaptive routing algorithm that uses a small number of seed paramters to perform a highly efficieint parallel search through the routing possibilities of the network. The ACO algorithm has been shown to quickly identify effieint routes in telecommunication networks. Another feature of ACO is that of non-convergence, as backup, secondary routes are also created and maintained in addtion to primary ones. All these routes are continuously polled for their fitness and points of congestion as the algorithm iterates, and the probabilities for routing the traffic between nodes are adjusted accordiingly due to the parallel searching properties of the algorithm.

While there have been many areas of application for the ACO algorithm, little is known about its underlyinig dynamics. So, as well as serving as a good model for a general peer-to-peer system the analysis of the underlying dynamics of ACO is of interest in its own right.

The main aim of this project is supported by two further aims. Firstly, the description of the overall state of the peer-to-peer (or ACO) system, and secondly, the development of metrics that can specify how organised (or correlated) the system is at any point in its evolution. In order to do this we must first examine the behaviour of the peer-to-peer system under two categories of behaviour; at equilibrium, and at non-equilibrium.

3.1.2.1 Equilibrium behaviours

We are concerned here with the state that the system has reached after a large number of time steps have passed, and we describe the resultant topology of the system as it has evolved to cope with the demand that has been placed on it by the demand function. In equilibrium scenarios these is no change in the system imposed by either changes in the demand function or nodes entering and leaving the peer-to-peer system. We expectthat under these conditions after a long simuation time the system will have settled into a configuration that enables it to cope with the demand placed on it.

3.1.2.2 Non-Equilibrium behaviours

Here we seek to track the changes in the peer-to-peer system's overall configuration in response to changes in the environment. We measure the properties of the system in order to define the behaviour the system in response to a perturbation such as a spike in traffic demand or a disaster which knocks part of the system out. Specifically we are interested in the ability of the system to adapt to changing environmental circumstances. Examples of non-equilibrium behaviours are bursts in the demand function and also the entering and leaving of nodes in the overall peer-to-peer system. In measuring these properties of the peer-to-peer system under these non-equilibrium conditions we hope to track how the peer-to-peer system is able to adapt to changing conditions.

3.1.3 Chapter Structure and rationale

In this chapter we define a theoretical framework that describes the process of self-organisation within peer-to-peer systems and also propose a number of metrics and analytical techniques that can be used to specify the organisational state of a peer-to-peer system at any point in its evolution. We use ant colony optimisation (ACO) as the main mechanism for routing traffic within the ant peer-to-peer system. A graph is generated as a consequence of the routing of traffic within the peer-to-peer system. We can therefore, model the topological properties of the graph generated by the peer-to-peer system. We note however, that this method of modelling peer-to-peer systems as a graph generating algorithm is generally applicable irrespective of the method (or algorithms used) of routing traffic within the peer-to-peer system.

Large scale peer-to-peer systems are by their very nature complex systems to analyse and measure. The challenge in this work is to specify a model that is simple enough to be understood using well-defined analytical methods, while at the same time being flexible and rich enough to adequately reflect the diversity in behaviour of peer-to-peer systems in a range of real environments. This chapter has been divided into a number of sections which focus on different aspects of the work necessary to address this challenge.

- Section 3.2 - Background - describes the necessary background to consider the behaviour of a peer-to-peer system from different points of view.
- Section 3.3 - The theoretical framework - describes the motivation and rationale for the theoretical framework that has been devised to describe the self-organising system

and also treats the constituent components. Each component is defined and the main parameters governing their behaviours and the mechanisms for interactions between the components are described. We also focus on the the graph generating algorithm and mechanism for routing (ant colony optimisation) and explain some of its features.

- Section 3.4 - Metrics and measureables - concentrates on strategies for measuring properties of the peer-to-peer system. The important questions regarding the equilibrium, non equilibrium behaviours of the peer-to-peer system are elaborated upon and described in detail.

- Section 3.5 - Analysis - here we present a taxonomy of analysis and description of the behaviour of the self-organising system has been developed by understanding and applying techniques from graph theory, and condensed matter physics. We also describe a novel method for determining the extent to which a given graph is random, and a way of measuring the amount of structure in the graph. Further graph and network statistics are then presented to give a comprehensive view of the methods of measuring topological properties of a network, and the profile of the traffic flowing through it. We also indicate the theoretical ideas behind preferrential attachment in Section 3.5.4.

3.2 Background

In this section we set out the necessary background for considering the behaviour of a peer-to-peer system. We begin by giving a brief introduction to peer-to-peer systems and the way in which their properties contrast with other systems. We then consider some of the properties of ant colony optimisation, and specifically how to formulate network routing as an ant optimisation problem. Finally, we consider how routing in ant peer-to-peer systems would work.

3.2.1 Introduction to peer-to-peer systems

Peer-to-peer architectures are different from client-server architectures as each peer can act as both a client and a server, and every node is as important as every other node within the system as a whole. In client-server systems this is not the case as there is an implicit hierarchy (servers are more important) which often guides the topology of the system. In contrast in a peer-to-peer system there is no need for a central point of coordination. Because of this lack of centralisation peer-to-peer systems have several advantages over client-server systems, some of which are:

1. Due to its decentralised nature the peer-to-peer system is resilient against attack or failure in a way that the client-server system is not.

2. The scaling properties of peer-to-peer systems are more favourable than client-server systems.

3. Because of the scaling properties of peer-to-peer systems they are more appropriate for the dissemination of large quantities of data to a large distributed user community, as it is easier to place copies of high demand files and data at disparate nodes in the system to improve redundancy and assist in load balancing of the system.

3.2.2 Ant Colony Optimisation theory and components

In this section we focus on the ACO meta-heuristic. A good account of the ACO meta-heuristic is given in [18]. We must first define the terms *heuristic*, and also *meta-heuristic*. A heuristic is a method of obtaining good solutions to complex problems at relatively low computational cost without being able to guarantee the optimality of the solutions. There are two classes of heuristics, constructive or local search. For a description of local search heuristics the reader is referred to Chapter 2 of [18]. In general local search methods start from an initial solution and repeatedly try to improve the current solution by applying local changes. ACO is constructive and works by building solutions to combinatorial optimisation problems incrementally. Solution components are added together to generate the entire solution. The solution components are generated so that each one achieves maximum heuristic benefit as defined by the heuristic function.

A meta-heuristic (as defined in Chapter 2 of [18]) is a general algorithmic framework which can be applied to different optimisation problems with relatively few modifications in order that they be adapted to those specific problems. In the ACO meta-heuristic ants cooperate to find good solutions to problems by communicating using *stigmery*. Stigmery is defined as indirect communication mediated by the environment. In the case of ACO this manifests itself as the laying down of pheromone as ants pass through parts of the configuration space of the problem which they are working on. Favourable configurations are denoted by high levels of pheromone on the constituent links on the representative graph, indicating a high frequency of traversal of ants through those parts of the configuration space.

In general a combinatorial optimisation problem (Π) can be defined as a triple (S, f, Ω). S is defined as being the set of candidate solutions of the problem being considered. f is defined as the objective function which is to be maximised during the optimisation process. Finally, Ω is defined to be the set of constraints which separate out the feasible solutions (those that satisfy the constraints) from the infeasible solutions (those that do not satisfy the constraints). The overall goal for combinatorial optimisation problems is to find good solutions at minimum cost.

Each feasible solution will have a cost associated with it. For ACO to be applied the problem is represented as a graph (a set of nodes and edges). The feasible solutions are explored by the ants in the system, as the ants build solutions by performing randomised walks on the representative graph of the problem. Each ant has a memory which can be used to store information about the path it follows on the graph. This memory can be used to build feasible solutions, evaluate the solutions found, and also retrace the paths backwards to the source node of the ant. Moving from node to node on the representative graph is done by applying a probabilistic decision rule, where one of a number of possible next moves is chosen stochastically. Once the ant reaches its destination on the problem graph it then retraces its steps back to the source and lays down pheromone as it does so. The probability of an ant taking a given path through the representative graph is related to the amount of pheromone on the constituent links of the path. More efficient routes through the representative graph will be traversed more frequently by ants and will therefore have pheromone laid down on the constituent links more frequently. All of the pheromone on all of the links in the representative graph is subject to an evaporation (at each iteration in the simulation). Unless the pheromone on a link is renewed by the traversal of ants over it, the pheromone on that link will disappear over time and the link will effectively become deleted from the representative graph.

There are three stages to the workings of ACO. These are

- Construct Ant Solution — this is where the colony of ants concurrently and asynchronously explore other nodes on the connection graph. Movements are made by applying stochastic local decision policies.

- Update Pheromone — this is where the pheromone is updated due to the retracing of steps back towards the source by the ants, and also the evaporation of pheromone which is not renewed by the passing of ants. In this way the system is forgetting solutions that are not appropriate to the optimisation problem, and reinforcing those solutions which are appropriate for the optimisation problem being considered.

- Daemon actions — these actions perform housekeeping tasks such as the collection of global information for the purposes of producing aggregate statistics. However, this step can also be considered to be part of the specific simulation of the problem or system being considered.

Using the framework outlined in this section, ACO has been applied to a number of problems such as scheduling, and also routing (such as the travelling salesman problem). In the next subsection we focus on the specific application we are interested in here, that of network routing.

3.2.3 Network routing applications of ACO

The problem of network routing is one of finding minimum cost paths between nodes in a network. In real life network routing problems such as telecommunications routing, the cost of the paths may vary due to varying degrees of congestion on different parts of the network, or because the topology of the network varies. It is this variation in the cost of network paths that necessitates the use of heuristics. The problem of network routing, in the context of ACO, is formulated using the following components:

- Construction Graph — consisting of the set of nodes to be routed between and the set of links which connect the nodes.
- Constraints — the only constraint is that the ants only use the connections in the connection graph.
- Pheromone trails and heuristic information — each connection in the connection graph will have many pheromone trails associated with it. Each connection could have associated with it one possible value for each possible destination an ant located on the node at the beginning of the link can have. Each link is assigned a value that is the sum of all of the pheromone trails that pass over it.
- Solution Construction — each ant has a source and destination node, and moves from the source to the destination by hopping from one node to the next until the destination has been reached. At each intermediate node between the source and destination the next node is chosen using a probabilistic decision rule, which is a function of the ant's memory and local pheromone levels.

3.2.4 Routing in Ant Peer-to-Peer systems

Our ant peer-to-peer system is comprised of a number of nodes. Each node is given a coordinate on a two dimensional grid, and a radius of awareness, within which it is able to see neighbouring nodes. Each node also has a routing table which holds the pheromone levels of each (destination,first-hop) combination [1] that the node knows about. By this we mean that for every destination node that the node in question knows about, there are a number of first hop, intermediate, nodes that traffic could be routed to which would take the traffic closer to the destination. Initially the nodes search in their local vicinity for nodes that they can see, set up rows in the routing table populated with low seeding values of pheromone.

A proportion (between 0 and 100 percent) of the nodes in the system are then configured to be able to send out ants. Every node, however, performs a routing function irrespective of

[1] A destination-first-hop combination is the data structure that encompasses two node identities. Firstly, the ID of the destination node that the ant is ultimately supposed to reach. Secondly, the first-hop node that the ant is supposed to use on the first leg of its journey from its current node towards the destination node.

whether it can create ants and send them out in the first place. These ants are sent to random destinations within the radius of awareness of their source node. At each intermediate node the ant is entered into that node's queue. When it is the ant's turn to be processed, a look-up is performed in the routing table of the intermediate node and a next-hop node chosen from the entries in the routing table that correspond to the destination node in question. The probability of a given next-hop node being chosen depends on the pheromone level between the current node and the next-hop node being considered. The greater the pheromone level [2] of any given entry the higher the chance that it will receive further traffic, although this is not certain, as the traffic is allocated randomly according to pheromone level. When an ant reaches the destination it turns round and retraces its steps back to the source. As it retraces its steps, pheromone is deposited in the appropriate entry of the routing tables of the nodes that the ant has visited. The more efficient the route that the ant has followed, the more frequently this addition of pheromone will occur in the routing tables and the greater the reinforcement. As such, traffic tends to converge on the efficient routes. Payload in the network is carried within the ants. We can distinguish between routes in our network with large pheromone concentrations which will carry the majority of the traffic making them suitable for payload traffic, and those routes with low concentrations which would be suitable for sensing and control of the environment.

A decay process also runs which reduces the amount of pheromone on all of the routes in the network by a fixed proportion. The decay process runs every iteration as does the process of initialising and sending out ants from the nodes given permission to do so. Also every node processes all of the ants in its own queue every iteration. In so doing, we reinforce the strong routes, purge the system of routes that are not being reinforced, and also explore new potential routes in the system.

From the description above we see that the ant peer-to-peer algorithm is adaptive in its own right. By the continual reinforcement of useful routes, purging of weak ones, and exploration of new ones the algorithm can deal with changing traffic conditions and points of congestion and load. One other property assists in this which is *non-convergence* which ensures a number of candidate network solutions are available at one time. Each ant effectively does a search in a small subset of the network. This information is aggregated in the routing tables of the nodes.

The general mechanism for self-regulation is as follows:

1. The system produces a stimulus which signifies an imbalance of some kind.

2. The stimulus causes an action which rectifies the state of the system.

3. The stimulus is then removed once equilibrium is regained.

[2] The pheromone levels of each of the entries in the routing tables of the nodes in the peer-to-peer system will vary as we move from node to node. Within a specific routing table of a specific node the probability of ants being routed down specific routes is computed by comparing the relative values of the pheromone levels of the entries within the routing table in question

4. The action is then stopped.

ACO follows this mechanism for self-regulation, and the stimulus is indicated in terms of a pheromone level. This is best illustrated by an example. Consider the case of network routing where there are two alternative paths (path 1 and path 2) between a pair of nodes in a network. In the scenario where both routes are uncongested, the rate at which pheromone is deposited on both routes will be roughly equal. This means that the probabilities of routing ants down either route is similar, and the throughput of traffic down each will remain even and balanced. In the case where one of the routes becomes congested (e.g. path 1), then the rate at which ants pass through path 1 will decrease, meaning that the amount of pheromone deposited on path 1 will decrease, and therefore the probability of ants being routed down path 1 will decrease in comparison to routing down path 2. The result of this imbalance will be that a greater than equal proportion of ants will be routed down path 2, which in turn will lessen the traffic on path 1 and assist in the clearing of congestion in path 1. Once the congestion in path 1 is cleared, ants will begin to flow more smoothly through path 1 depositing pheromone on path 1 and therefore reinforcing the routing probabilities along path 1 in the routing tables of the nodes concerned.

The congestion example just described gives an indication of how a stigmery-based ant system is able to self-correct and return to equilibrium as a reaction to perturbation within it.

The main parameter for the ACO algorithm is the pheromone decay rate. If this is high, decay happens quickly and then the routing algorithm is selective (depth first search); if it is low, then decay is slow and the routing algorithm is not selective (breadth first search).

Little is known about the underlying dynamics of the ACO algorithm, as until now the focus has been on the convergence or otherwise toward global optima rather than how the topology of the solutions evolve as the processes specified in the ACO algorithm iterates. Meulea and Dorigo [45] compare ACO with stochastic gradient descent and derive results about the speed of convergence to a global optimum. Merkle and Middendorf [44] characterise the behaviour of ACO in terms of fixed points of the pheromone matrix and their divergent or convergent properties. In Stutzle and Dorigo [67] a proof is provided of the probability of convergence to a global optimum in relation to the amount of time that the algorithm is allowed to run and evolve. ACO has been applied to telecommunications routing problems [34] and has compared favourably with other routing schemes such as minimal routing and quality of service routing schemes.

In summary we know from the literature above that ACO has good convergence properties, and has been successfully applied to telecommunications routing problems. However, an open question is how that convergence to a good routing solution is achieved by ACO.

3.3 Theoretical Framework

In this section we consider the theoretical framework. We begin by considering the motivation for the theoretical framework. Section 3.3.2 then considers the theoretical framework itself, by firstly presenting a rationale for the framework and detailing some of the shortfalls of current models, and then describing the framework in detail by considering the components and mechanisms within it.

3.3.1 Theoretical Framework — Motivation

As stated above peer-to-peer systems have no central point of coordination, which suggests that organisation in the system is due to two mechanisms, individually or in conjunction.

1. Load balancing internal to the system either preventing or responding to congestion,

2. Symmetries, correlations, or other persistent spatio-temporal patterns in the demand function the system is subjected to.

We term these *consequential forms* of organisation.

Other forms of organisation may be imposed on the system in order to modify its operation, such as shutdown or partial shutdown. These are intentional forms of organisation, rather than the forms of organisation consequential to use as stated above. Both of the consequential forms of organisation cause traffic to originate from and terminate at some nodes in preference to others.

The main aim of this thesis is to explore self-organisation in peer-to-peer systems which by necessity have large numbers of degrees of freedom and huge configuration space. We model a peer-to-peer system as a graph (set of vertexes and edges) which indicates the connections between the nodes in the system.

The routing of traffic within the peer-to-peer system we are considering in this thesis is based on a scheme that uses Ant Colony Optimisation (ACO). The ACO algorithm is used here for several reasons. It is an adaptive routing algorithm that uses a small number of seed parameters to perform a highly efficient parallel search through the routing possibilities of the network. The ACO algorithm has been shown to quickly identify efficient routes for applications in telecommunication networks [29, 49]. Another feature of ACO is non-convergence, as backup, secondary, routes are also created and maintained in addition to the primary routes for the system [18]. All these routes are continuously polled for their fitness and points of congestion as the algorithm iterates, and the probabilities for routing the traffic between nodes are adjusted accordingly due to the parallel searching properties of the algorithm.

As well as serving as a good model for a general peer-to-peer system the analysis of the underlying dynamics of the ACO algorithm is also of interest in its own right.

The main aim of this project is supported by two further aims. Firstly, the description of the overall state of the peer-to-peer (or ant colony) system and secondly, the development of metrics that can specify how organised (or correlated) the system is at any point in its evolution. In order to do this we must examine the behaviour of the peer-to-peer system under two categories of condition: the system at equilibrium, and the system at non equilibrium.

The following sub-sections focus on equilibrium and non-equilibrium behaviours respectively.

3.3.1.1 Equilibrium Behaviours

We are concerned here with the state that the system has reached after a large number of timesteps have passed, and we describe the resultant topology of the system as it has evolved to cope with the demand that has been placed on it by the demand function. In equilibrium scenarios there is no change in the system imposed by either changes in the demand function or nodes entering and leaving the peer-to-peer system. We expect that under these conditions after a long simulation time the system will have settled into a configuration that is capable of coping with the demand placed on it. However, the configuration may not be the global optimum.

3.3.1.2 Non-Equilibrium Behaviours

Here we seek to track the changes in the peer-to-peer system's overall configuration in response to changes in the environment. We measure the properties of the system in order to define the behaviour of the system in response to a perturbation such as a spike in traffic demand or a disaster which knocks part of the system out. Specifically we are interested in the ability of the system to adapt to changing environmental circumstances. Examples of non-equilibrium behaviours are bursts in the demand function and also the entering and leaving of nodes in the overall peer-to-peer system.

3.3.2 The relationship between the Theoretical Framework and preferrential attahcment

As stated in the introduction, we have modelled the behaviours of a general peer-to-peer system in a general theoretical framework. The relationhsip between the theoretical framework and the mechanism of preferential attachment is simple. We can view preferential attachment to be an instance of a graph generating algorithm within the general theoretical framework presented here. However, even consideration of the preferrential attachment mechanism (as a model for graph generation, as described in Section 3.5.4) reveals further shortfalls.

- The preferential attachment framework applies only to networks which grow in terms of the number of nodes within them. In our situation we are considering a peer-to-peer system with a constant total number of nodes available to it.

- The preferential attachment framework provides no way of relating the behaviour of the overall system to the mechanisms by which traffic is routed within the peer-to-peer system. By considering the framework below we are able to relate the measurements we make of the behaviours of the system to the values of parameters which would drive the routing decisions at the individual nodes. In the specific case of our ant peer-to-peer system using ACO as the traffic routing mechanism these would be the pheromone decay rate, range of awareness, and also percentage of nodes sending ants in the system.
- The outcome of growing a network using the preferential attachment framework is that a scale free network is produced. This is a prescribed solution, which is not flexible enough to be able to model the varying forms of degree distribution resulting from the self-organisation within a peer-to-peer system. In other words we need a model which is adaptable enough to model graphs which are not scale-free in nature.

This framework is described in Figure 3.3.2. We can see that there are a number of components and also interactions between the components. These are described below.

3.3.2.1 Components of the Theoretical Framework

There are four major components within the framework. All but two of the major components have sub components within them.

- Component 1: *Short Term Demand Regulation*. This is where the congestion control and the goals for the desired topology are defined. This component has three sub components.
    - The graph generating algorithm — this component is responsible for the evolution and maintenance of the connections between the nodes in the system. In this case the graph generating mechanism is a routing mechanism based on ACO.
    - The measureables — this component is responsible for measuring the outcome of the evolution (in this case characterising the topological properties of the graph).
    - The parameters — these are the control variables that alter the evolution of the system.
- Component 2: *Long Term Structure Regulation*. This is the mechanism for pruning the configuration space which defines the possible network topologies from which the system can choose to meet any given demand. This is self-organisation at the meta level (self-organisation of self-organisation). The major imperative here is to prevent the system from adopting highly inefficient, brittle, or fragile topologies or configurations

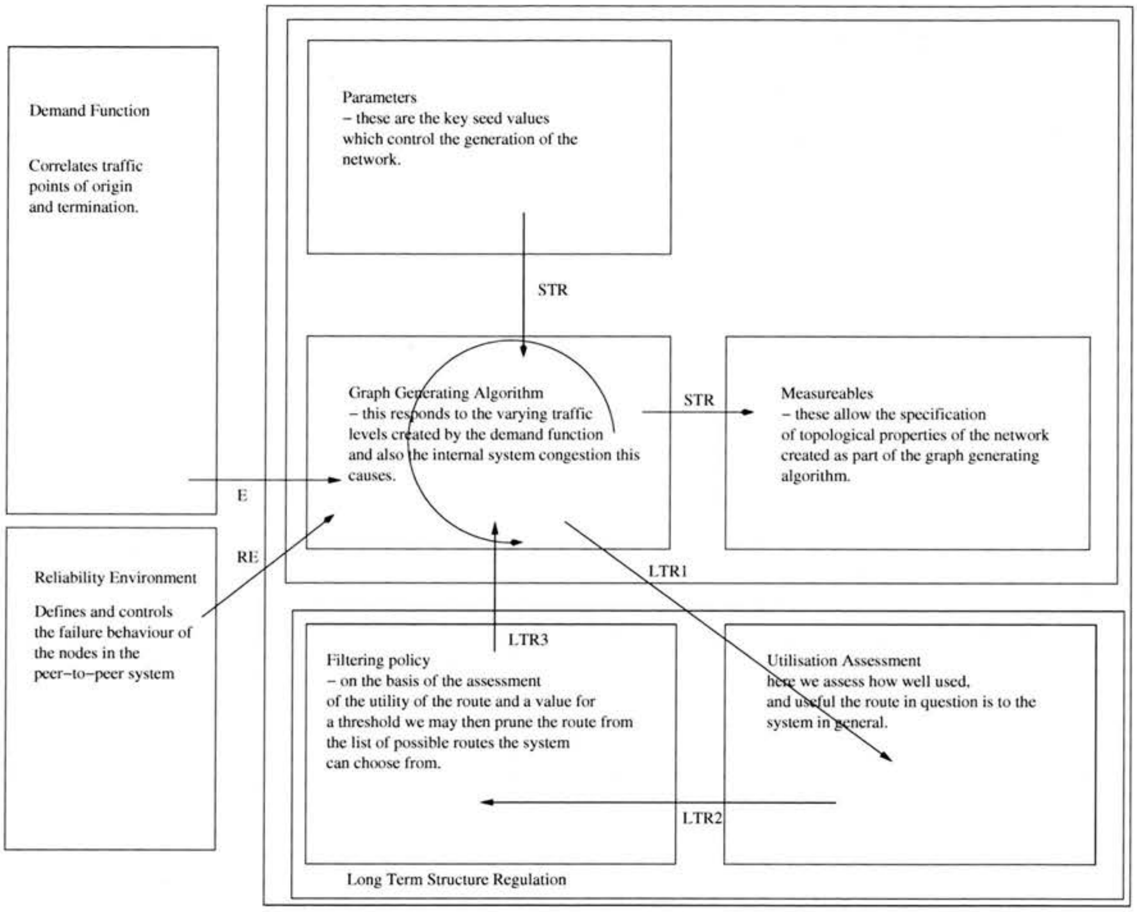

Demand Function
Correlates traffic points of origin and termination.
Parameters
– these are the key seed values which control the generation of the network.
STR
Graph Generating Algorithm
– this responds to the varying traffic levels created by the demand function and also the internal system congestion this causes.
STR
Measureables
– these allow the specification of topological properties of the network created as part of the graph generating algorithm.
E
RE
Reliability Environment
Defines and controls the failure behaviour of the nodes in the peer–to–peer system
LTR1
LTR3
Filtering policy
– on the basis of the assessment of the utility of the route and a value for a threshold we may then prune the route from the list of possible routes the system can choose from.
Utilisation Assessment
here we assess how well used, and useful the route in question is to the system in general.
LTR2
Long Term Structure Regulation

Figure 3.1: Theoretical framework for self-organisation in peer-to-peer systems

which leave the system with no means of coping with an evolving environment. This mechanism works in a slow time scale in comparison to Component 1, taking many iterations to ingrain the efficient routes and eradicate the inefficient ones. It has two sub components.

  - *Utilisation Assessment* — here we assess how well used and useful the route in question is to the system in general.
  - *Filtering Policy* — On the basis of the assessment of the utility of the route it will either be included or excluded from the configuration space of the system.

  These effects have been noted in [76] where evidence for the effect on the resulting topology of the graph created by the overall peer-to-peer system of routing table entry decisions made at the node level can be found. Specifically it is shown that by using different replacement policies, the hit rate, clustering, and path length of the FreeNet peer-to-peer system can be crafted to those of a small world network. Standard FreeNet does exhibit small world characteristics, but these can be degraded by a least recently used caching policy. In this instance we can view the least recently used caching policy as an example of component 2 of our theoretical framework, namely Long Term Structure Regulation.

Components 1 and 2 are both internal to the system. They act over different time scales, but they can both reflect different aspects of the system's ability to self-organise.

- Component 3: *External Environment*, which by definition is external to the system, this is modelled by a demand function. This is the mechanism by which load is placed on the system, and the traffic is generated. Typically this is a probability distribution which details the probabilities that traffic passes between any two nodes in the peer-to-peer system.

- Component 4: *Reliability Environment*, which defines the reliability properties of the nodes constituent in the peer-to-peer system. Typically this is composed of a reliability model which calculates the probabilities of the nodes in the peer-to-peer system being in an operational state given environmental factors. Examples of studies of reliability environments in peer-to-peer systems are stated in detail in the literature survey and include [12] where studies of the Gnutella peer-to-peer system have shown that the availability of the nodes within the system oscillate in a given 24 hour period, and also [66] where further studies of peer lifetime in Gnutella, BitTorrent, and Kad have indicated that session times follow a power law distribution. Theoretical models of peer lifetime have included modelling using a Pareto distribution [37] which is used to model a range of peer lifetimes and to incorporate these in the model.

3.3.2.2 Mechanisms within the Framework

As we can see from Figure 3.3.2 there are interactions between the components of our theoretical framework. These interactions can be seen as components of mechanisms (or in some cases whole mechanisms) which describe the ways in which the different components of our framework can affect one another. The interactions and mechanisms are as follows:

1. E - The mechanism for exchanging information with the external environment is composed of just one interaction. Here information about the demand placed on the system is passed to the graph generating algorithm.

2. STR - The mechanism for Short Term Regulation. This is the mechanism by which all of the sub-components within our short term regulation component exchange information. Necessarily the order of the interactions are as follows:

    (a) Graph generating algorithm adapts to its own internal congestion state, and the external demand function.

    (b) The resulting graph state is then measured by a set of specific measureables.

3. LTR - Long Term Regulation, this is the mechanism by which the long term structure of the system is decided, and encoded. This mechanism is composed of three interactions.

    (a) LTR1 a snapshot of the system is taken, and the routes in the system are analysed for their usefulness or fitness. This could be last time used, or frequency of use, or the proportion of total traffic carried by the route in question, for example.

    (b) LTR2 a filter is applied to the routes according to some criterion.

    (c) LTR3 the weak routes are eradicated from the system.

4. Parameters - this is the main mechanism for initialising the self-organisation process of the peer-to-peer system. Here we specify the initial conditions from which the peer-to-peer system will evolve.

5. RE - Reliability Environment, this is the mechanism by which the reliability of the nodes in the peer-to-peer system is modeled. The reliability model includes a specification of the operational, partially operational, and non-operational states of the nodes in the peer-to-peer system, and also the conditions under which transitions between these states are made.

3.4 Metrics and Measureables

In this section we consider how the system can best be measured. We begin by considering the relationship between equilibrium and non-equilibrium behaviours of the system. We then consider, in Section 3.5, some of the standard analytical techniques people use to measure the properties of networks and graphs. These standard techniques are then built upon to create a proprietary measure of the amount of randomness in a graph. This is described in Section 3.5.2, and is original work constituting part of the contribution of this thesis. Finally, further background regarding graph and network statistics is presented in Section 3.5.3.

3.4.1 Approach to the measurement of the peer-to-peer system

Many aspects of system behaviour need to be examined and incorporated into the system model in order to ensure that it adequately represents the system. As stated above these fall into two broad categories: equilibrium and non-equilibrium behaviours.

3.4.1.1 Equilibrium Behaviours

Here we consider the topology of the system, where the major and minor connections are in the system, and what the most efficient way of routing information between parts of the system is. We model this primarily by using graphs to describe the connections between the nodes, where each connection has variable capacity. These topological solutions are reached by the system after a settling period which allows the system to configure itself. The key questions that the model and simulation experiments need to address is:

- What is the distribution for the number of connections between the nodes in order to construct a network which has adequate redundancy in routes and also small path length? In the context of equilibrium behaviours we can test for adequate redundancy, by considering the behaviour of the peer-to-peer system in response to congestion at peer nodes caused by constant levels of demand, ewven when this demand causes a significant proportion of the nodes in the system to be in a congested state.

- What is the average path length of this topology and how does the average path length relate to the starting configuration of the ACO algorithm?

- How successful is the system at routing information between the source and destination (what proportion of the total packets sent arrives successfully at its destination)?

The capacities of the edges in the graph describing the peer-to-peer system do vary, according to the propensity of traffic to use a given edge in the graph [3]. This is of significance when considering the response of the system to change. As damage or congestion on routes which are of greater capacity will have greater consequences for the system as a whole, than damage or congestion on lower capacity routes. The ability of the system to adapt to changes

[3] In the context of our ant peer-to-peer system this is reflected in the variation of pheromone levels on edges in the graph. These pheromone levels in turn translate into variations of probabilities that traffic is routed along a particular edge in the graph.

in the demands placed on these high capacity routes will be disproportionately important in determining the overall response of the system to change. In the next section we consider the non-equilibrium aspects of system behaviour.

3.4.1.2 Non-equilibrium Behaviours

Non-equilibrium behaviours consider the response that the system makes to change in the external environment. These are also useful in considering the recovery of a system when knocked off balance by an external event such as a spike in traffic demand or failure of a large proportion of the nodes within the peer-to-peer system.

However, in order for the peer-to-peer system to continue to function this change must be productive and lead to a stable equilibrium distribution mentioned above. If this is not the case the peer-to-peer system will spend a great deal of time changing configurations, and may not be able to function efficiently as a result. The main questions considered are listed below:

- Does the system configuration (in terms of the degree distribution, average path length, and also the proportion of ants successfully reaching their destination) converge at all?
- How quickly does the system converge to an equilibrium distribution? Equilibrium can be determined by calculating a measure of change in the configuration of the peer-to-peer system from one simulation iteration to the next. If this measure is small then, we know that the system configuration has converged.
- What is the effect of the main control parameters on the ability of the system to maintain an optimum in this non-equilibrium situation. By the main control parameters we mean the pheromone decay rate, the initial range of awareness of the peers, and also the percentage of the nodes able to initiate and send out ants rather than simply route them.

By modelling the system at non-equilibrium we can also examine the system behaviour in the case of a perturbation or change in environmental circumstances. The main measureables here would be degree distribution and path length, and also the proportion of the ants successfully returning to their source nodes. In this way the non-equilibrium behaviour is intimately related to the property of system robustness. Exploration of how and if perturbation (both in terms of change in the demand function and also failure in the nodes of the system) causes the system to move into a stable state is key.

3.4.1.3 Prioritisation of key questions

The key questions above can be prioritised as follows. First we must identify the behaviours of the peer-to-peer system under constant environmental conditions. The identification of these behaviours or configurations will enable separation of the effects of changes in the environment from changes of the internal configuration of the system (in this case the abilities of the nodes to route traffic) due to congestion or node failure when later analysing the non-equilibrium behaviours.

3.5 Background Analysis

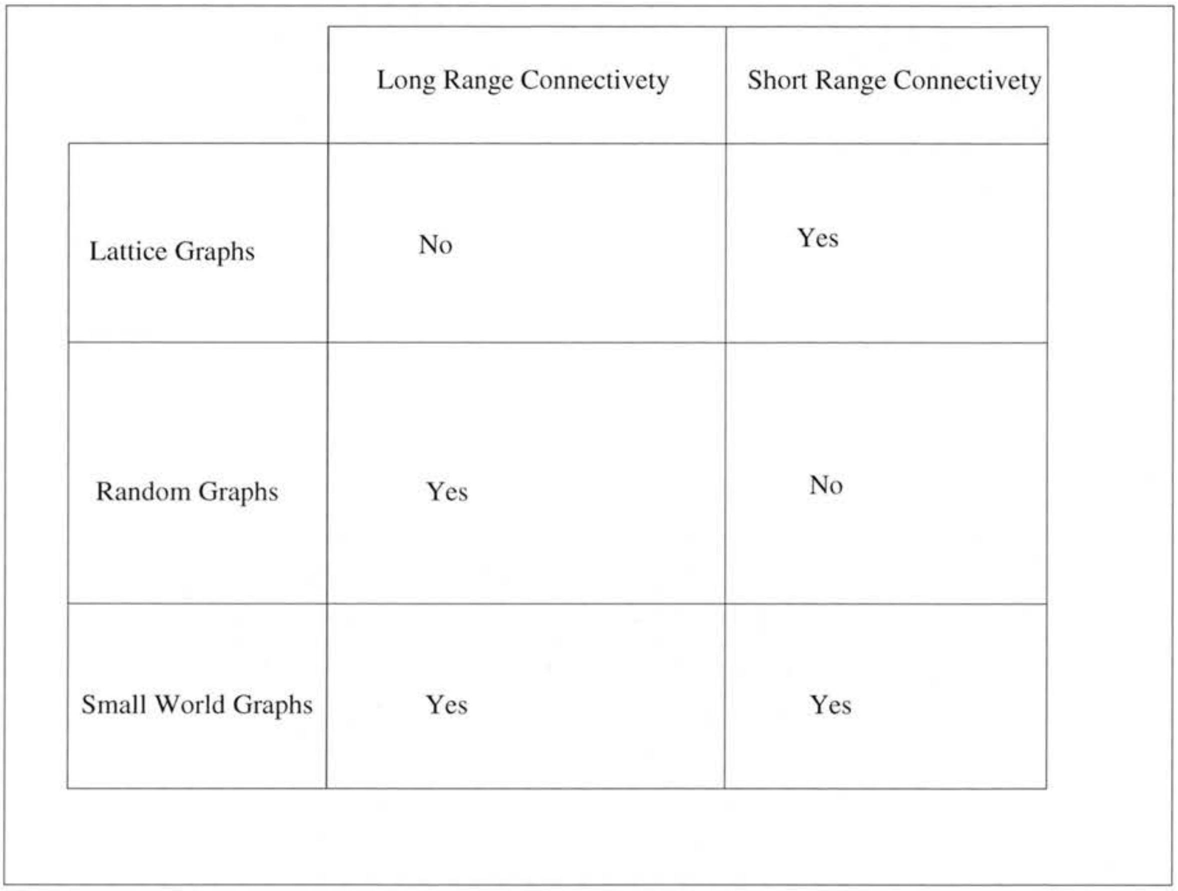
In this section we present the main analytical techniques we use to track the evolution of the peer-to-peer system. We consider first the meaning of the degree distribution of a graph and the categories of the graphs that can be generated. We then consider how a degree distribution can be analysed. Related measurements for the description of the behaviours of peer-to-peer systems are then presented.

3.5.1 Degree Distribution Signatures and Graph Topologies

Graphs can be categorised according to a number of different criteria with differing connection properties. Each class has a characteristic degree distribution. The diagram in Figure 3.2 indicates that the major classes are as follows.

Firstly, we have *lattices*. These graphs have regular connection properties where the topology surrounding each node is the same. In these graphs the local connections with surrounding nodes are efficiently arranged. The degree distribution of these graphs are Kronecker delta functions at a given degree value. An example of a lattice would be a two dimensional grid. Secondly, we have *random graphs* that are constructed by forming random connections between nodes. In these graphs there is the possibility of long range connections allowing efficient transportation between one area of the graph and another. The degree distribution of these graphs is Poisson. Thirdly, we have *small world networks* [51]. These graphs are special as they have super-connecting properties. This means that they combine both the efficient local connection properties of lattices, and efficient long range connections of random graphs. As such, these small world graphs lie between lattices, which have order, and random graphs, which have disorder. It has been shown that small world graphs result from social and cooperative processes such as labour markets [68].

As our graph emerges we can characterise the topology by looking at its degree distribution, and inspecting how it changes as the system evolves. Standard results for the degree distributions of scale-free, lattice and random networks have been derived [1, 33]. We can also look at the spectral density of the eigenvalues of the graph connectivity matrix. By comparing the degree distributions and the eigenvalue density of our evolving graphs with these standard results for random graphs and lattices we can examine the patterns that are emerging in the graph of our self-organising peer-to-peer system. Moreover, we can also measure how random or organised the graph is as a whole or in part. By using this method, the scale-free compo-



	Long Range Connectivety	Short Range Connectivety
Lattice Graphs	No	Yes
Random Graphs	Yes	No
Small World Graphs	Yes	Yes

Figure 3.2: The Universe of Possible Graphs

nents of the world wide web have been measured and the mechanism of *preferential attachment* has been examined [6]. The directed nature of the connections in a peer-to-peer system can be accommodated by allowing for separate in-degree and out-degree distributions. A further analysis can then be performed by measuring the preferential attachment probabilities of both independently. However, this approach is only valid on the assumption that the preferential attachment mechanism is valid, and is only relevant to graphs and networks which grow.

3.5.2 Degree distribution analysis — The meaning of the Poisson distribution, and the measurement of randomness

It is known that random graphs can be characterised by degree distributions which are Poisson in form. A metric that could measure the difference between the degree distribution of our ant system and the related Poisson distribution would enable us to identify the points in our configuration space where the least amount of randomness (or greatest amount of structure) is generated by our ant colony system. In Figure 3.3 below we indicate an analysis process for doing this, and we also define such a metric. We describe the analysis process here. From Figure 3.3 we see that there are two analysis paths, path A and path B. We derive the distance measure by comparing the outcomes of analysis paths A and B.

In path A we take the raw degree distribution data and model it using a Gaussian Mixture Model[4], which allows us to smooth across the multiple runs of the system. In Path B we generate the comparison data by taking the raw degree distribution data, calculating its mean value, and then generating the associated Poisson distribution. The significance of the Poisson distribution is that this is the degree distribution signature of a network with the same average degree which has no structure and is entirely random.

We can measure the distance between the two distributions produced by path A and path B respectively. This is done by forming a symmetrised version of the Kullback-Leibler divergence between the two distributions. We explain this here. The Kullback-Leibler divergence between two probability distributions $R(x)$ and $S(x)$ is defined as follows:

$$D_{KL}(R \mid S) = \sum_x R(x) log \frac{R(x)}{S(x)}$$

This is an asymmetric quantity as $D_{KL}(R \mid S)$ is not the same as $D_{KL}(S \mid R)$. In order to form the distance we can compute the sum of the two divergences, so we define the distance Δ as

$$\Delta(R,S) = D_{KL}(R \mid S) + D_{KL}(S \mid R)$$

[4] A Gaussian Mixture Model is a method of expressing any function as a weighted sum of gaussian functions with individual values for the mean and variance

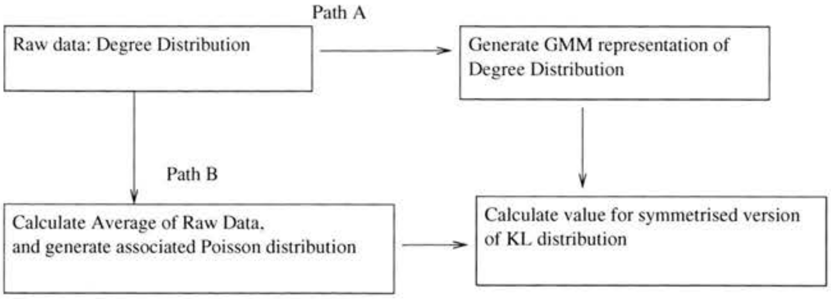

Path A
Raw data: Degree Distribution
Generate GMM representation of Degree Distribution
Path B
Calculate Average of Raw Data, and generate associated Poisson distribution
Calculate value for symmetrised version of KL distribution

Figure 3.3: Statistical Analysis of degree distribution

There is also one other subtlety that we must deal with, which is the problems of zero's and infinities. If we consider the definition for D_{KL} if the quantity $S(x)$ falls to zero then the value for our divergence metric will become infinity. This is meaningless, and we need to avoid this in order to be able to effectively use the metric. We do this by constructing the distance between our two distributions from the addition of two components, namely the distance of each distribution from its mid point. So if P_{mid} is the average of our two distributions, namely;

$$P_{mid} = (R(x) + S(x))/2$$

we can refine our distance metric to be

$$Dist_{total} = \Delta(R, P_{mid}) + \Delta(S, P_{mid})$$

Note that this metric conforms to the triangle inequality, and is symmetric, so can be considered to be a distance.

We can use this metric to compare the degree distributions of our raw data and the associated random graph. In general, we know that if our D_{KL} value is small then there is great similarity between the degree distribution of our random graph and the degree distribution of our raw data. Consequently, we know that the ant colony system is not particularly organised. However, if the reverse is true and our D_{KL} value is large then we know there is significant difference between our degree distributions and our system is substantially organised (i.e. far from random).

In principle this method can be used for both the in-degree and the out-degree of the nodes in the peer-to-peer system. As we are considering the onward routing of the ants from source to destination and back to source again, we are concerned here with the out degree of the nodes.

3.5.3 Graph and Network statistics

In this section we consider further metrics for measuring both the topology and performance of a peer-to-peer system. These fall into two broad categories: graph statistics, (which enable the measurement of the topological properties of the network), and network statistics, which reflect how effectively the peer-to-peer system is handling the traffic that is passing through it.

- Graph Statistics - these statistics are direct measurements of the topology of the resultant graphs.

  1. The degree distribution can be specified for both inbound and outbound connections within the graph plane. However, specifically, we are concerned here with the out degree distribution — see the section on degree distribution analysis.
  2. Path length is defined as the average number of node transitions (or hops) that need to be undertaken for a successful tour.
  3. Distribution of path lengths of the graph. Here we are measuring the average number of hops between nodes, and the proportion of the total nodes that lie within hop lengths.
  4. Clustering coefficient. A highly clustered graph has a large proportion of nodes which are connected to each other, and therefore a low characteristic path length.

     The cluster coefficient of the network indicates the degree of independence of the connections between the given nodes.

     $$\gamma = \frac{E(\Gamma(v))}{E_{max}(\Gamma(v))}$$

     where Γ is the neighbourhood operator which defines the set of nodes directly adjacent to the node in question and E is the number of edges between a given set of vertexes, and v is the vertex in question. $E_{max}(\Gamma(v))$ calculates the maximum number of possible edges between the vertexes in the neighbourhood being considered, i.e. N(N-1)/2 for non-directed graphs and N(N-1) for directed graphs, where N is the number of nodes in the neighbourhood.

     γ, the cluster coefficient, ranges from zero to one: If γ is zero then all the nodes in the graph are independent of one another; and if γ is one then every node is connected to every other node.
  5. Clustering coefficient distribution — this is the distribution created by measuring the cluster coefficients in the nodes in the network. By considering this distribution we can see whether there are a small number of nodes with many interconnections, and a large number of nodes on the periphery, or many of the nodes are highly interconnected. The clustering coefficient distribution measures the degree of mutuality of the connections between the nodes. That is, the set of neighbour nodes in many of the nodes in the system is similar.
  6. The diameter of the graph, L, is the largest path length in that graph.

The graph statistics mentioned above are applicable to the analysis of any graph or network topology, and are general in nature. This is in contrast to Network Statistics detailed below which are specific to the problem being addressed in this thesis.

- Network Statistics - these statistics are the direct measures of the traffic characteristics of the peer-to-peer system.

  1. The workload on the system can be specified in terms of the number of ants sent, as this gives an indication of the total amount of traffic that the system has had to process.
  2. The efficiency of the system can be measured in terms of the number of ants successfully completing a tour (i.e. a round trip from source node to destination node and back again), as this gives an indication as to how successful the internal routing scheme of the system is.
  3. The loss rate of the system can be measured in terms of the number of ants dropped [5]. Clearly if there is a high loss rate in the system then the efficiency of the system is reduced.
  4. The latency of the system can be measured in terms of the average number of hops the ants have made between their source and destination nodes, as each hop indicates a delay in the time it takes an ant to get from source to destination.

3.5.4 A note on preferential attachment

Preferrential attachment has been discussed several times several times in this chapter. In this section we give an account of the underlying theoretical model of preferrential attachment. The evolution of a graph which evolves according to the preferential attachment model can be specified in terms of preferential attachment probabilities. The probability $P(K)$ that a given vertex in a network is connected to K other vertexes decays as a power law following:

$$P(K) = bK^{-\gamma}$$

Preferential attachment is defined in the following manner, as the network grows and new connections are formed we see a reinforcement model forming:

$$P_i(K_i) = \frac{K_i}{\sum_j K_j}$$

where i is the index of the node in question, and j is a variable which ranges over all of the indices of the nodes in the graph in question.

[5] An ant is dropped from the system undder two conditions only. Firstly, the ant arrives at a node whose queue is full. Secondly, the ant arrives at a node when the node is in a non-operational state.

So, the greater the number of connections a given node has the greater the probability that more connections will be added to the given node. Preferential attachment probabilities have been measured in real networks (i.e. citation, collaboration, and the Internet) [30]).

3.5.5 A note on the connection between condensed matter physics and topological rearrangements in networks

There is a relationship between the topological rearragements in netowrks and condensation processes such as Bose Einstein condensation. This model purports that there are three distict phases that a system adopts.

- *Winner takes all* phse in which the entire system is connected through a small number of nodes which have the majority of vertices in the network attached to them.
- *Scale-free* phse in which the structures that are adopted by the system are small-scaled and are replicated and assembled ever more nureoursly in order to accomodate growth in the system.
- *Fit-get rich* phase in which small discepancies in the fitnesses of the nodes compund ending up with a medium number of well-defined regions in the graph which are individually highly clustered.

subsectionThe relationship between macroscopic and microscopic degree distributions By considering the degree distribution we can infer the probability rule for preferential attachment being displayed at the individual nodes. This analysis is done by considering how rewiring one of degree K to an alternative destination node, affects the degree of the source node, the original detination node of the link, and the final destination node of the connection. Note that this analysis is contingent on the assumption that all edges iin the graph are equivalent, being valid. The process of re-wiring the network can be likened to a process of distributing balls in boxes. By imposing the condition that the overall degree distribution of the network remains the same over time we can find a stationary from the difference equation that the analysis produces. Also by choosing constants in our equation properly the relationship between prefential attachment and the overall stationary degree distribution of the graph produced is as follows:

$$f(K) = \frac{(K+1)P(K+1)}{P(K)}$$

where f(K) is the degree distribution of the individual node, and $P(K)$ and $P(K+!)$ refer to the overall degree distribution of the network.

3.6 Emergent behaviours

At the begining of the simulation runs, the nodes are placed on a two dimensional grid. As the average desity of the nodes throughout the grid is constant the average number of neighbouring nodes each node in our peer-topeer system has is also constant. This means that initially the routing tables of the nodes will be the same size. Similarly the queu lengths of the nodes in the peer-to-peer system are set at constant length Initially then, each node will be created equal, with a given amount of resource. However, as the peer-to-peer system evolves to cope with the demand placed on it, the doles of its contstituent nodes will need to change to accomodate this. The change in node role is reflected in the way the resources of the nodes are deployed. These reources can be used to either store information about routing in a routing table, or store ant traffic in queues. As the simulation proceedes that roles of the nodes will change.The trade-off between nodes which hold trafic information, or routing information will define a series of speicialisations. Nodes whose queues are perpetually full and which have small routing tables will have evolved to hold large numbers of ants, and will perform a traffic flow regulation role by capping the amount of traffic flowing through the system. Nodes with large routing tables will perform a routing role, and will enable the system to be aware of different parts of itself. There may also be some multi-purpose nodes which perform a hybrid role. At each iteration of the simulation all the nodes process all the routing requests of the ants that have accumulated in their queues and send them on to the next destination or terminate them if the node in question happens to be the destionation node. Ants that arrive at a node when the nodes ant queue is full are dropped and deleted from the simulation, and therefore do not contribute to reinforcement of any of the routes.

3.7 Conclusion

In this chapter we have developed a framework for the measurement of graph properties of a network as it evolves. We have also determined that we wish to generate small world graphs as they have favourable connection properties. We have also derived a method of measuring the similariy of two different networks. We note that preferential attachement can be derived from a statistical mechanical analysis of networks. Further we see that the mechanism of preferential attachment is applicable only under a limited number of circumstances. We also note that due to information exchanging mechanism of ACO the information in the routing tables of the system will continue to grow and shrink as the sysem evolves. The motivation behind the development of this simulator and also the experiments detailed later in this thesis is to investigate how the topological properties of the peer-to-peer network can be varied by changing the values of parameters governing emergent behaviour in peer-to-peer systems under conditions of both equilibrium and non-equilibrium.

Chapter 4

Ant peer-to-peer simulator

4.1 Introduction

In this chapter we focus on the ant peer-to-peer simulator that has been developed during the course of this work. Supporting material such as the definitions of file formats and the meanings of the fields are detailed in the associated Appendix A and referenced in the appropriate sections of this chapter. Further the results of validation tests between release 1 and release 2 of the simulator are documented in Appendix B.

4.1.1 Chapter structure

In order to support the focus of this chapter stated above, we have divided the chapter into a number of main sections. In Section 4.1.2 we explore the motivation for the development of the simulator, and the rationale for two releases of the simulator. In Section 4.1.3 we identify the reasons for developing an ant peer-to-peer simulator of our own. Section 4.2 is dedicated to the description of the main functions and control mechanisms of the release 1 of the simulator. Also in Section 4.2 we detail the main inputs and output files of the ant peer-to-peer simulator. Supporting information on the exact file formats of the input and output files for release 1 is detailed in Appendix A. Section 4.3 is dedicated to the description of the design of release 1 of the simulator by documenting the static application architecture, main control loop, and main ant processing routine for release 1 of the simulator. In Section 4.4 we examine in detail the reasons for the enhancement of the simulator. In Section 4.5 we focus on the functionality and control mechanisms for release 2 of the simulator, and we contrast these with those of release 1 where applicable. We also examine the main input and output files of release 2 of the simulator with main input and output files being described and supporting file formatting information detailed in Appendix A. The design for simulator release 2 is documented in Section 4.6 where the main differences in application architecture, main control loop, and ant processing, with respect to release 1, are identified. Section 4.7 deals with some of the implementation issues that arose on the project. Finally Section 4.8 details the main techniques for testing and validating the simulator. The results of validation tests between release 1 and release 2 are also documented in Section 4.8.

4.1.2 Motivation for the simulator

The motivation for creating this simulator was to develop a tool which would support this investigation into self-organisation in peer-to-peer systems. The investigation has one primary aim which is detailed below.

- To explore the underlying dynamics of an ant peer-to-peer system based on ant colony optimisation under equilibrium and non-equilibrium conditions.

In order to support this aim the simulation tool needed to be capable of the following functions.

- Enable control of the main operations of the ant peer-to-peer system (variation of the routing policies executed at the nodes).
- Enable control of the environment in which the ant peer-to-peer system is operating (variation in the grid size, and the density of the nodes on the grid).
- Enable the measurement of the outcome, both in terms of the topology of the network produced (path length and degree distributions) and the statistics for the ants in the system (number of ants dropped and the number of ants home).

A discrete event model was chosen when designing the simulator enabling the clock and events to be controlled precisely in the simulation. This in turn enables direct comparison of results between different starting scenarios at different points of evolution in the system (i.e. after varying numbers of simulation iterations have passed) in a way that a continuous event simulation would not. In the continuous case, it would be more difficult to compose the points to which the system has evolved when comparing simulation runs.

The rationale for the second release of the simulator is to enable finer control of the ant peer-to-peer system by separating out the demand function from the percentage of nodes that are able to send ants in the system. This is detailed further in Section 4.4.

4.1.3 Justification for building an ant peer-to-peer simulator

In conducting this research several pre-existing tools were reviewed as candidates for use in performing the experiments during the course of the work. The main candidate was AntHill, which has been reviewed in the literature survey in Section 2.6. The AntHill application follows a service provision model, where a series of interconnected nests are provided with a set of services by ants that visit the nests in the anthill. In the AntHill framework, the nests are capable of performing significant computations on the information that they receive from the ants that visit[1]. The computation capability that the nodes possess is in support of the goal of the AntHill platform to enable the development of Ant algorithms. In the work of this thesis the motive is different, as the focus is on investigating the underlying dynamics of an ant-based routing algorithm, by analysing the traffic and topological characteristics of its operation. The intention is to keep the routing calculation performed at each node as simple as possible, while measuring aggregate properties of the resultant P2P system. In support of the goal of this investigation a clear understanding of how the routing calculation was performed was needed. It was therefore decided to implement a peer-to-peer simulation with a series of nodes each with a relatively simple mechanism for routing the ants, but to focus on presenting the user with a way of varying the values of each of the quantities used in the routing calculation.

A further factor which prevented AntHill from being appropriate is the use of the tiered architecture that AntHill employs, where ants making transitions between nests do so through a series of protocol stacks as a consequence of the AntHill application architecture. Inspection showed that the results AntHill produced would be affected by the platform that the simulation engine was run on. In order to support the aim of this thesis it was decided that a simulation tool composed of a simpler, and less platform sensitive, application architecture was needed in order to assist in the comparison of results generated by multiple platforms.

4.1.4 Major releases of the ant peer-to-peer simulator

During the course of this research two releases (release 1 and 2) of the simulator occurred and were used in simulation runs to produce experimental data. The release 2 version of the simulator is a refinement of the release 1 version. The version presented in Section 4.2 is the release 1. We choose to present the release 2 version later in this chapter in Section 4.5 in the context of a comparison to the application architecture and functionality of the release 1 version.

Note that the basic calculations performed by the engines in both the releases of the simulator are the same. The enhancement between release 1 and release 2 affects the node control and overall system control parts of tool control, as explained below. However, these enhancements do not affect the way in which the ants are processed by the nodes in the ant peer-to-peer system.

[1] Examples of such computations include the routing of ants within different components of the nest in order to simulate the passing of service requests between components in different server architectures.

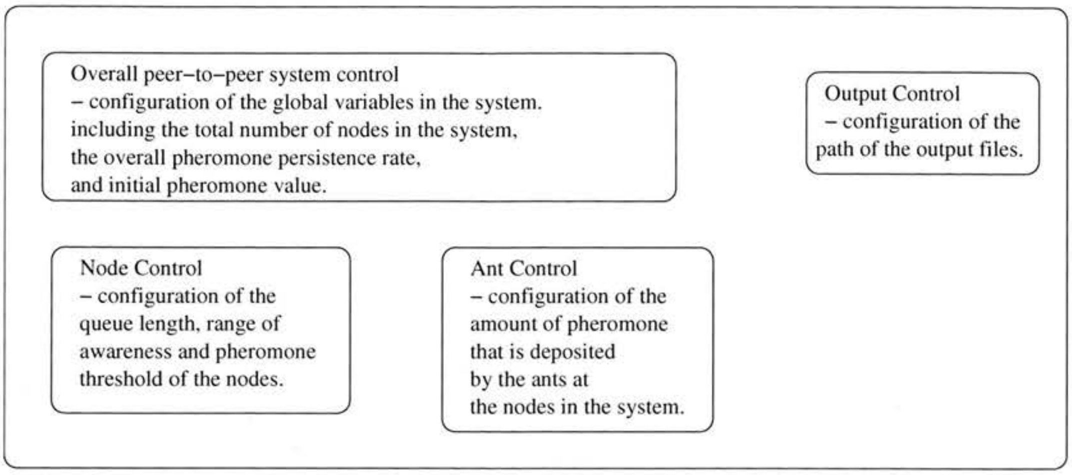


Figure 4.1: Categories of control in the ant peer-to-peer simulator

4.2 The release 1 version of the simulator

In this section we focus on the release 1 version of the simulator. In Section 4.2.1 we summarise the main functions of the simulator. Sections 4.2.2, 4.2.3, 4.2.4, 4.2.5 focus on mechanisms for controlling the overall simulation, and the nodes, and ants within the simulation respectively. The mechanism for output control is described in Section 4.2.5. The main configuration files used for initialising the simulations are described in Section 4.2.6, and the resulting output files are described in Section 4.5.4.4.

4.2.1 Main functions

The purpose of the simulator is to calculate and record the routing decisions made by a system of nodes handling traffic, which also have to cope with varying levels of congestion in the peer-to-peer system as a whole. The main functions of the simulator are as follows:

- To model the communication between the nodes in an ant-based peer-to-peer system.
- To enable snapshots to be taken of the peer-to-peer system at regular intervals during the evolution of the system in order to record the state of the system.
- To enable control of the behaviour of the ants and nodes in the peer-to-peer system.
- To enable control of the environment in which the peer-to-peer system is operating.

Figure 4.1 summarises the different categories of control within the simulator.

Each of these aspects of control of the simulator and the meaning of the associated control parameters are described in Sections 4.2.2, 4.2.3, 4.2.4 and 4.2.5 below.

4.2.2 Overall peer-to-peer system control

In this section we describe the main parameters which control the overall peer-to-peer system in which the simulation takes place. The simulation can be managed in the following way by setting a number of control parameters which determine how the monitoring of the system is performed (as it evolves). These are as follows:

- The total number of nodes in the simulation.
- The size of the spatial grid on which nodes are placed in both x and y dimensions.
- The overall pheromone persistence rate, indicates the proportion of the pheromone that remains from one iteration to the next during the simulation.
- The initial range of awareness of the individual nodes which affects the initial routing decisions that the ants make as they explore the system, and also the rate at which new possibilities are added to the system as a result of successful ant tours.
- The percentage of nodes in the system that can send ants. Note that all ants that are operational can route ants.
- The initial pheromone value placed in the routing tables at start up of the simulation in order to get the simulation started.

4.2.3 Node control

In this section we describe the main parameters for configuring the behaviour of the nodes.

- The queue length of the node indicates the maximum number of ants that a node can hold before dropping subsequent ants that arrive at the node.
- The threshold value controls how many routes in the routing table are pruned every time a snapshot is made of the system. For example, at each snapshot the average level of pheromone amongst all the routes in the routing table is calculated. If the threshold value is set to 10 percent then those routes whose value is less than 10 percent of the average are pruned.
- The range of awareness of the nodes which governs which other nodes in the system the nodes can see and route to at the start up of the simulation.

4.2.4 Ant control

The behaviour of the ants in the system can be configured using a single parameter.

- Pheromone increment rate. This governs the amount of pheromone to be deposited in the routing tables of the nodes during the course of their tours around the system.

4.2.5 Output control

The path to the output file can be configured using the following information.

- Results path, which indicates the path to the output file in which the results will be saved every time a snapshot is made of the system.

The frequency with which the information is deposited in the output files can be controlled by setting the following variables:

- The number of iterations for which the simulation engine is run in any given simulation.
- The number of iterations between snapshots.

4.2.6 Configuration files

The simulator takes input files (detailed in Section 4.2.6.1) which specify the starting configuration of the simulation and generates output files (detailed in Section 4.2.6.3) during the course of the simulation. A node information file also must be supplied indicating the positions of the nodes on the grid and their capabilities (i.e. whether a given node is capable of sending out ants or not). The node information file is detailed in Section 4.2.6.2 and Figure 4.2 indicates the simulation process as a whole.

4.2.6.1 Input files

Simulation runs are initiated in two ways. Firstly, by use of configuration files. In this section we focus on the format of the configuration files used to start the simulations. In release 1 there is a single input configuration file format. Secondly, the simulator can be configured by command line, where the user is prompted for values of the input variables. The input configuration file format is detailed in Appendix A, specifically Section A.2.1.

4.2.6.2 Node information file

In both modes of use (i.e. batch and command line interactive) the simulator needs to be supplied a node placement file which stipulates the x and y coordinates of the nodes in the peer-to-peer system on the grid. This is important when trying to assess the effect of varying the values of some of the key parameters of the system (such as pheromone persistence rate) on the resultant system topology. By being able to ensure that the nodes are placed at the same points on the grid at start up time, and also that the respective nodes in the system have the

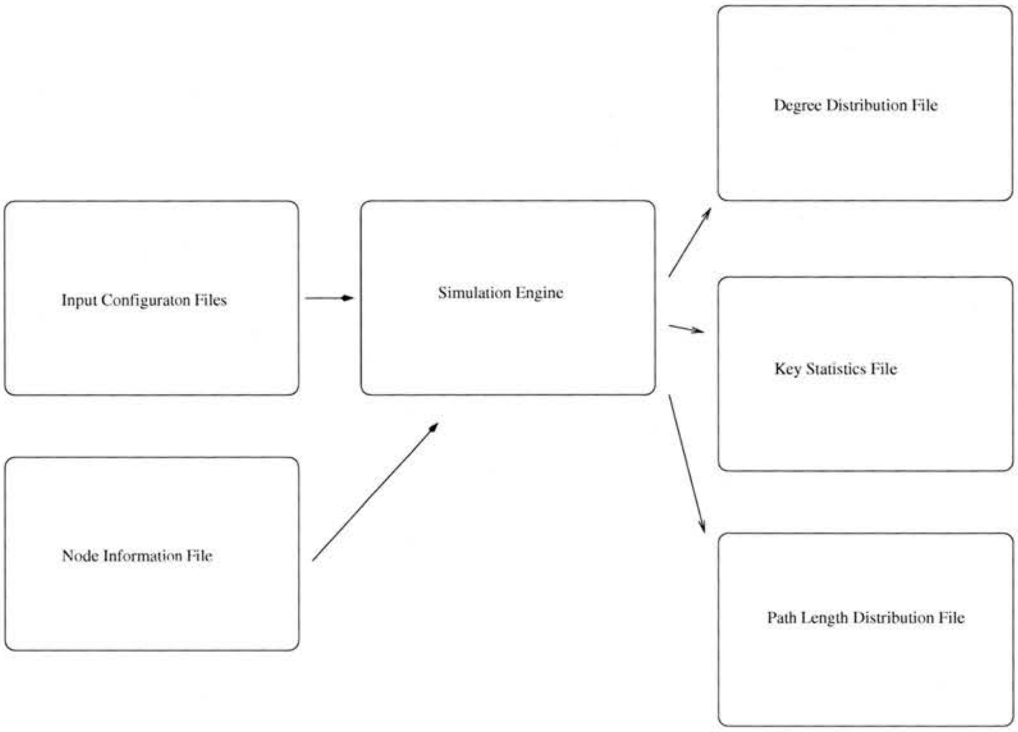


Figure 4.2: The simulation process

same capabilities (i.e. to either be able to initiate ants and route them, or simply to be able to route them) the effects of varying some of the other parameters can be isolated. The format of the node information file for release 1 is detailed in table A.2, in Appendix Section A.2.2.

4.2.6.3 Output files

The simulator generates three output files as depicted in Figure 4.2. These output files are the degree distribution file, the path length distribution file, and the key stats file. Each of these files is generated every time a snapshot is made of the simulation, at a frequency defined in the initial configuration file for the simulation. The meanings of the files, and their formats are described respectively below in this section.

Degree distribution file This file contains the degree distribution information[2] of the nodes in the simulation. We define the degree of a node as the number of direct connections to other nodes the given node has. The degree distribution is a plot of a given degree against the proportion of the total nodes in the system that have that given degree. Different types of topology have different degree distribution functional forms. The fields in this output file are described in Table A.3 in Section A.3.1.

[2] The values measured are the out-degrees of the nodes in the simulator.

Path length distribution file The path length is the number of hops a given ant takes between source and destination nodes. The values for the path lengths in this output file are generated from tours of ants that make it successfully back to the source node. The format of the file is detailed in table A.4 in Section A.3.2.

Key statistics file In this section we detail the format of the key stats file, which holds many of the aggregate statistics generated during the course of the simulation runs. The main aggregate information held here is the average path length, average degree, and the total number of ants dropped during the course of the simulation to the measurement point. The format of the file is detailed in Table A.5 in Section A.3.3.

4.3 Simulator design overview for release 1

In this section we indicate some of the detail of the workings of the simulator. Specifically we focus on the application architecture in Section 4.3.1, the main control loop for the simulator in Section 4.3.2, and finally the main ant processing routine in Section 4.3.3.

4.3.1 Application architecture

In this section we detail the simulator architecture, by indicating what the main components of the simulator are, what their main responsibilities are, and how the components interact. Figure 4.3 indicates the static design of the application, including the multiplicity relationships between the classes.

We now go through each of the objects in the class diagram and indicate the main functions of the objects.

SimulationObject — this object is responsible for getting hold of the parameters for the simulation, as defined in the configuration files (Section 4.2.6). This object also manages the simulation process as a whole by keeping track of the simulation iterations and determining when snapshots are taken of the current state of the system.

Network — this object contains the main simulation engine of the simulator, and is responsible for the generation of the forward ants.

Ant — this object holds all the information regarding the ants in the system including their current location, the amount of pheromone they lay down in the routing table entries, whether the ants are travelling forwards (i.e. towards their destination node) or backwards (towards their source node).

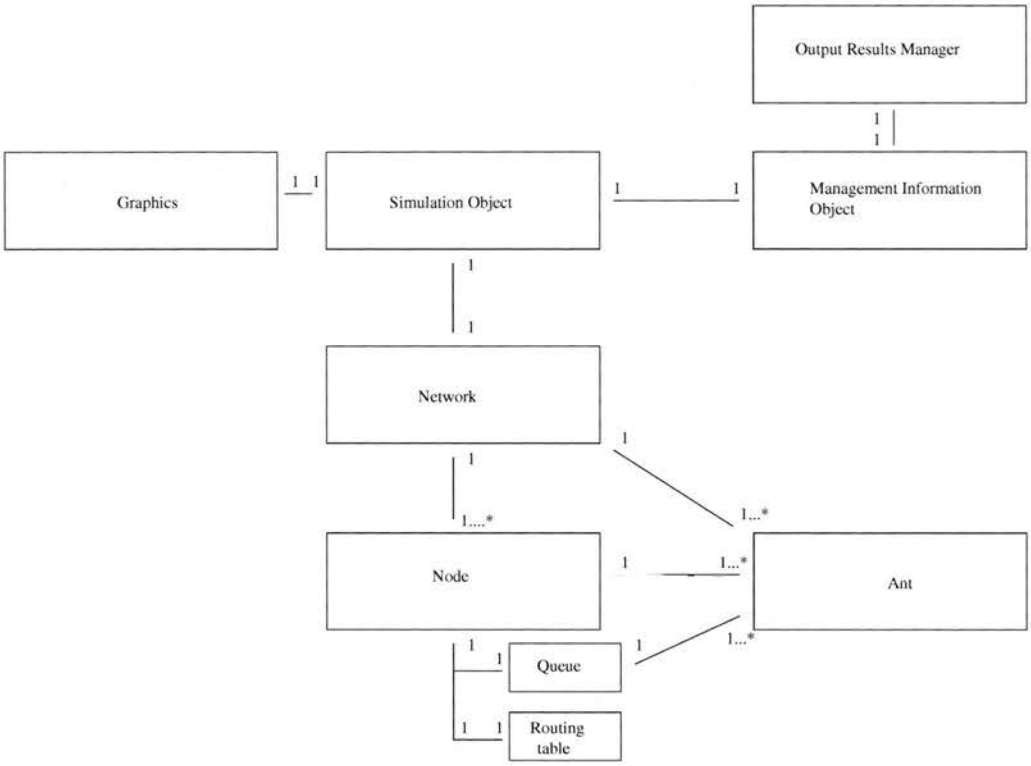


Figure 4.3: The class diagram for the ant peer-to-peer simulator

Node — this object holds the information regarding the nodes in the system including the position of the node on the grid, whether the node can send ants or not, and the pheromone persistence rate that is to be used in the routing table update process.

Queue — this object is held within the Node object, and is used as a first-in-first-out queue for the processing of ants. This object also manages the batch sizes for the processing of ants in the system.

Routing Table — this object is responsible for the containment of the routing information within the node object by holding pheromone levels. This is also where the ants that travel through the system update the pheromone levels.

Management Information Object — this object is used in the collection of management information from the system, and its routines are invoked every time a snapshot is taken of the simulation.

Output Results Manager — this object is responsible for the production of the output files as described in Section 4.5.4.4 and is invoked every time a snapshot is made of the state of the system.

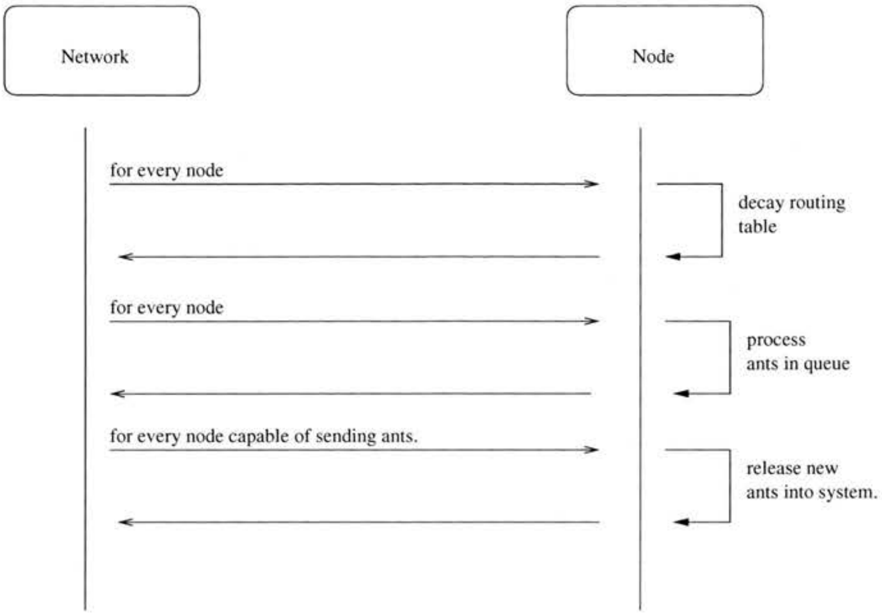


Figure 4.4: The main ant processing routine for the nodes in our ant peer-to-peer system

4.3.2 Main control loop

Having described the static architecture of the simulator, we now focus on the main control loop of the simulator. The main control loop is executed every iteration during the simulation, and is administered by the Network object, and consists of three basic steps described graphically in Figure 4.4 and listed below.

1. Decay routing tables — here we cycle through all of the nodes decaying the pheromone values in the routing tables of the nodes according to the pheromone persistence rate that has been supplied either in the configuration file or by the user.

2. Process ants in queue — for every node we process the ants in the queue of the node. The order in which the nodes are processed is random every iteration. The way that the process ants in queue routine functions is described in detail in Section 4.3.3.

3. Release a new ant from each node which is able to send ants into the system.

The reason for including Figure 4.4 is to enable easy comparison with the control loop functionality in release 2 of the simulator listed later in this chapter, in Section 4.6.2.

4.3.3 Main ant processing routine

In this section we describe the way in which the ants are processed in the queues of the nodes. Decisions about how the ants are processed in the system are made according to two criteria.

Firstly, whether the ant has completed the leg of its trip (i.e. forward or backward). Secondly, the direction in which the ant is travelling namely, forward (from its source towards its destination) or backward (from its destination towards its source). Finally, for nodes which are not at their destination a check to see whether the path is valid is made before onward routing. The criterion applied to assess whether the path is valid is simply to assess whether the node the ant is being routed to is accepting further ants or not in the present iteration of the simulation. In release 1 of the simulator the criterion for a node not accepting ants is only that the node is congested (i.e. the queue of the node is full). Under these conditions ants that arrive at the node are dropped from the overall peer-to-peer system. At all other times the node is able to accept and process ants. A flow chart indicating the flow of control in the ant processing routine is detailed in Figure 4.5.

Table 4.1 summarises the activities of the ants traversing the different decision paths.

Decision Path	Condition under which ant follows decision path
A	Ants in transit moving back towards its source
B	Ants in transit moving towards destination
C	Ant has successfully returned to its source having visited its destination
D	Ant has successfully reached its destination now needs to be turned round back to its source

Table 4.1: Decision path conditions for ant processing in simulator

In Figure 4.5 we see that the flow chart has been divided into four decision paths (indicated by vertical columns) which define the way in which different categories of ants are processed in the system. The criteria which define each of the decision paths are summarised in Table 4.1. These criteria are formed by enumerating the combinations of whether the ant is a backward or forward ant and also whether the ant has completed the leg of its tour either to its destination (in the case of forward ants), or to its source (in the case of backward ants).

4.3.4 Time management and synchronisation

The iterations of the main ant processing routine are driven from a central loop which iterates until the total number of simulation iterations for this simulation has been reached. It is this control implementation which makes this simulator a discrete event simulator. Furthermore, the simulator is a synchronous simulator. Within each timestep the nodes are selected in an arbitrary order and their queues processed in turn. When a node's queue is processed all ants are processed in the order they appear in the queue and all other nodes remain unchanged.

We now consider the implications of the synchronous operations of the simulation engine

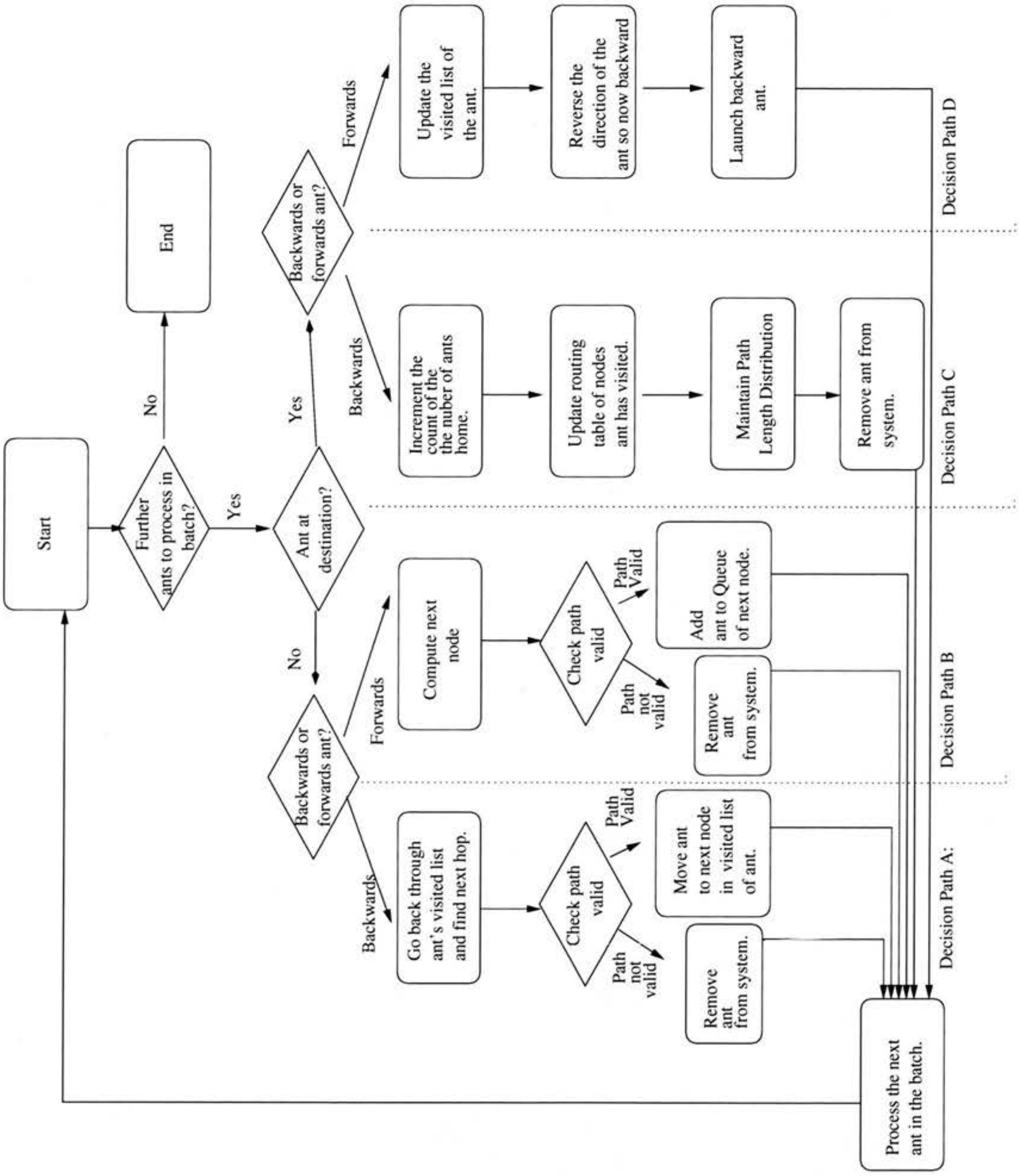

Start
Further ants to process in batch?
No
End
Yes
Ant at destination?
Yes
No
Backwards or forwards ant?
Forwards
Backwards
Update the visited list of the ant.
Reverse the direction of the ant so now backward
Launch backward ant.
Increment the count of the the nuber of ants home.
Update routing table of nodes ant has visited.
Maintain Path Length Distribution
Remove ant from system.
Backwards or forwards ant?
Forwards
Backwards
Compute next node
Check path valid
Path Valid
Path not valid
Add ant to Queue of next node.
Remove ant from system.
Go back through ant's visited list and find next hop.
Check path valid
Path Valid
Path not valid
Move ant to next node in visited list of ant.
Remove ant from system.
Process the next ant in the batch.
Decision Path A:
Decision Path B
Decision Path C
Decision Path D

Figure 4.5: Flow chart indicating the way in which the ants in our peer-to-peer system are processed in the queues of our nodes, partitioned into four major decision paths

in contrast to a corresponding asynchronous simulation engine. In the synchronous case, even though in each iteration the ants resident in the queues of each node in the system are processed, only one node queue's ants are processed at a time. This means that the congestion that the ants experience is small as the only congestion caused is by ants from the same source node within the simulation iteration in question. In the corresponding asynchronous case, within an equivalent time interval[3]all the ants in the queues of all the source nodes are processed at once so there is likely to be far more congestion in the system. This means that in the asynchronous case the proportion of ants reaching their destination safely is likely to be smaller than in the synchronous case. In summary the synchronous case forms an upper bound in terms of traffic statistics, and therefore also emergent topology.

4.4 Enhancement of the simulator

As documented later in this thesis, in the chapter on equilibrium behaviours, the results generated by release 1 of the simulator are complex. Release 1 had its limitations but was able to generate insight into the dynamics of the system. This observation of complexity led to the need for finer control of the system, especially in non-equilibrium situations. It is this need that drove the development of release 2 of the simulator. We motivate the development of release 2 by indicating the limitations of release 1 below.

We have seen in the sections above that release 1 of the simulator injects ants into the system every simulation iteration from every node which is able to send ants. This functionality couples the percentage of nodes sending ants to the demand function [4] that the system is responding to. In release 1 it was not possible to independently vary both the demand function and the percentage of nodes sending out ants in the peer-to-peer system. The motivation behind the enhancement for the simulator, (and therefore the creation of release 2), is to enable finer control of failure behaviour and demand functionality especially in non-equilibrium situations during the simulation runs. The associated configuration files, application architecture, and functionality of the release 2 version of the simulator are described in Section 4.5. As mentioned above, the way in which the ants are processed by the nodes is the same in releases 1 and 2 of the simulator. The extensions to include the failure and demand variation functionality are small extensions to the main operation of the simulator. Furthermore, as stated above the enhancements in release 2 do not fundamentally alter the way in which the simulation engine works.

[3] In the asynchronous case there are no simulation iterations.

[4] By the demand function we mean the rate at which ants are introduced into the ant peer-to-peer system. So many per simulation iteration.

4.5 The release 2 version of the simulator

4.5.1 Main functions

The main functions of release 2 of the peer-to-peer ant simulator are the same as the release 1 version of the peer-to-peer simulator detailed above. The control model has changed slightly to accommodate the need for independent control of the percentage of nodes sending out ants, the failure behaviour of the nodes in the peer-to-peer system and also the demand function placed on the peer-to-peer system.

The purpose of the release 2 simulator is still to calculate the routing decisions made by a system of nodes handling traffic, which also have to cope with varying levels of congestion and changes in the environment (such as removal of nodes, or the saturation of their capacity). The main functions of the simulator remain as stated in Section 4.2.1.

However, in addition there are also functions associated with node failure and demand placed on the system.

- To control the probabilistic failure of a series of nodes in the peer-to-peer system.
- To enable demand placed on the system to vary, both, dynamically during the course of the simulation runs, and independently of the percentage of total nodes sending out ants.

The control scheme can still be summarised by the diagram in Figure 4.1.

We now go on to describe how each aspect of simulator control was enhanced to accommodate the new functionality in the release 2 version of the simulator.

4.5.2 Overall peer-to-peer system control in release 2

In this section we describe how the main parameters which control the environment in which the simulation takes place have been enhanced. The following pieces of functionality have been added.

- The Demand Multiplier, which determines how many ants are released into the system each iteration. The meanings of the demand multiplier and the way in which burstiness of traffic within the system has been formulated is described in the Chapter 6.
- The Demand Mode indicating whether the demand function is constant demand or whether there are variations in demand during the course of the simulation.

Furthermore, release 2 supports two modes of operation. Firstly, *constant demand* where the demand on the peer-to-peer system is constant throughout the simulation run. Secondly, *bursty demand*, where the demand changes throughout the simulation run. These two modes are supported by the following two inputs:

- If the simulation is to be conducted with constant demand then a constant demand multiplier is specified.
- If the simulation is to be conducted using bursty demand then a path to the file containing the succession of demand multiplier values is to be specified.

4.5.3 Node control

In this section we describe how the node control behaviour has been enhanced to incorporate the failure behaviour of the system.

- The probability that the node will make a transition from the operational state to the non-operational state.
- The probability that the node will make a transition from the non-operational state to the operational state.

The meanings of both the operational and non-operational states are described in Chapter 7.

4.5.4 Configuration files

As for release 1, the release 2 version of the simulator takes input files (detailed in Section 4.5.4.1 and defined in Appendix A Sections A.4.1 and A.4.2) which specify the starting configuration of the simulation and generates output files (detailed in Section 4.5.4.4) during the course of the simulation. Figure 4.2 indicates the process.

4.5.4.1 Input files

Simulation runs are initiated in two ways. Firstly, by configuration file where the input values are fed to the simulator using a configuration file. In this section we detail the format of the configuration files used to start the simulations. There are two types of configuration files depending on whether the simulator is simulating constant demand on the system or varying demand on the system. These configuration files are detailed in the Sections 4.5.4.2 and 4.5.4.3 below, where it can be seen that they are subtly different in format. Secondly, the simulator can be configured by command line, where the user is prompted for values of the variables detailed below in the sections on configuration files in Sections 4.5.4.2 and 4.5.4.3.

4.5.4.2 Definition of configuration file for constant demand simulations

The format of the configuration file for constant demand is shown in Table A.6, in Appendix Section A.4.1. There are two main differences between this configuration file and the configuration file for release 1. Firstly, the inclusion of the demand mode flag, which indicates whether the simulation is to be run with constant demand or variable demand (a demand mode value of -1 indicates constant demand). Secondly, the inclusion of the demand multiplier variable, which indicates how many ants are to be released into the system every iteration. The exact definition of the demand multiplier is explained in Chapter 6.

4.5.4.3 Definition of configuration file for bursty demand simulations

The configuration file for the bursty demand scenario is detailed in Table A.7 and is very similar to the table for the constant demand scenario. The main difference is the value for the demand mode changes to 1, and therefore two further fields are required: the path to the file containing the demand information, and the number of demand measurements. The demand information is the sequence of values of the demand multiplier used in each subsequent iteration. The number of demand measurements is the number of demand multiplier values contained in the demand file, which must be sufficient to run the simulations. Finally, we no longer need a value for the demand multiplier in this scenario as those values will be read from the file containing the demand information. The fields, formats, and meanings of the configuration file for bursty demand are described in Table A.7 in Section A.4.2.

4.5.4.4 Output files

The release 2 simulator generates three output files as described in the Figure 4.2. These output files are the degree distribution file, the path length distribution file, and the key stats file. Each of these files is generated every time a snapshot is made of the simulation. The frequency of these snapshots is defined in the initial configuration file for the simulation. The way in which the output files differ from those of release 1 is indicated in this section below.

Degree distribution file This file contains the degree distribution information of the nodes in the simulation. The fields in this output file are described in Table A.8 in Appendix Section A.5.1. The main change is the inclusion of the value for the number of operational nodes, as this can change from iteration to iteration during the simulations, and needs to be recorded each time a snapshot is taken.

Path length distribution file In this section we detail the format of the path length distribution file. The values for the path lengths in this output file are generated from tours of ants that make it successfully back to the source node. The format of the file is detailed in Section A.5.2 of the Appendix A.

Key statistics file This file holds many of the aggregate statistics generated during the course of the simulation run. The main aggregate information held here is the average path length, average degree, and the total number of ants dropped during the course of the

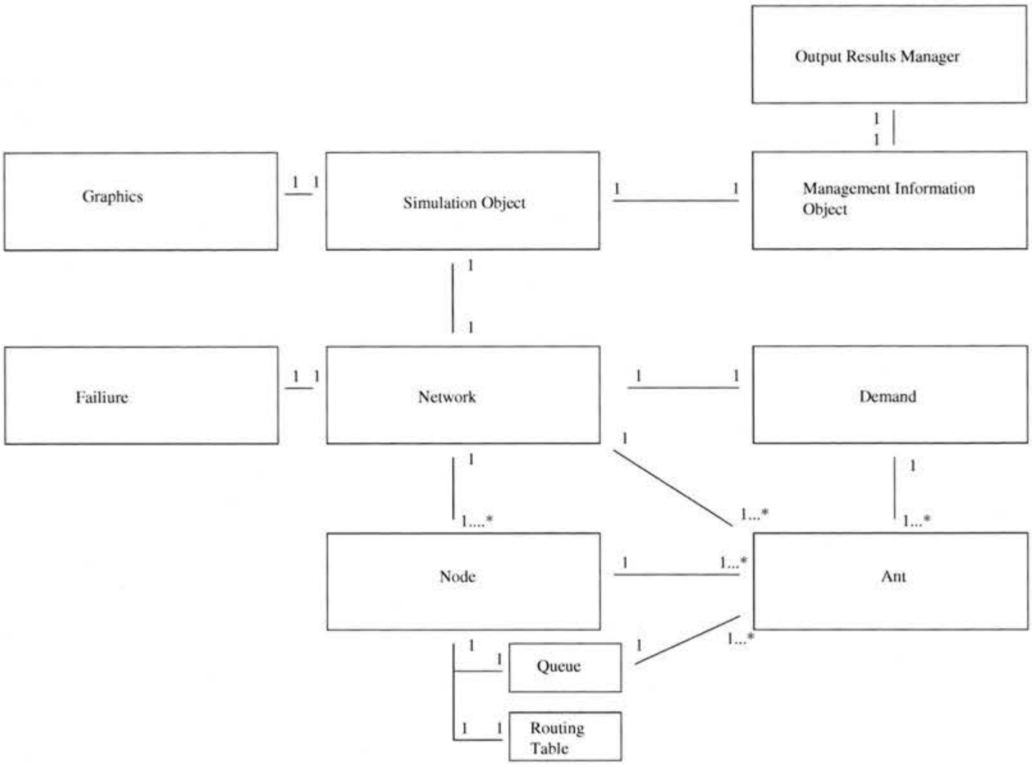


Figure 4.6: The class diagram for the ant peer-to-peer simulator

simulation to the measurement point. The format of the file is detailed in Table A.10 in Section A.5.3 of Appendix A.

4.6 Simulator design overview for release 2

In this section we explore how the main mechanisms of the simulator have had to change in order to accommodate the new functionality included in release 2 for demand and failure. Specifically we focus on the application architecture in Section 4.6.1, the main control loop for the simulator in Section 4.3.2, and finally the main ant processing routine in Section 4.6.3.

4.6.1 Application architecture

In this section we detail the release 2 simulator architecture, by indicating what the main components of the simulator are, and how this differs from the release 1 version. Figure 4.6 indicates the static design of the application, including the multiplicity relationships between the classes.

In the case of the release 2 version of the simulator the new objects are as follows.

Demand Object — this object is responsible for the regulation of the number of ants injected into the system at every iteration. It uses either a constant demand model or a bursty demand model depending on the nature of the configuration file.

Failure Object — this object is responsible for the computation of which nodes are in the operational and non-operational states at every iteration. It also performs the computations necessary to determine which nodes make transitions between these two states every iteration of the simulation.

There were also enhancements to the network object which now controls the failure of the nodes in the system and also the variations in demand in the system during the progress of the simulations. This is described in the main control loop section below.

4.6.2 Main control loop

The main control loop of release 2 is executed every iteration during the simulation, and is administered by the Network object, and consists of five basic steps described in Figure 4.7 and listed below.

1. Compute failures and resurrections — here we simply compute the operational status of the nodes in the ant peer-to-peer system.

2. Compute demand on the system — here we compute the number of ants to be released into the system using either the constant or bursty demand model depending on the configuration file that has been supplied or the values of the parameters supplied by the user at run-time.

3. Decay routing tables — here we cycle through all of the nodes decaying the pheromone values in the routing tables of the nodes according to the pheromone persistence rate that has been supplied either in the configuration file or by the user.

4. Process ants in queue — for every node we process the ants in the queue of the node. The order in which the nodes are processed is random every iteration. The only stipulation is that each node which is in an operational state within the ant peer-to-peer system must have the ants in its queue processed each iteration during the simulation. Nodes that are in non-operational states in a given iteration do not have the ants in their queues processed. The way that the process ants in queue routine functions is described in detail in Section 4.6.3.

5. Release a new ant from each node capable of sending ants until the total new ant count equals the value calculated in the demand step.

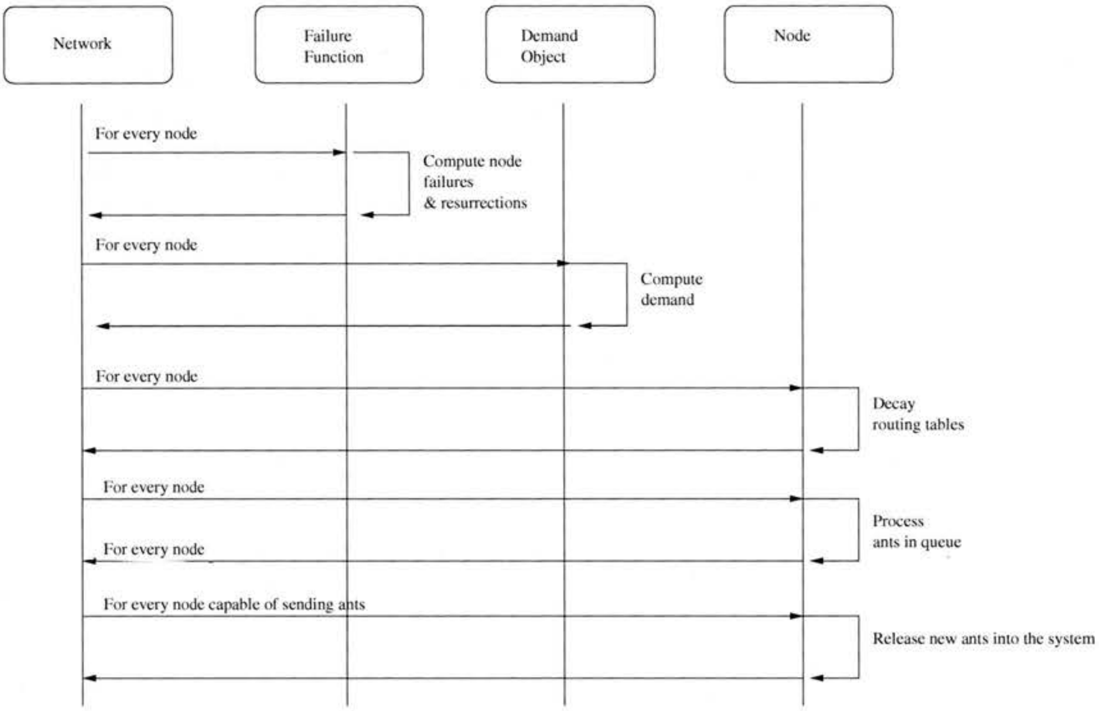


Figure 4.7: The main ant processing routine for the nodes in our ant peer-to-peer system

4.6.3 Main ant processing routine

The main ant processing routine for release 2 of the simulator is the same as for release 1, namely that shown in Figure 4.5. The only change is the criterion applied to assess whether the path is valid or not when assessing how to route ants onward. We must now consider a path to be invalid if the node to which the ant is being routed is in a non-operational state. Ants that want to take invalid paths are dropped from the system.

4.6.4 Time management and synchronisation in release 2

The iterations of the main ant processing routine are driven from a central loop, as for release 1, which iterates until the total number of simulation iterations for this simulation has been reached. We include the computation of failures and demands in the system.

4.7 Implementation issues

The comments stated in this section on implementation issues apply equally to both release 1 and release 2 of the simulator.

4.7.1 Design methodology

The development of a complex application such as the ant peer-to-peer simulator necessitated the use of modelling tools to assist in the tracking of the functionality, and message passing in the main code segments, and also the definitions of the main data structures. UML diagrams such as the sequence diagram in Figure 4.7 and the class diagrams in Figure 4.3 were used for this purpose.

4.7.2 Usefulness of UML

The use of UML to assist in the documentation of the classes in the application and the message passing between the classes during the operation of the simulator enabled classes to be clearly defined, with sets of method functions within those classes which minimised the number of ways in which the key variables in the system could be changed. The UML documentation also enabled the re-use of the appropriate method functions during development and therefore, decreased the amount of coding that was needed. This enabled a modular design which reduced the amount of testing that needed to be done.

4.7.3 Implementation in Java

Both the release 1 and release 2 versions of the ant peer-to-peer simulator were implemented using Java which ensured the tool's portability. Eclipse 3.0 was used as the main compiler to assist in the development of the code. All of the Java code in this simulator uses Java SDK Version 1.4.2.

4.8 Testing and validation

A number of techniques were used to test and validate the simulator during its development. The interested reader is directed to [35] for additional background on these techniques which are listed below.

Unit and Integration Testing The modular design (reflected in the application architecture (section 4.3.1) and release 2 (section 4.6.1)) enabled the testing of each of the major objects before integrating them into the application in its entirety. Integration testing was then used to ensure that the objects worked together within the application, and that information was passed effectively and accurately during the course of the calculation.

Structured Walk-Through For the major subroutines of the simulator a structured walk-through of the code was performed to enable independent scrutiny of the functionality. The walk-through was performed by a number of fellow students all of whom had knowledge of the application, and a good understanding of the Java programming language.

This method was of most value when crafting and testing the main ant processing routing for both releases (detailed earlier in Sections 4.3.3 and 4.6.3).

Trace Outputs Test runs were performed and the values of the major variables were tracked during the simulation. This method was most applicable when unit testing the objects stated in the sections on application architecture. The most demanding objects to get right were the Routing Table object which holds routing table information and Management Information object which holds information about the aggregate statistics of the peer-to-peer system during the course of the simulations.

Consistency Checks The values of the main measurements taken during the simulation runs were calculated in more than one way in the simulation engine, and the values compared in order to expose bugs in the application. Examples of such variables are the degree distribution and also the path length distribution of the system. These were most useful when checking the calculations needed in the Management Information object, and also the correct operation of the ant processing routine detailed in Sections 4.3.3 and 4.6.3.

Control Experiments A further method of validation was the use of control experiments. Here scenarios where the behaviour of the system was predictable were encoded using configuration files, and the simulations run and simulation results examined. An example of a control experiment scenario is where the nodes in the simulation have a very large initial radius of awareness, so the initial substrate graph in the peer-to-peer system is a perfect graph, where all the nodes are directly connected to one-another. Under these circumstances we would expect a large proportion of ants to successfully reach their destination, although some may fail due to congestion at the destination nodes filling the queues. The results of the simulations indicated that a high proportion (in the order of seventy percent) of ants did successfully get through to the destination nodes.

4.8.1 Validation of simulation literature

Inaccurate simulation models lead to false conclusions. The literature presents two main methods of validating simulation models in peer-to-peer systems. Firstly, validation of a simulation of a distributed routing algorithm is conducted by direct comparison to experimental data [41]. Secondly, consistency of results across repeated experiments is used as the method of validation [50].

4.8.2 Relevance of simulation validation literature

Clearly, the method of direct comparison with experimental data is only appropriate where experimental data is available. This is not the case in our situation, so we chose to use repeatability of experimental results as the main method of validation of our simulator.

4.8.3 Validation between release 1 and release 2

In order to validate between the two releases of the simulator, a series of checks for consistency of the results produced by both releases of the simulator was performed. However, it was only possible to perform these consistency checks in the limiting case where all nodes in the peer-to-peer system were operational. This is because, as indicated above, the failing of nodes is not supported in release 1. However, the calculations in the simulation engines of both releases remain the same, and the effectiveness of this is what is being checked for. The output variables which were used to perform the consistency checks are the average degree distribution, and the average path length. Both of these values were measured at five disparate points in the configuration space in the system (as defined by a series of values for range of awareness, pheromone persistence rate, and percentage of nodes in the system able to send out ants). The results of these consistency checks are documented in Appendix B. We note that the expectation is not to produce an exact match in the values of the metrics between releases, but observe that the results are similar in the different parts of the configuration space. This is because the ant colony routing algorithm is stochastic in nature and, as such the routing decisions taken at each of the peers when processing ants will not necessarily be the same between different simulation runs of the system.

Chapter 5

Equilibrium Behaviours

5.1 Introduction

This chapter documents the behaviour of an ant-based peer-to-peer system, under conditions of a constant demand placed on the system. We examine the different topologies produced by a number of runs of the ACO algorithm with different seeding parameters. The seeding parameters are the *pheromone persistence rate* applied to the routing data at each node, the *initial range of awareness* of the nodes in the system, and the *percentage of nodes in the system* that can create and send out ants in the first place which are then routed on an ongoing basis within the system. The seeding parameters define the configuration space of the ACO system. The primary method of measuring the topology of the system is to look at aspects of the degree distribution, and path length distribution of the system once the system has evolved for 3000 iterations. We assume that 3000 simulation iterations of each configuration is sufficient to bring the system to equilibrium. This assumption is justified in Section 5.1.1.

5.1.1 A note on the warm up and initialisation period for the ant peer-to-peer system

During the operation of the ant peer-to-peer system information is exchanged between the peers in the system. We must distinguish between the information exchanged during the course of the initialisation of the system (where during warm-up primarily the routing tables are populated) and the information exchanged between the peers during the operation of the system (where during operation the system is coping with the traffic demand placed on it, and the changes in the entries in the routing tables are small in comparison to during the warm-up phase). The need for this distinction is that during warm-up the routes of the system have not formed, and also what routes there are in the routing tables of the system may not be useful, as the reinforcement mechanisms of ACO will not have been given a chance to filter out useful from useless routes in the system. So in the warm up phase, the operation of the ant peer-to-peer system is more vulnerable to disruption due to either congestion or failure, than in the operational phase of the system where the information in the routing tables in the nodes are more stable and the routes between the peers in the peer-to-peer system are more likely to be complete. In the equilibrium experiments detailed in this chapter we justify the choice of 3000 simulation iterations as being the length of run for the simulation by noting that after 3000 iterations the values of the parameters measured (detailed in Section 5.2) vary only by between 5 and 7 percent between different simulation runs starting at the same point in the configuration space, meaning that the values are within a 95 percent confidence limit. This confidence limit criterion was used during these experiments to determine when the system has reached its asymptotic limit (beyond which allowing the system to evolve will cause no significant change to the system).

5.1.2 Chapter structure

The remainder of the chapter is organised as follows. Section 5.2 describes the meanings of the main parameters that drive the system. Section 5.3 then discusses the main experimental observations made under equilibrium conditions. Finally, Section 5.4 draws all the equilibrium experimental data together to create a summary.

5.2 Experimental setup and main parameters

The experiments focused on the behaviour of the ACO peer-to-peer algorithm, at convergence, having been seeded with varying initial conditions. The meaning of the main parameters seeding each of the simulation runs are detailed in the sections below. Simulations are initiated using a range of values of seeding parameters. Each simulation is run for 3000 iterations. In the equilibrium experiments presented here the number of ants released into the system every iteration remains constant once defined by the percentage of nodes sending out ants. Also all nodes remain in an operational state (i.e. are capable of routing traffic) throughout all the simulation runs. The effect of extreme values in each of the variables, and the constraint on the system that they represent is summarised here.

Pheromone persistence rate The constraint that this variable represents is information persistence. If the pheromone persistence rate is too stringent (i.e. at a minimum) then all the information in the routing tables is destroyed, along with any structure in the resulting network. If there is no pheromone decay (i.e. at a maximum) then entries in the routing tables are not removed and the nodes have no way of knowing the best way of routing packets.

Range of awareness The constraint that this variable represents is one of adequate connection between the nodes. If the range of awareness is too great (i.e. at a maximum) then the routing tables become populated early on in the simulation, and these routing entries reflect environmental properties (such as initial node placement [1] and relative distance on the grid), but do not reflect learned behaviour such as the establishment of good routing solutions to problems of congestion. If the range of awareness is at a minimum there is simply inadequate connection between the nodes and no network can be established.

Percentage of nodes sending out ants in the system Given that in the ACO mechanism information is transferred by the laying down of pheromone, a minimum number of ants must circulate in the system in order for a minimum pheromone level to be laid down in the network. The constraint that the percentage of nodes sending out ants represents in this system is one of information currency. If there is an insufficient number of ants circulating in the system (i.e. the percentage of nodes sending out ants is too low) then there is insufficient information exchange between the nodes. If there are too many nodes sending out information then the system becomes congested (i.e. the percentage of nodes sending out ants is too high) and still not enough information gets through. Under these conditions the queues in the nodes will become full with ants and further ants added to the system will be dropped as a result.

We note that the overall stipulation is that conditions must enable sufficient information exchange between the nodes in the peer-to-peer system. We now explain the configuration space for the experiments.

In combination the range of values of each of the seeding parameters form a configuration space. We can visualise this as a three dimensional grid as indicated in Figure 5.1.

5.2.1 Pheromone persistence rate

The pheromone persistence rate governs how quickly the information of a new route decays in the system, and can be related to the number of iterations it takes for this new routing information to vanish completely from the routing table of a node in the system. Mathematically we can express this using the binomial theorem. If r is the percentage of pheromone that decays each iteration then we can express the fraction of original pheromone remaining after n iterations as:

$$(1-r)^n = \sum_{i=1}^{n}(-1)^i \frac{n!r^n}{(n-i)!(i)!}$$

The interested reader is referred to chapter 6 of [7] for a detailed explanation.

For the purposes of calibration we shall consider the quantity $(1-r)$ the *pheromone persistence rate*. In our experiments this quantity varies between 1, i.e. no pheromone decay, and 0.9

[1] Note the effect of initial node placement, i.e. patterns of nodes on the grid, was not investigated in these experiments

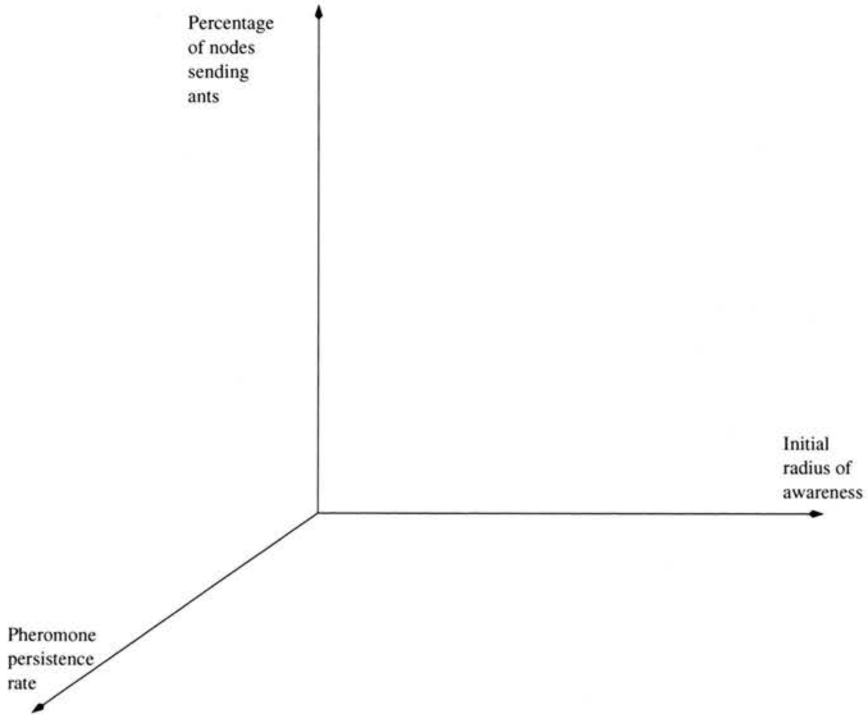

Percentage of nodes sending ants
Initial radius of awareness
Pheromone persistence rate

Figure 5.1: Configuration space of equilibrium experiments

i.e. 10 percent of pheromone decaying every iteration. The justification for this range of values is that we wish to tune the memory of the system to the number of simulation iterations that it takes for ants to return back to the source node. In practise a round trip of 10 hops (equivalent to a pheromone persistence rate of 0.9) was adequate for ants to get from the source to the destination node in most cases.

From the binomial theorem we know that this collapses to

$$(1-r)^n \simeq 1-nr$$

for very small values of r approaching zero, as under these conditions the higher terms in the series can be neglected.

If we now consider the conditions under which pheromone deposited vanishes to zero then we can set the right hand side to be equal to zero.

$$1-nr=0$$

which re-arranges to [2]

$$n=\frac{1}{r}$$

[2] This analysis is dependent on the value of r being very small.

We can read this as the number of iterations it takes for the pheromone to vanish is equal to the inverse of the percentage of pheromone that decays at each iteration. So, for example if five percent of the pheromone decays each iteration then it will take twenty iterations for the total pheromone in the system to decay.

We can relate this to the number of iterations between snapshots when measuring change in the system, which needs to be at most half the duration of the vanishing time, in order for the system to be sampled at a sufficient rate in comparison to the rate of change of the system for the change to be visible in the time dynamics of the data collected [3].

5.2.2 Initial range of awareness

The initial range of awareness of the nodes is the initial radius of awareness within which the nodes are aware of one another. Associated with the range of awareness is the notion of a *connection threshold.* The connection threshold is the minimum range at which connections can be formed between nodes in the system. This enables structures to evolve. At ranges of awareness below the connection threshold, in general, nodes cannot see one another when the simulation is initialised and because of this there can be no substantial information exchange. We can formulate this mathematically by considering the following inequality:

$$(\pi) * (d^2) * \rho \geq 2$$

where π is the constant 3.14, ρ is the density of the nodes on the grid, and d is the distance between neighbouring nodes on the grid.

The reason for the greater than two stipulation is that in order for a connection to be made two or more nodes need to be visible to one another. We can define the connection threshold, $d_{connection}$, by re-arranging the equation above as follows

$$d_{connection} = \sqrt{\frac{2}{\pi\rho}}$$

We see that the connection threshold is inversely proportional to the square root of the node density ρ.

[3] Note that when collecting aggregated statistics about the overall state of the system over a long simulation run, it is not necessary to sample the system at twice the rate of the vanishing time. The reason for this is that sampling is only necessary when we a re concerned with the time dynamics of the system. Aggregated statistics are acrued during the entire length of the simulation run, and need only be collected at the end of the simulation run. The only exception to this is in determining that the system has converged. In order to do this all we need do is compare the similarity of the results between subsequent snapshots, and as long as the frequency of the snapshots are at most half the vanishing time this enables convergence to be determined.

For the conditions of our experiment the value of $d_{connection}$ is roughly 60 units [4].

In the experiments the initial range of awareness used to seed the simulation runs ranged from 40 units to 180 units, enabling us to examine the effect of being either above or below the connection threshold.

5.2.3 Percentage of nodes that can send out ants

The percentage of the total number of nodes that can initiate and then send out ants is just a fraction between zero and one, and is an indication of the level of traffic circulating in the system. In general, we note that unless there is a minimum level of traffic circulating in the system, stimulated by a minimum level of demand placed on the system, the control processes which prune bad routes and promote good routes cannot function. Further, under these conditions the structures created in the network will remain random, as no self-organisation will be possible.

In our equilibrium experiments the percentage of nodes that can send out ants is varied from 20% to 100% in steps of 20%, giving five distinct values.

5.2.4 Density of the nodes on the grid

In this section we consider a further parameter of the system, the density of the nodes on the grid. In all the experiments this is kept constant. The nodes in the experiment were all placed on a two dimensional grid at random locations. The node density is simply the number of nodes in the gird divided by the total area of the grid that they are placed on. Nodes are placed on the grid according to position information fed into the simulator in the node information file, as discussed in Chapter 3.

$$\rho = \frac{m}{A}$$

where m is the total number of nodes in the system and A is the total grid area.

In the equilibrium experiments, the area of the grid remained constant at 1000×1000 units, and the number of nodes in the system remained constant at 200. This means that the node density remained constant at $\frac{1}{5000}$ nodes per square unit. As stated above the connection threshold, $d_{connection}$ is roughly 60 units.

5.2.5 Summary of experimental parameter values

We sumarise the experimental parameter values here.

[4] In this case a unit is simply the length distance used by the graphics program in order to position points in a 2 dimensional coordinate system.

Field name	Range of Parameter value
Percentage of nodes sending out ants	Ranges from 100 percent to 20 percent in steps of 20 percent
Pheromone persistence rate	Ranges from 1.0 to 0.9 in increments of 0.02
Range of awareness	Ranges from 40 units to 180 units
Length of simulation run	3000 simulation iterations
Snap-shot frequency	500 simulation iterations
Queue length of nodes	100
Pheromone threshold value	0.1 percent
Pheromone increment rate	3

Table 5.1: Parameter values for equilibrium simulations

5.3 Quantitative analysis of the system

In this section we present the main results of the simulation runs as described above. We do this by plotting the main measurements we have made of the system against the values of the seeding parameters (i.e. the position in configuration space). The main measurables are

- The average degree,
- The percentage of the ants sent out that make it back to the node of origin,
- The average path length of the ants between source and destination nodes,
- The diameter of the graph — this is calculated by taking the maximum of the path length distribution created at the end of that particular run.
- The KL distance — this measures the distance of the network topology from a random graph [5].

[5] The KL distance is different from the other values measured, as the value is derived from a statistical analysis of the degree distribution of the graph generated by the simulation calculation. This is in contrast to the other measurements which are taken directly from the graph generated by the simulation calculation.

Each simulation run was repeated five times, and the results of the measurables were averaged. This repetition was necessary because the stochastic nature of the ant algorithm meant that the average behaviour of the algorithm could not be determined from one run alone. The random number generator was re-seeded between each replication. In total approaching 1000 simulation runs were executed in order to produce the graphs detailed here. The following subsections detail the results. Each simulation run took between two and three days to complete.

5.3.1 Results for the average degree of the node in equilibrium conditions

The degree of a node is defined as the number of other nodes it has direct connection to in one hop. A graph then, can be signified by a degree distribution, which is the plot of the degree against the proportion of nodes in the graph which have that degree. Graphs generated by peer-to-peer systems are directional. A link from node A to node B does not automatically imply that there is a corresponding link from node B to node A. The degree distribution produced when considering links that start at the nodes in the graph is called *the out-degree distribution*. This is different from *the in-degree distribution* which is the degree distribution produced when considering all the links that terminate on the nodes in the graph.

The figures presented here (5.2, 5.3, 5.4, 5.5, 5.6) detail how the average out-degree of the nodes changes as the range of initial awareness, and the pheromone persistence rate are varied. Each figure has been generated using a value for the percentage of nodes that can send out ants, which is varied as described above.

In the following subsections we analyse the effect of the seeding parameters, namely, the initial radius of awareness of the nodes, the pheromone persistence rate, and finally, the percentage of nodes sending out ants.

5.3.1.1 Effect of the initial radius of awareness of the node

Consideration of the five Figures 5.2, 5.3, 5.4, 5.5, and 5.6 shows that in all cases as the initial radius of awareness increases the average degree of the nodes also increases. This can be explained as follows. As the initial radius of awareness of a central node increases the number of potential neighbours of that central node increases. This is reflected in the graphs.

5.3.1.2 Effect of the pheromone persistence rate

In all cases the pheromone persistence rate seems to have a negligible effect on the average degree of the nodes. The only small effect seems to be that at high initial radius of awareness, as the pheromone decay rate decreases (i.e. persistence rate approaches 1) there is an elevation in the average degree of the node beyond the trend set in the other parts of the configuration space. This is explained by observing that as the pheromone persistence approaches 1 there is

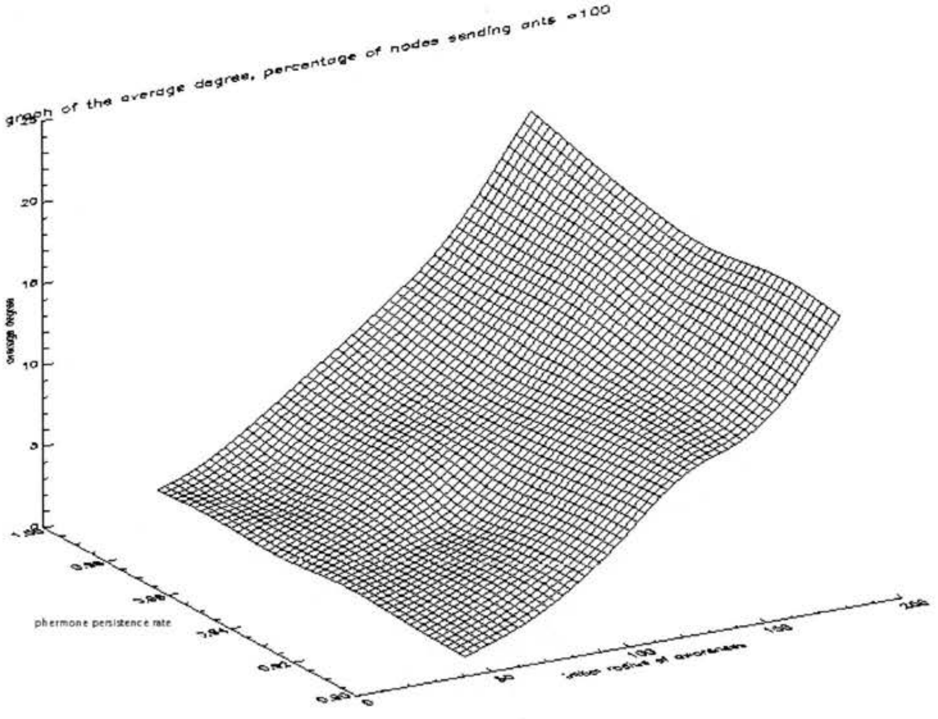


Figure 5.2: Average degree for 100 percent nodes sending ants under equilibrium conditions, x axis initial radius of awareness, y axis pheromone persistence rate, z axis average degree

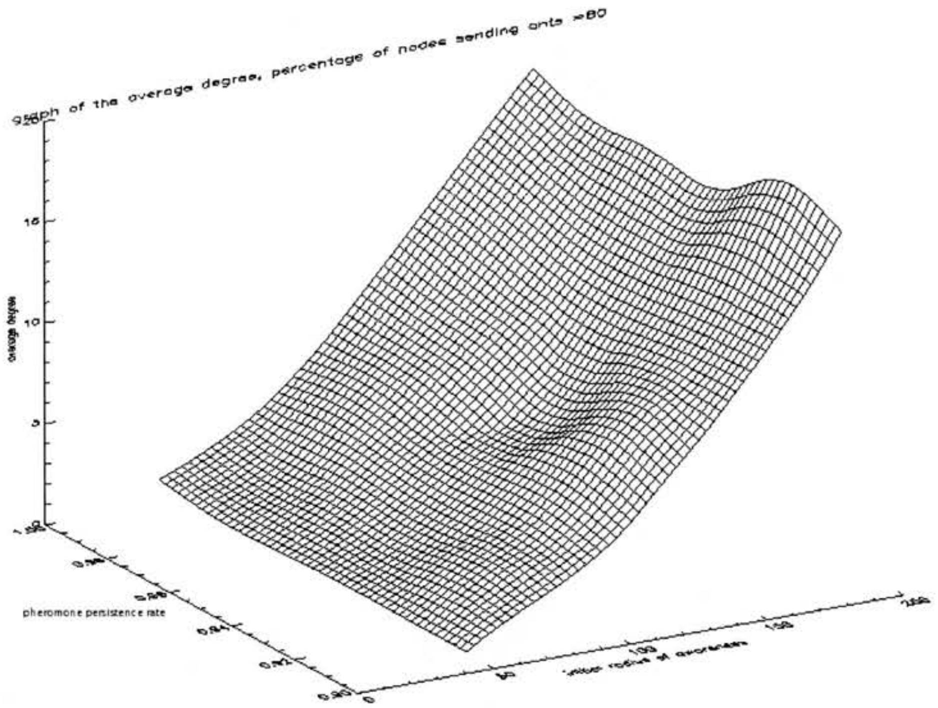


Figure 5.3: Average degree for 80 percent nodes sending ants under equilibrium conditions, x axis initial radius of awareness, y axis pheromone persistence rate, z axis average degree

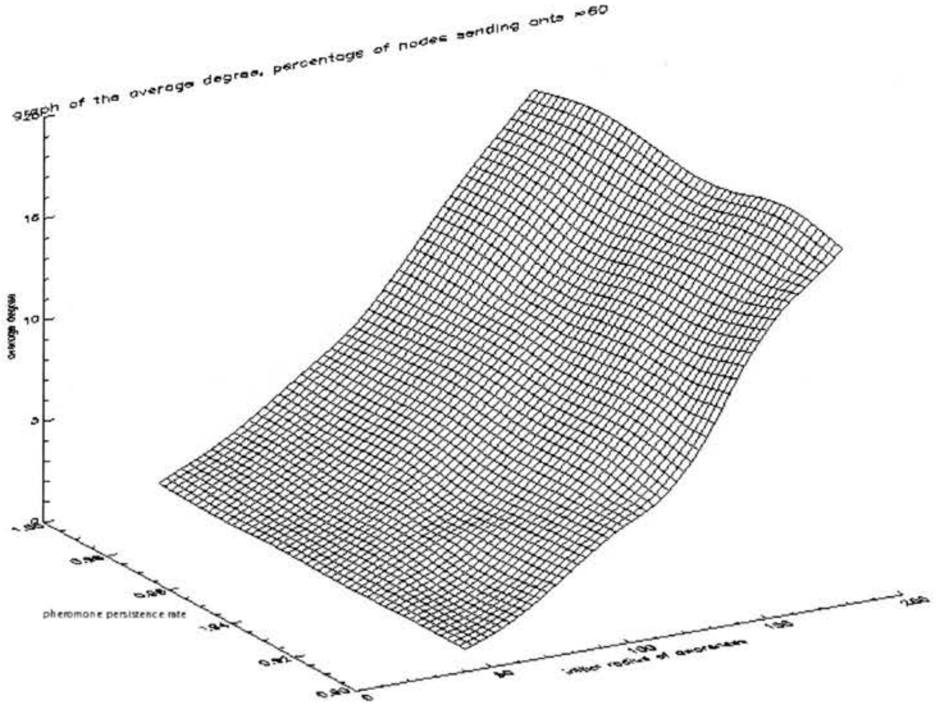


Figure 5.4: Average degree for 60 percent nodes sending ants under equilibrium conditions, x axis initial radius of awareness, y axis pheromone persistence rate, z axis average degree

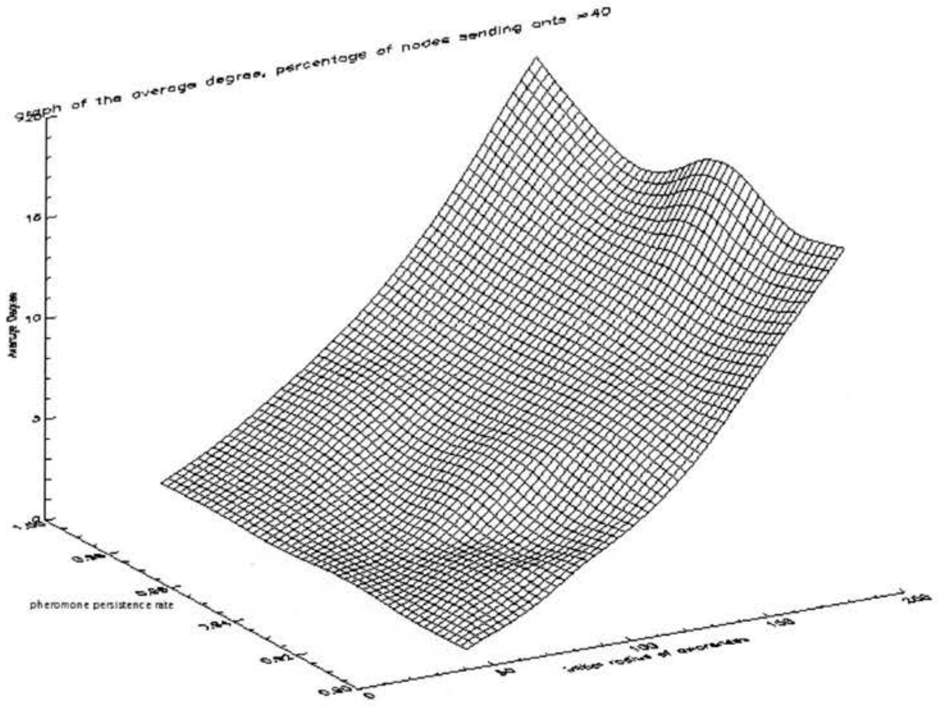


Figure 5.5: Average degree for 40 percent nodes sending ants under equilibrium conditions, x axis initial radius of awareness, y axis pheromone persistence rate, z axis average degree

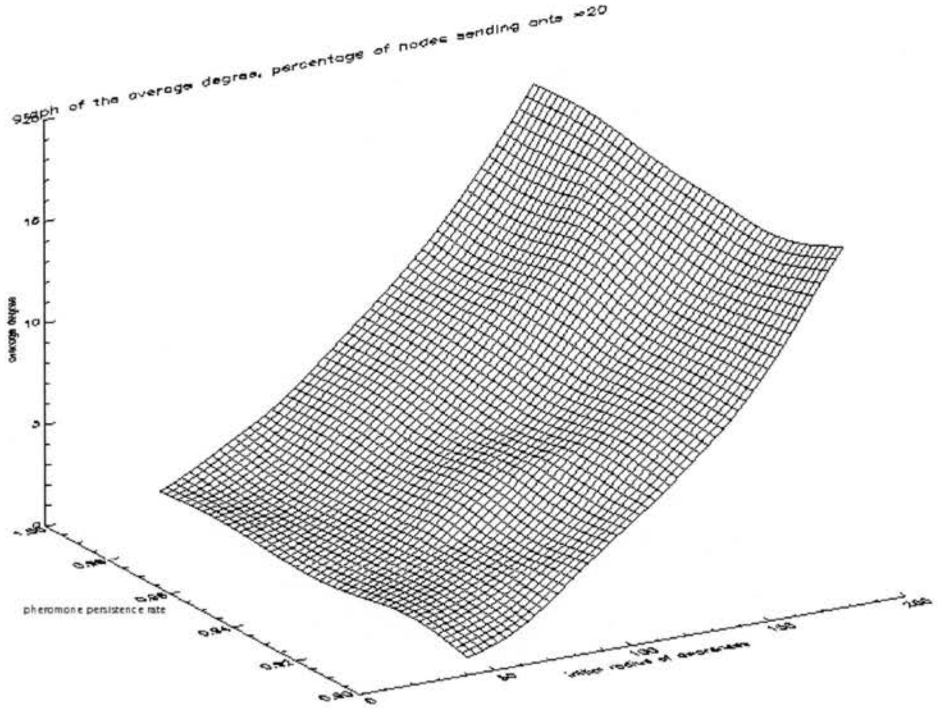


Figure 5.6: Average degree for 20 percent nodes sending ants under equilibrium conditions, x axis initial radius of awareness, y axis pheromone persistence rate, z axis average degree

little or no loss of information in the routing tables of the nodes. This means that any given node will have a larger number of entries on average in its routing table, which in turn will elevate the average degree of the nodes in the system. The average number of entries in the routing tables ranged from four to eighteen depending on the value for the initial radius of awareness.

5.3.1.3 Effect of the percentage of nodes sending out ants

As the percentage of nodes sending out ants in the system decreases from 100 percent to 20 percent we see in general a slight reduction in the average degree. This slight reduction can be explained if we consider the circulation of traffic in our peer-to-peer system enabling a renewal of the pheromone levels within the links of our peer-to-peer system. As the percentage of nodes sending out ants each iteration reduces, the number of ants laying down pheromone in the peer-to-peer system reduces, and the extent to which the links in our graph are renewed diminishes. As a decay process is applied to the pheromone levels in our system every iteration of the simulation, routes that are not renewed will be pruned. As the renewal process is weakened by moving from 100 percent of nodes sending out ants to 20 percent, the decay process will dominate and this means that the average degree of the nodes in our system will be reduced.

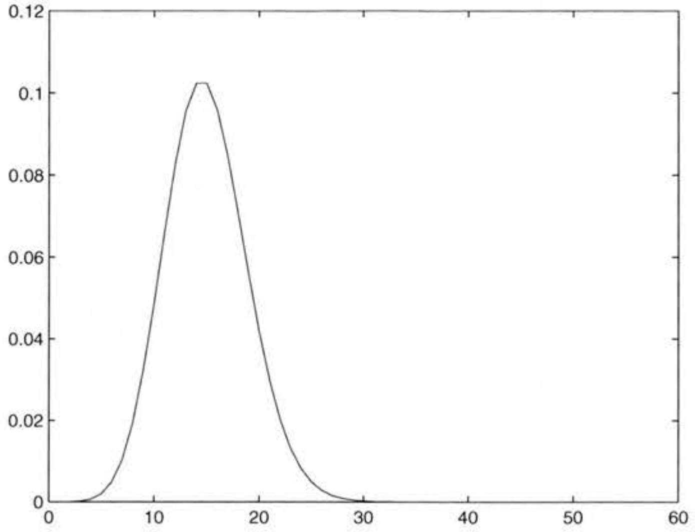


Figure 5.7: A sketch of a Poisson degree distribution

5.3.1.4 Average degree experiments conclusion

The behaviour of the average degree of our system is remarkably clear. Initial radius of awareness has the greatest effect of increasing the average degree with increasing radius of awareness. Pheromone persistence rate has negligible effect, and finally, as we reduce the percentage of nodes sending out ants this has a small effect of reducing the average degree of the nodes.

5.3.1.5 A note on the qualitative nature of the degree distribution

During the simulation experiments it was noted that although (as documented in Section 5.3.1) there is a relatively simple relationship between the input parameter values and the average degree of the network, the functional form of the degree distribution changes markedly with the input parameter values. During this project attempts to generate a statistical model that accommodated all the varying forms of the degree distribution were made. However, this proved to be very difficult. In this section we present a qualitative description of the varying forms of the degree distribution and also a qualitative discussion of how these forms may arise.

We note that when the simulation runs are initiated there is very little structure in the network. This is characterised by a degree distribution that is Poisson in nature. This is confirmed by modelling the degree distribution with Gaussian Mixture Models, and calculating the KL distance between the resulting Gaussian Mixture Model distibution, and a Possion distribution with the same average value. We note that the distance is small. We indicate this in Figure 5.7.

From observations of the degree distributions, modelled using Gaussian Mixture Models,

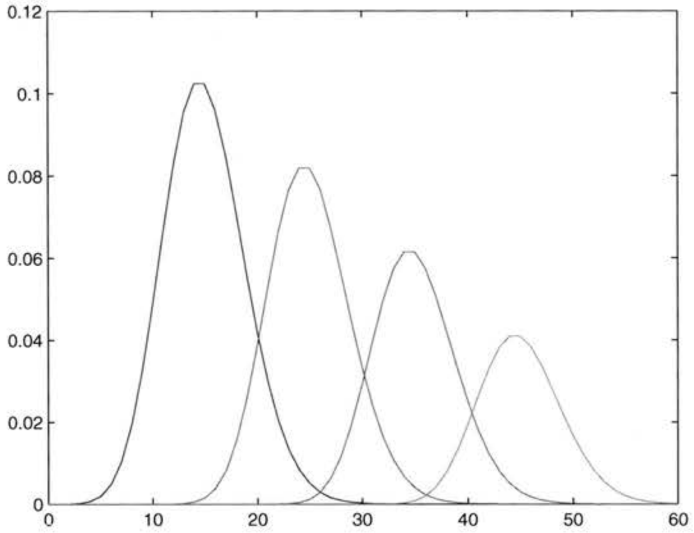


Figure 5.8: A sketch of a Poisson degree distribution with additional peaks

we observe that, as the system evolves, additional structure is seen which is reflected in the degree distribution in the form of peaks at average degrees at larger degree values than the average degree of the initial Poisson distribution. In Figure 5.8 we sketch an example of a degree distribution which is based on a Poisson distribution with additional peaks.

In this way we observe that the system moves from a Poisson distribution to a bi-modal and then a multi-modal distribution as the parameter values driving the simulations change. We also observed that the sensitivities of the form of the degree distribution to changes in the driving parameters were different. We list the effects of changing the parameters here.

- Initial radius of awareness — an increase in the initial radius of awareness caused a substantial increase in the number of peaks in the final degree distribution of the system. The development of these additional peaks was hastened as the initial radius of awareness increased.

- Pheromone persistence rate — in general as the pheromone persistence rate moved from 1 (no pheromone decay) to 0.9 (substantial pheromone decay), the amount of additional structure in the degree distribution lessened. This can be easily understood in terms of the decay of detailed routing information in the routing tables of the nodes of the peer-to-peer system. The faster the routing information decays, the less structure can be supported in the network, so the closer to a random graph the network becomes, and the closer to a Poisson distribution the degree distribution becomes.

- Percentage of nodes sending out ants — the effect of this parameter was not as simple as that of the initial radius of awareness and pheromone persistence rate. The equilibrium simulations were executed using release 1 of the ant peer-to-peer simulator. In release 1 the number of ants released into the system each simulation iteration was tied to the percentage of nodes sending out ants in the peer-to-peer system. As the percentage of nodes sending out ants rose from 20 percent, the amount of pheromone being laid down in the system increased. This meant that there was an increasing amount of information being deposited in the routing tables of the nodes in the peer-to-peer system. However, at very high values of percentage of nodes sending out ants (namely 100 or 80 percent) congestion effects in the network prevent ants from reaching their final destination node, and also prevent information from being deposited in the routing tables of the nodes. So at high percentages of nodes sending out ants we see a decrease in network structure, which is reflected in a reduction in the structure of degree distribution of the peer-to-peer system. The two opposing effects of increasing the amount of pheromone being laid down and also increasing congestion as the percentage of nodes sending out ants increases, create an optimum at around 60 percent of nodes sending out ants. At 60 percent of nodes sending out ants, we see that there is a great variety in the forms of the degree distribution, whereas departures from this value (either lower, or higher) reduce the variety in the degree distribution forms for the reasons stated above.

We also note that for initial radius of awareness values lower than the connection threshold (as defined in Section 5.2.2) the degree distribution is Poisson, indicating a random graph and also indicating a failure for the network to form, as expected under these conditions.

5.3.2 Results for the proportion of ants successfully home in equilibrium conditions

In this section we analyse the results for the proportion of ants that return home safely to the nodes. Figures 5.9, 5.10, 5.11, 5.12, and 5.13 show graphs of the proportion of ants reaching home against the initial radius of awareness and the pheromone persistence rate, as the proportion of nodes sending out ants each iteration within the run varies from 100 percent to 20 percent in steps of 20 percent.

5.3.2.1 Effect of the initial radius of awareness of the node

As the radius of awareness increases the proportion of ants reaching home also increases. This can be explained by considering the conditions under which congestion occurs. A node which has a large radius of awareness will be aware of many neighbouring nodes, and will therefore be able to route incoming ants to a large number of intermediate nodes, and is thereby able to more effectively balance the load of the incoming ants and assist the system in avoiding congestion

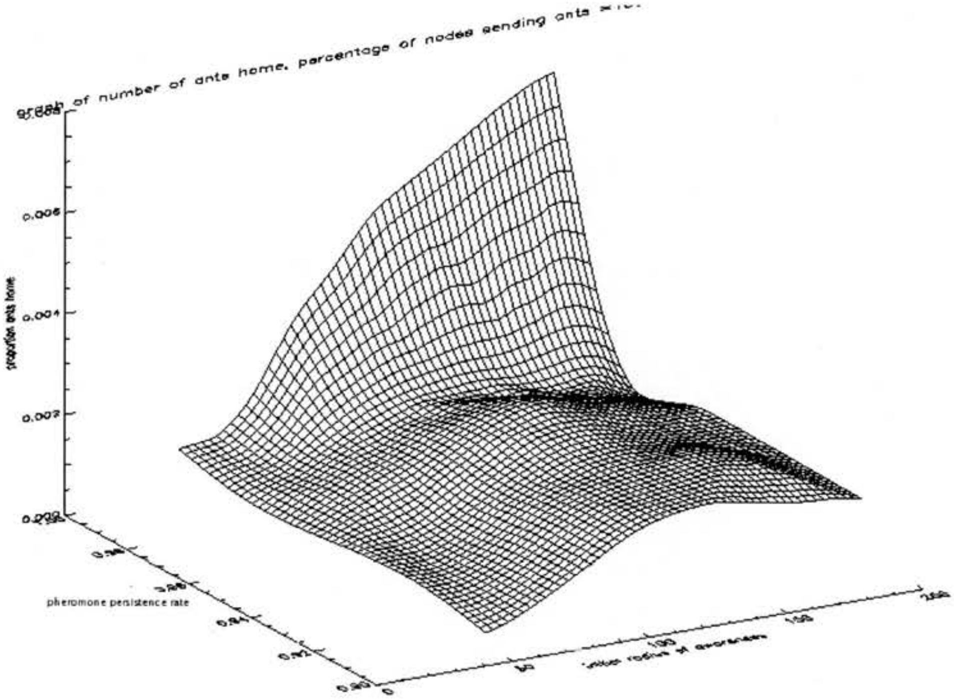


Figure 5.9: Proportion ants getting home for 100 percent nodes sending ants under equilibrium conditions, x axis initial radius of awareness, y axis pheromone persistence rate, z axis proportion of ants getting home

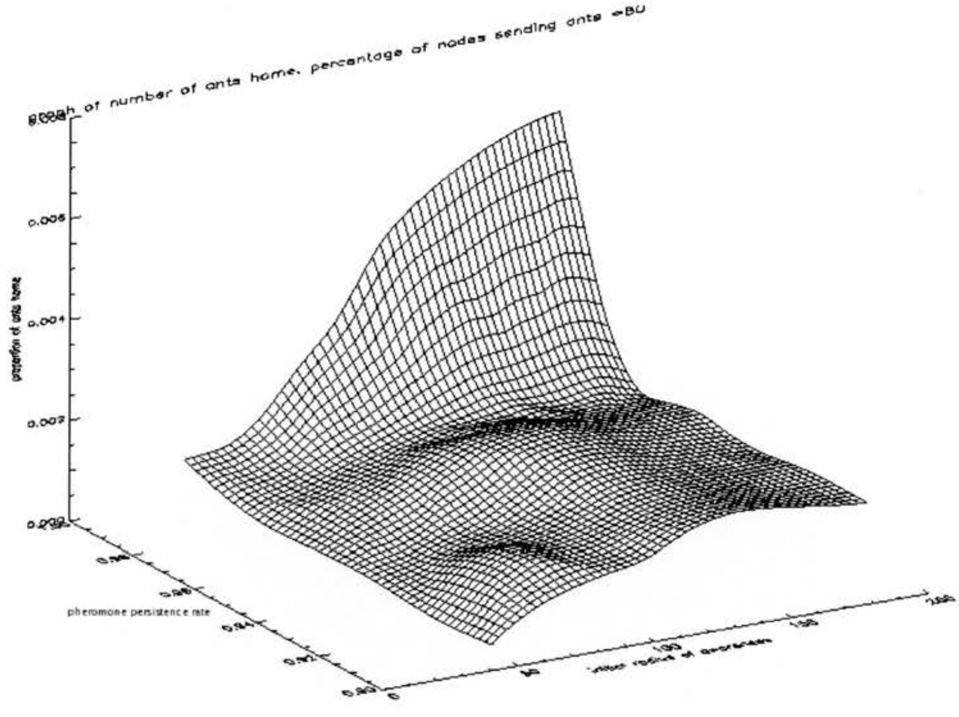


Figure 5.10: Proportion ants getting home for 80 percent nodes sending ants under equilibrium conditions, x axis initial radius of awareness, y axis pheromone persistence rate, z axis proportion of ants getting home

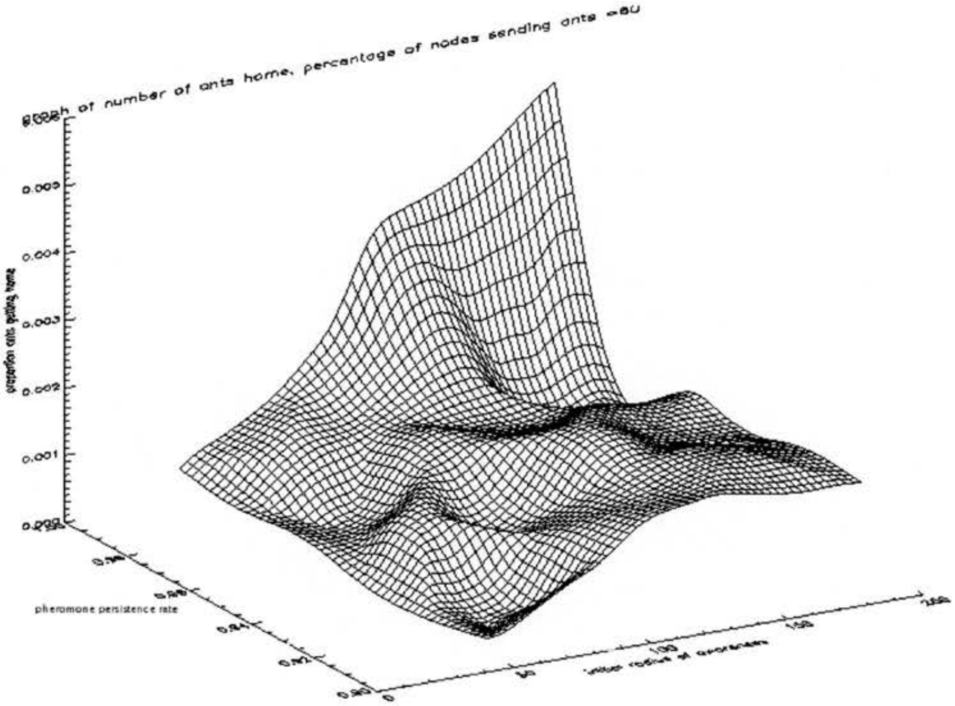


Figure 5.11: Proportion ants getting home for 60 percent nodes sending ants under equilibrium conditions, x axis initial radius of awareness, y axis pheromone persistence rate, z axis proportion of ants getting home

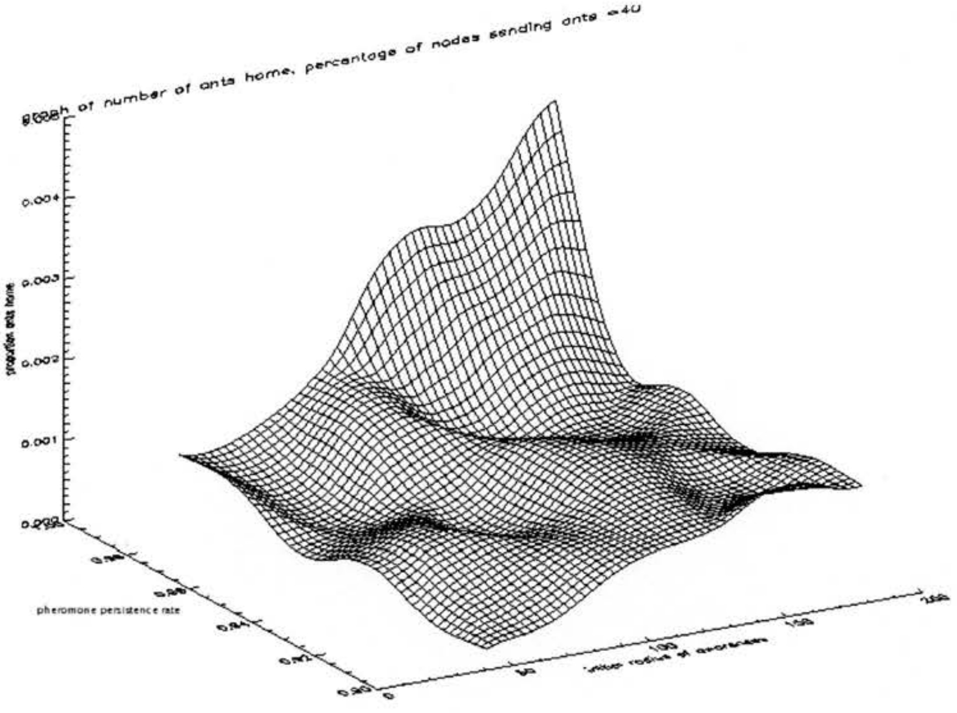


Figure 5.12: Proportion ants getting home for 40 percent nodes sending ants under equilibrium conditions, x axis initial radius of awareness, y axis pheromone persistence rate, z axis proportion of ants getting home

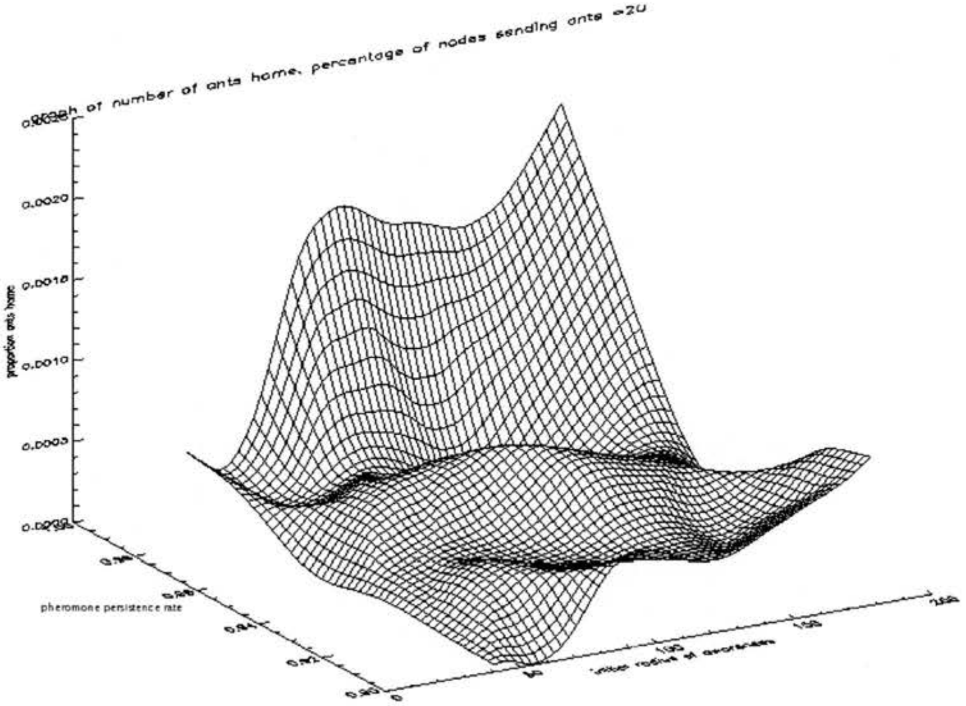


Figure 5.13: Proportion ants getting home for 20 percent nodes sending ants under equilibrium conditions, x axis initial radius of awareness, y axis pheromone persistence rate, z axis proportion of ants getting home

by disseminating the incoming traffic among a larger population of outgoing routes. In our equilibrium experiments congestion is the only cause of ants being dropped from the system. As congestion avoidance becomes more effective due to an increasing radius of awareness of the nodes in the system, fewer ants are dropped. The converse argument is true too. So as the radius of awareness decreases the number of ants dropped by the system increases. This reaches a limit at the connection threshold, which as stated above is the threshold of the radius of awareness below which the nodes cannot see one another on the grid. Below the connection threshold very few ants successfully make it home to their source nodes having visited their destination nodes.

5.3.2.2 Effect of the pheromone persistence rate

For all values of the percentage of nodes sending out ants in our system we see that, in general, as the pheromone persistence rate decreases from 1 there is a reduction in the number of ants reaching home in the system. This can be interpreted using a similar argument to the one explaining the effect of the percentage of nodes sending out ants. As the pheromone persistence becomes more stringent, i.e. reduces from 1, the routing information in the routing tables of the nodes in the system disappears more quickly from iteration to iteration in the simulations. If the routing information disappears completely, then the nodes in the system are capable of doing nothing other than route randomly. Random routing is clearly an ineffective policy for getting ants from their source to destination and back to the source again. Conversely, if there is no decay in the pheromone information contained in the routing tables of the nodes (i.e. pheromone persistence rate is 1) the proportion of ants successfully reaching home rises. An explanation for this rise is that we are able to take advantage of the reinforced routing information laid down by the ants as they pass through to identify efficient routes. Under these conditions inefficient routes will also be re-enforced but are more likely to lead to ants being dropped by the system.

5.3.2.3 Effect of the percentage of nodes sending out ants

The average proportion of ants reaching home decreases as the percentage of nodes sending out ants every iteration decreases from 100 percent to 20 percent. This can be explained by observing that as the percentage of nodes sending out ants decreases the amount of pheromone laid down every iteration decreases. Another way of looking at this is that the renewal process for the routes weakens as the proportion of the total number of nodes that can send out ants decreases. We have stated above that every simulation iteration a decay process runs which reduces the amount of pheromone on all of the links in the graph by a fixed factor. This means that those routes which are not renewed every iteration gradually disappear from the network. As the renewal process weakens due to the reduction in the proportion of nodes sending out ants, the decay process strengthens in relative terms. This in turn means that there is less structure in the graph encoded by the pheromone levels, which in turn means that the routing information is likely to be random. Random routing is less effective than directed routing, and fewer ants effectively get through to their destinations.

5.3.2.4 Proportion of ants successfully reaching home: conclusion

We see that in general in the equilibrium scenario, the proportion of ants successfully reaching home having completed a tour to a destination node and back again, can be optimised by the following actions

- Increasing the percentage of nodes that initiate ants every iteration and therefore increasing the number of ants in the system. In the equilibrium experiments we cannot tell which has the greater effect due to the lack of fine control in the control surface of this version of the simulator;
- Placing the pheromone persistence rate as close to 1 as possible (i.e. no pheromone decay);
- Increasing the initial radius of awareness of each node as much as possible.

Note that these optimisation criteria are only valid in the equilibrium scenario, where once set at the beginning of the simulation, there is no change in the demand function, nor any change in the number of nodes sending out and routing ants from iteration to iteration in the simulation.

5.3.3 Results for the average path length of the ants in equilibrium conditions

In this section we detail the effect of the main parameters on the average path length between source and destination travelled by the ants in the system that have completed their tour. Figures 5.14, 5.15, 5.16, 5.17, and 5.18 show how the path length changes as the main parameters of the system are varied.

5.3.3.1 Effect of the initial radius of awareness of the node

Consideration of the figures for 100 percent nodes sending ants through to 40 percent nodes sending ants (Figures 5.14, 5.15, 5.16, 5.17) indicates that the average path length has little response to the initial radius of awareness of the nodes. The average path length remains around two hops indicating that the majority of the trips the ants make are short. This is true until very high pheromone persistence rates (persistence rate = 0.98). Above this value we see that as the radius of awareness increases the average path length begins to increase. When the

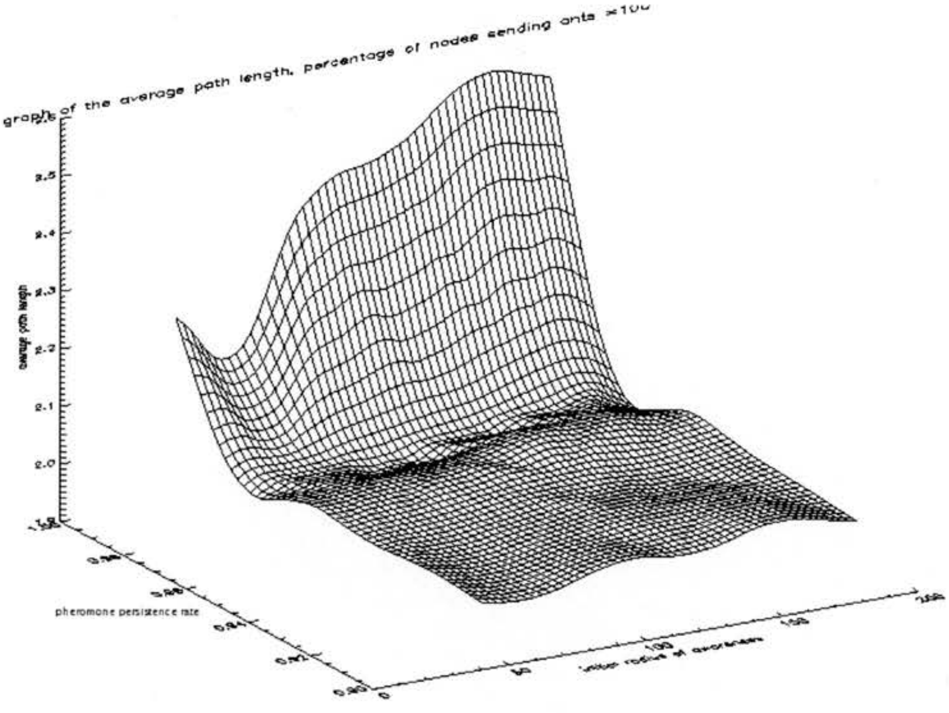


Figure 5.14: Average path length for 100 percent nodes sending ants under equilibrium conditions, x axis initial radius of awareness, y axis pheromone persistence rate, z axis average path length

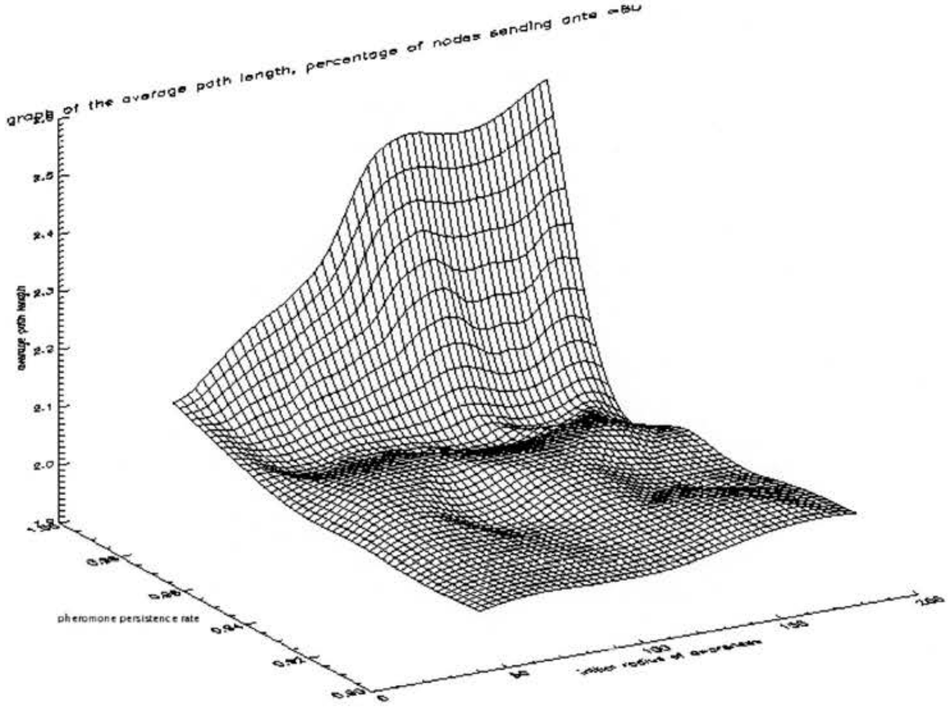


Figure 5.15: Average path length for 80 percent nodes sending ants under equilibrium conditions, x axis initial radius of awareness, y axis pheromone persistence rate, z axis average path length

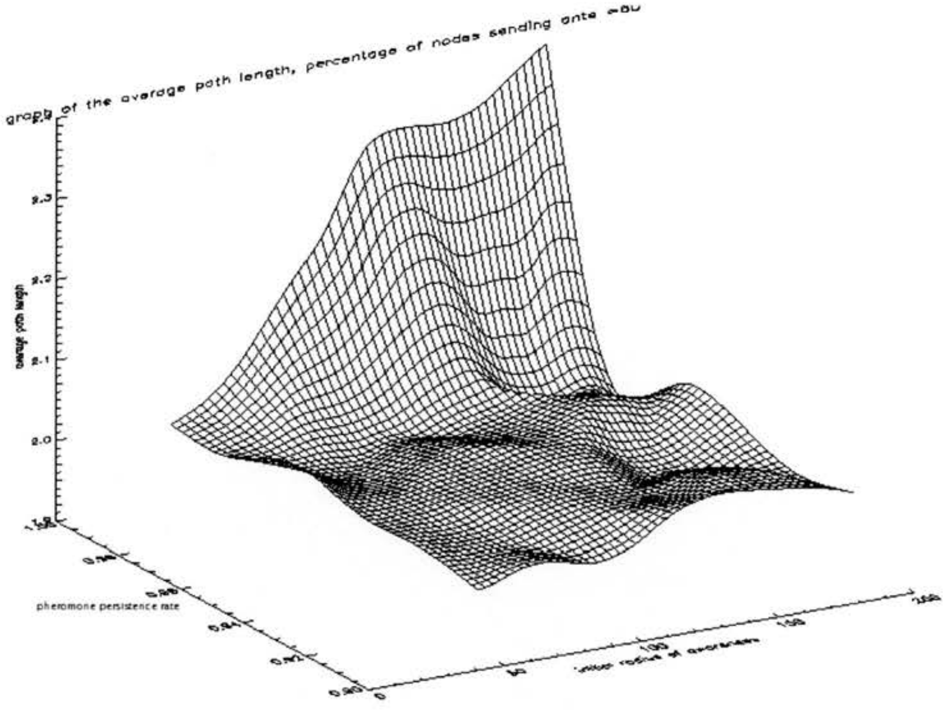


Figure 5.16: Average path length for 60 percent nodes sending ants under equilibrium conditions, x axis initial radius of awareness, y axis pheromone persistence rate, z axis average path length

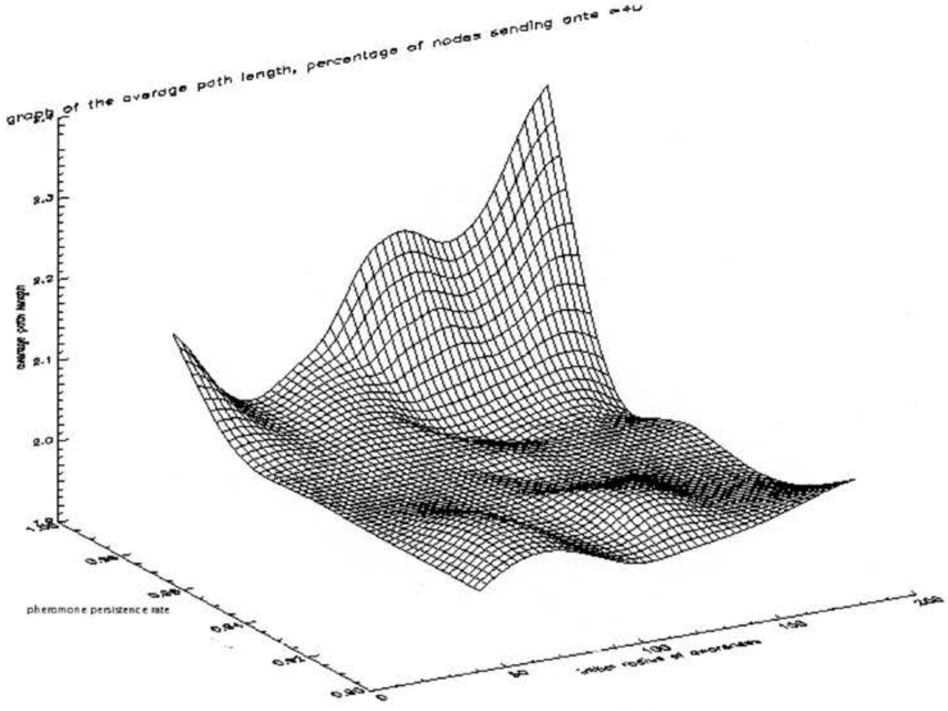


Figure 5.17: Average path length for 40 percent nodes sending ants under equilibrium conditions, x axis initial radius of awareness, y axis pheromone persistence rate, z axis average path length

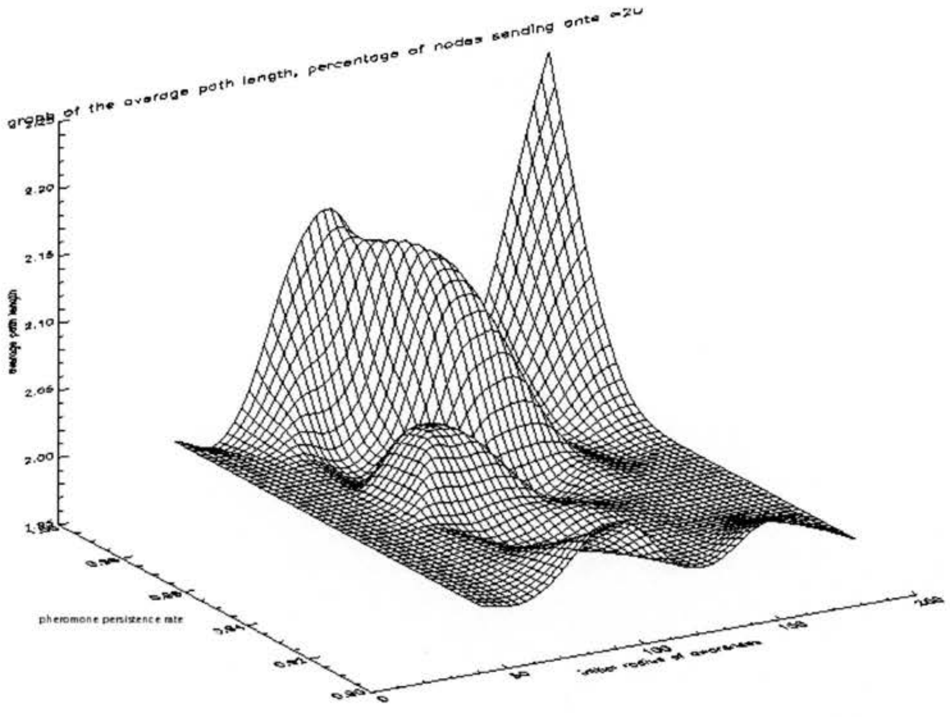


Figure 5.18: Average path length for 20 percent nodes sending ants under equilibrium conditions, x axis initial radius of awareness, y axis pheromone persistence rate, z axis average path length

value of the pheromone persistence rate approaches 1 the nodes effectively do not forget routes from iteration to iteration. Under these conditions routes are not pruned from the routing tables of the nodes. We must note that the pheromone persistence rate must be very close to 1 in order for this effect to be observed. Further the closer to 1 the pheromone persistence rate becomes the greater the extent of the rise in the average path length.

Consideration of the average path length for 20 percent nodes sending out ants (Fig 5.18) reveals two competing effects. When there is little pheromone decay in the system (pheromone persistence rate close to 1) we see that as we increase the initial radius of awareness of the nodes the average path length increases. However, there is an abrupt down turn in the average path length of the nodes when the initial radius of awareness of the nodes exceeds 90 units. An explanation for this could be that under the conditions of high radius of awareness the ants leaving a node do so in a diffuse fashion as there are many potential neighbouring nodes that the ants can be routed to. This diffusion of traffic causes the average path length to reduce. This reduction counteracts the propensity of the system to increase the average path length when the pheromone persistence rate approaches 1 for the reasons stated above. This is true up to a radius of awareness (in this system around 150 units). Beyond this routes originating from different nodes begin to significantly overlap, and ants originating from different nodes block one another's path through the network. The increased congestion at very high initial radii of awarenesses removes the availability of short path length routes causing the ants to travel over longer path lengths, and increasing the value of the average path length of the system.

5.3.3.2 Effect of the pheromone persistence rate

As stated in the previous section we see that the effect of the pheromone persistence rate on the average path length of the system is negligible up until values of 0.98. Above 0.98 the pheromone persistence rate causes an increase in the average path length of the ants in the system. An explanation for this is that under these conditions inefficient routes are not pruned from the routing tables of the nodes in the system. This in turn causes the system to rely on inefficient (high path length) routes when the more efficient routes become congested.

5.3.3.3 Effect of the percentage of nodes sending out ants

The general effect of the reduction in the percentage of nodes sending out ants is to reduce the maximum of the average path length from 2.8 (100 percent nodes sending out ants) to 2.25 (20 percent nodes sending out ants). The explanation for this is that as the percentage of nodes sending out ants decreases there is less congestion in the network as a whole, and the more efficient routes (low path length routes) are able to carry a higher proportion of the traffic through the network.

5.3.3.4 Average path length: conclusion

We have seen that the initial radius of awareness of the nodes has little effect on the average path length unless there is little pheromone decay (i.e. pheromone persistence rate is close to 1). We have also seen that as we increase the pheromone persistence rate (i.e. pheromone decay is close to 1) the average path length increases in general. Further, we have seen that as the percentage of nodes sending out ants (and therefore also the average traffic level in the network) decreases, the average path length also decreases. In order to optimise the average path length of the system we require the following conditions to prevail:

- pheromone persistence rate should be as high as possible (i.e. as close to 1 as possible);
- percentage of nodes sending out ants in the system should be as small as possible, in order to reduce congestion in the intermediate nodes.

5.3.4 Results for the diameter of the graph in equilibrium conditions

We define the diameter of the network here as the maximum path length between source and destination travelled by all of the ants successfully reaching their destination in the network. In this section we consider the effect of the driving parameters of the system, i.e. initial radius of awareness, the pheromone persistence rate, and the percentage of nodes sending out ants, on the diameter of the network. Graphs showing how the diameter of the network responds to the driving parameters of the system are shown in Figures 5.19, 5.20, 5.21, 5.22, and 5.23.

5.3.4.1 Effect of the initial radius of awareness of the node

Up to moderate pheromone persistence rates (0.96) the diameter of the graph remains largely invariant to the initial radius of awareness of the nodes in the system (see figs 5.23, 5.23, 5.23, 5.23, 5.23). Above this threshold the diameter seems to increase with increasing initial radius of awareness. Although this effect is only slight when the proportion of nodes sending out ant is 80 percent and above, the effect does increase markedly as the proportion of nodes sending out ants reduces towards 20 percent. It is under conditions of low pheromone decay rate (i.e. persistence rate close to 1) and also low average traffic (i.e. proportion of nodes sending out ants close to 20 percent) when the system is not congested and also when inefficient routes are not being pruned from the routing tables of the nodes in the system, that long tours of the system are made possible. Outside of these conditions the diameter of the network seems unresponsive to the initial radius of awareness of the nodes.

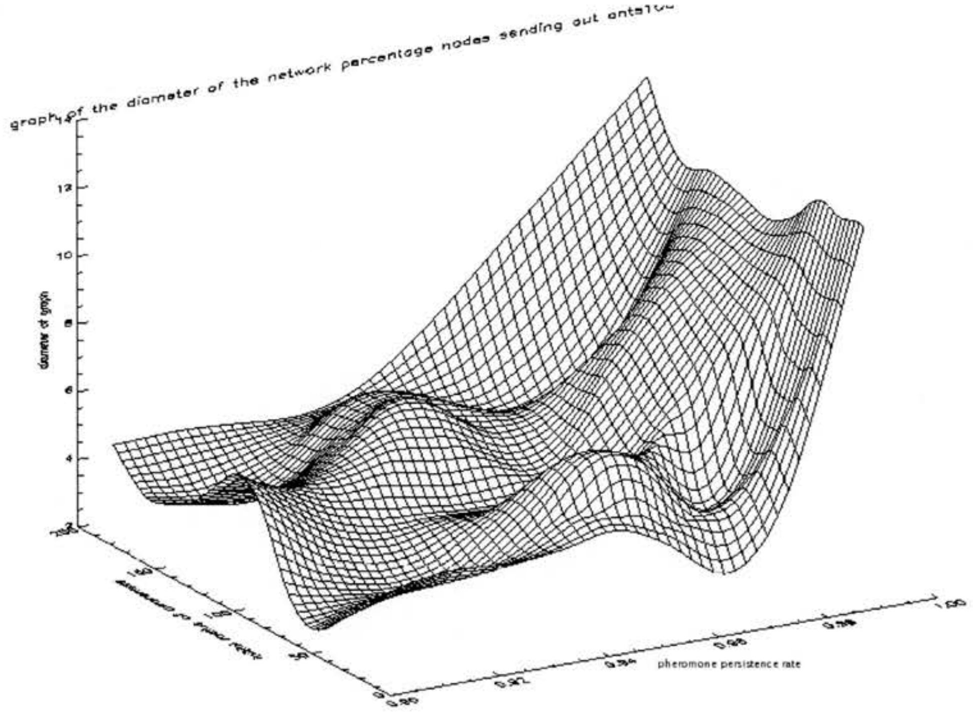


Figure 5.19: Diameter for 100 percent nodes sending ants under equilibrium conditions, x axis initial radius of awareness, y axis pheromone persistence rate, z axis diameter

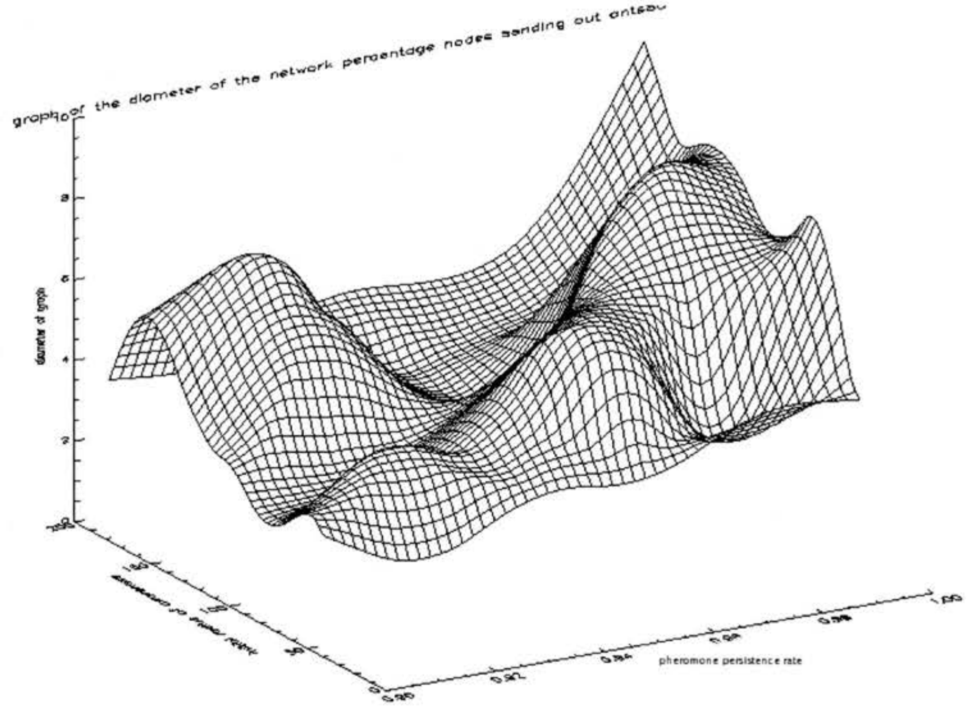


Figure 5.20: Diameter for 80 percent nodes sending ants under equilibrium conditions x axis initial radius of awareness, y axis pheromone persistence rate, z axis diameter

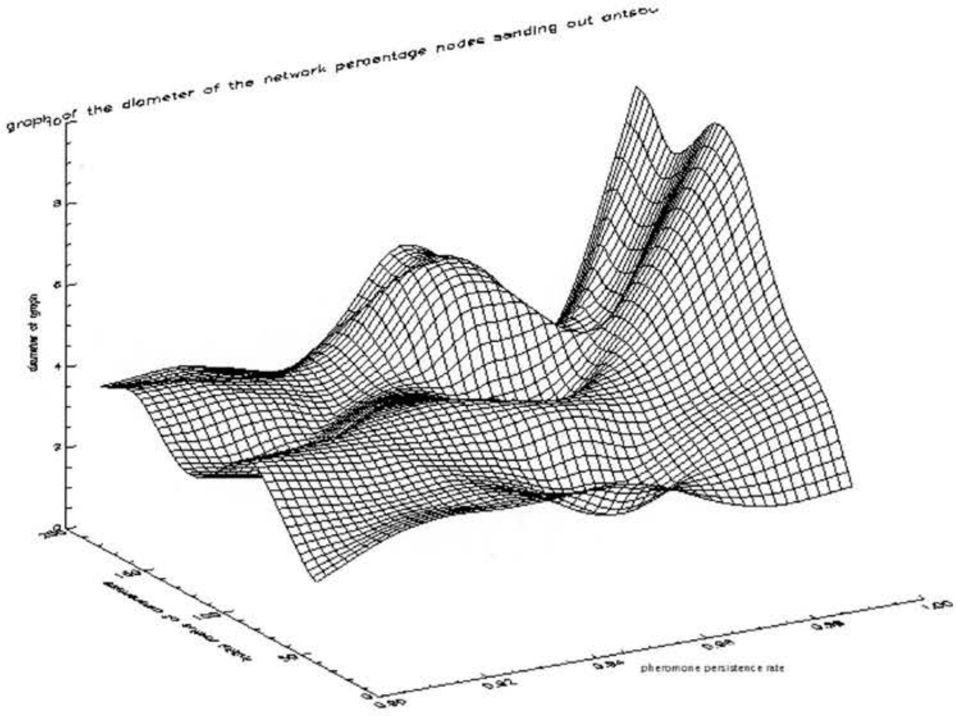


Figure 5.21: Diameter for 60 percent nodes sending ants under equilibrium conditions x axis initial radius of awareness, y axis pheromone persistence rate, z axis diameter

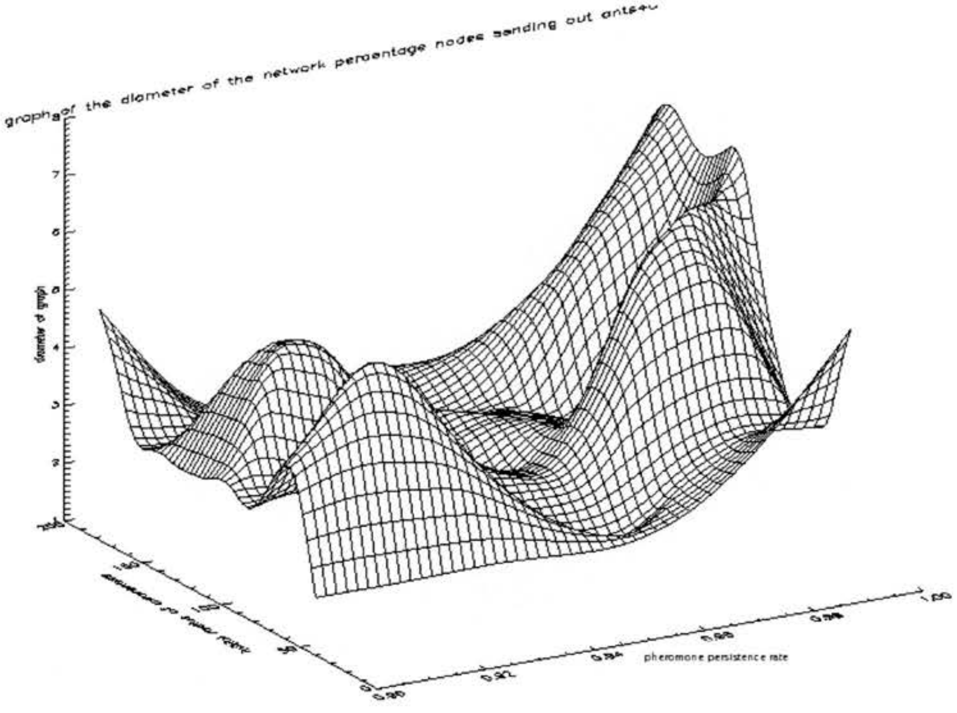


Figure 5.22: Diameter for 40 percent nodes sending ants under equilibrium conditions x axis initial radius of awareness, y axis pheromone persistence rate, z axis diameter

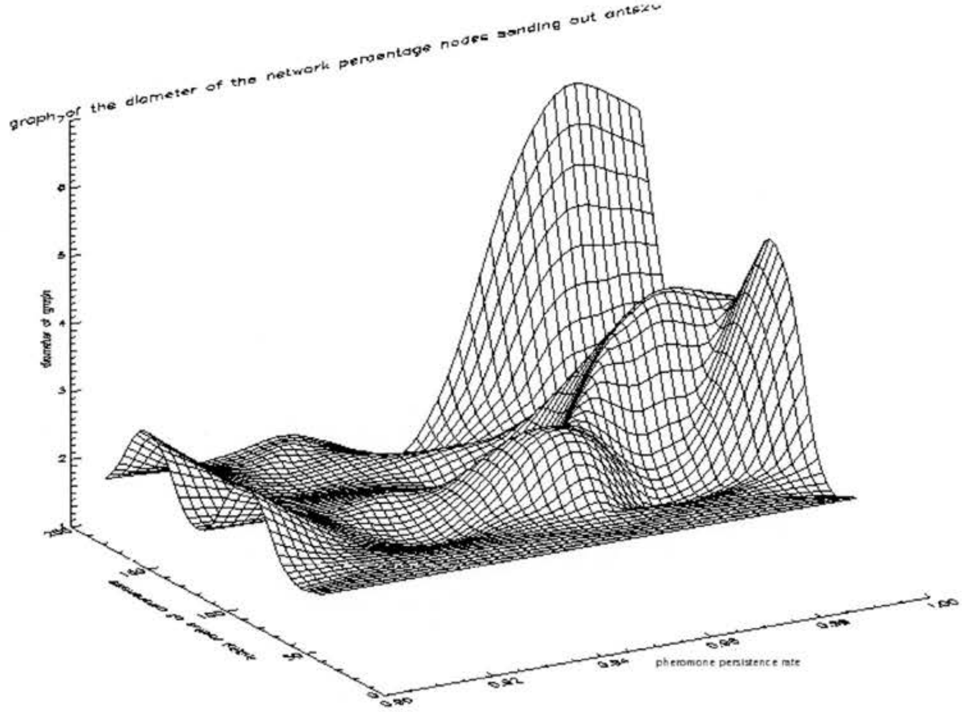


Figure 5.23: Diameter for 20 percent nodes sending ants under equilibrium conditions x axis initial radius of awareness, y axis pheromone persistence rate, z axis diameter

5.3.4.2 Effect of the pheromone persistence rate

In general, a more stringent pheromone decay rate (i.e. persistence rate close to 0.9) seems to reduce the diameter of the network. This can be explained as a more stringent decay enables nodes to prune inefficient routes from their routing tables, and therefore enables a reduction in the overall diameter of the graph.

5.3.4.3 Effect of the percentage of nodes sending out ants

As we increase the proportion of nodes sending out ants (and therefore the average traffic level) the diameter of the network tends to rise from 3 units to 4 units. The diameter also becomes more sensitive to the initial radius of awareness and pheromone persistence rate. This is explained by considering the increased levels of congestion created by the increased proportion of nodes sending out ants.

5.3.4.4 Network diameter: conclusion

In order to minimise the network diameter the following conditions must prevail.

- The proportion of nodes sending out ants must be as small as possible (close to 20 percent);

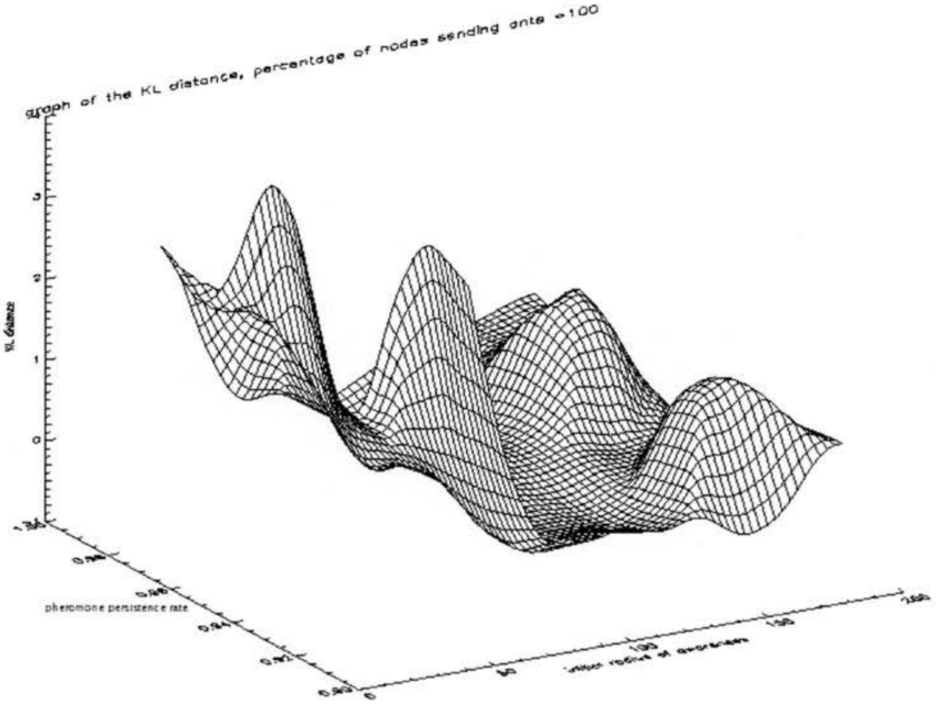


Figure 5.24: KL surface for 100 percent nodes sending out ants x axis initial radius of awareness, y axis pheromone persistence rate, z axis KL value

- The radius of awareness must be above 60 units;
- The pheromone persistence rate must be below 0.96.

5.3.5 Results for the KL surface

In this section we document the results of the Kullbeck-Leibler surface (D_{kl} as described in Section 3.5.2, on page 47). We note that a low value of the D_{kl} distance indicates that the graph that our ant peer-to-peer system has formed is similar to a random graph. Conversely, a large D_{kl} value indicates great difference between a random graph and the topology formed by our ant peer-to-peer system.

In the Figures 5.24, 5.25, 5.26, 5.27, 5.28, below we see the surface formed by the D_{kl} values at different initial radii of awareness and pheromone persistence rates. We analyse the effect of the initial radius of awareness, pheromone persistence rate, and also the percentage of nodes sending out ants individually below.

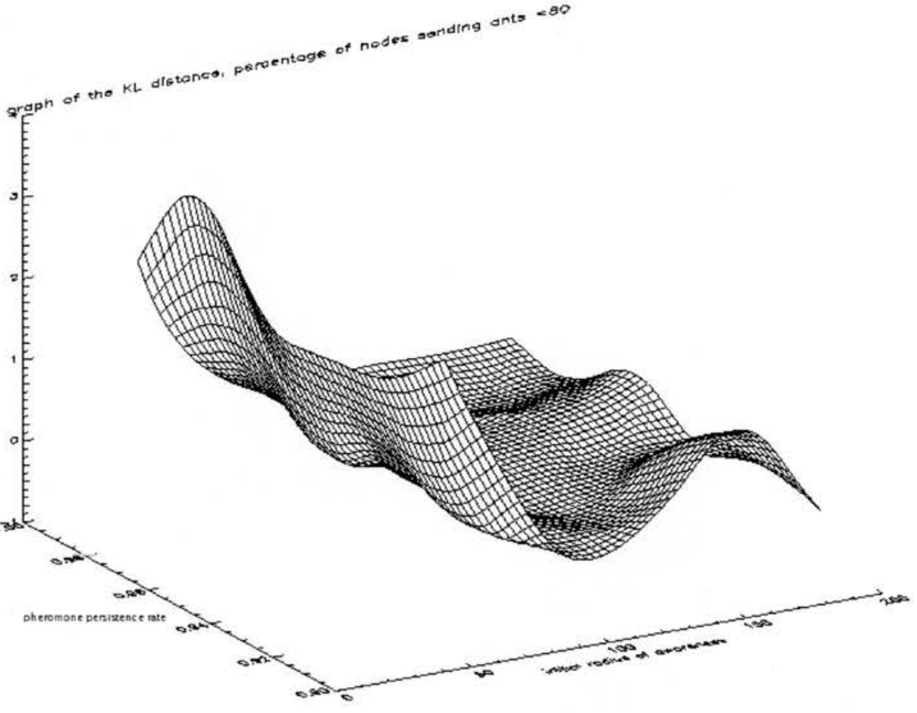


Figure 5.25: KL surface for 80 percent nodes sending out ants x axis initial radius of awareness, y axis pheromone persistence rate, z axis KL value

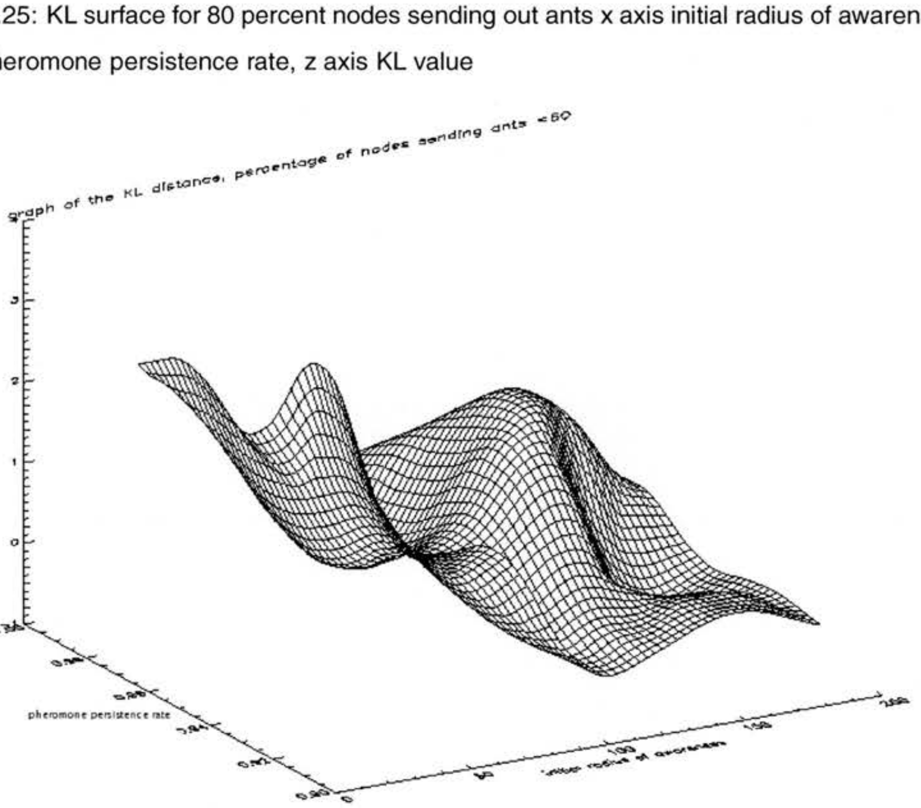


Figure 5.26: KL surface for 60 percent nodes sending out ants x axis initial radius of awareness, y axis pheromone persistence rate, z axis KL value

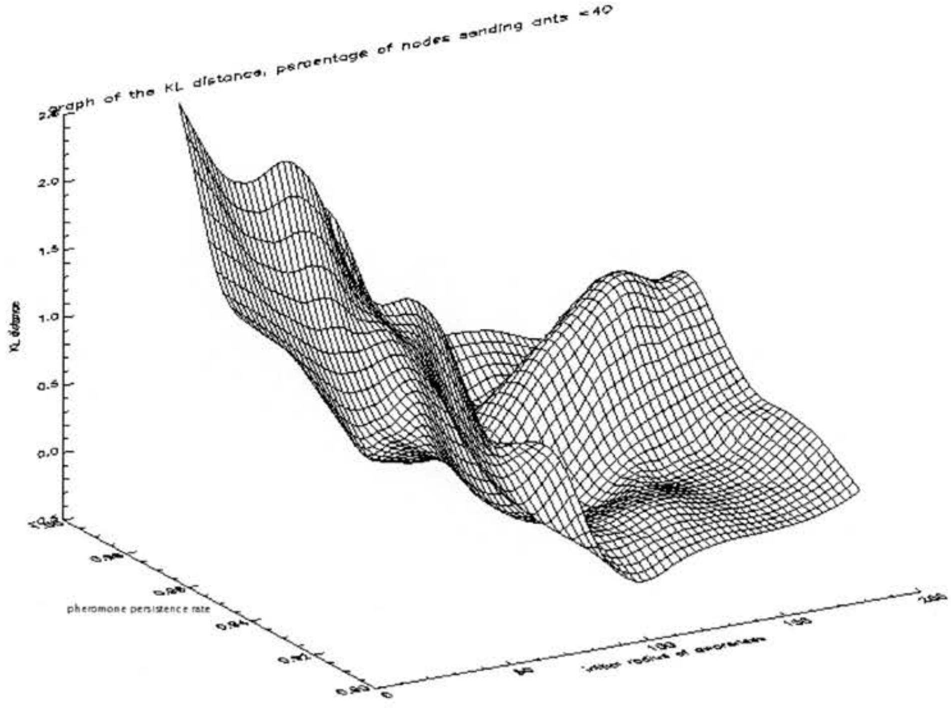


Figure 5.27: KL surface for 40 percent nodes sending out ants x axis initial radius of awareness, y axis pheromone persistence rate, z axis KL value

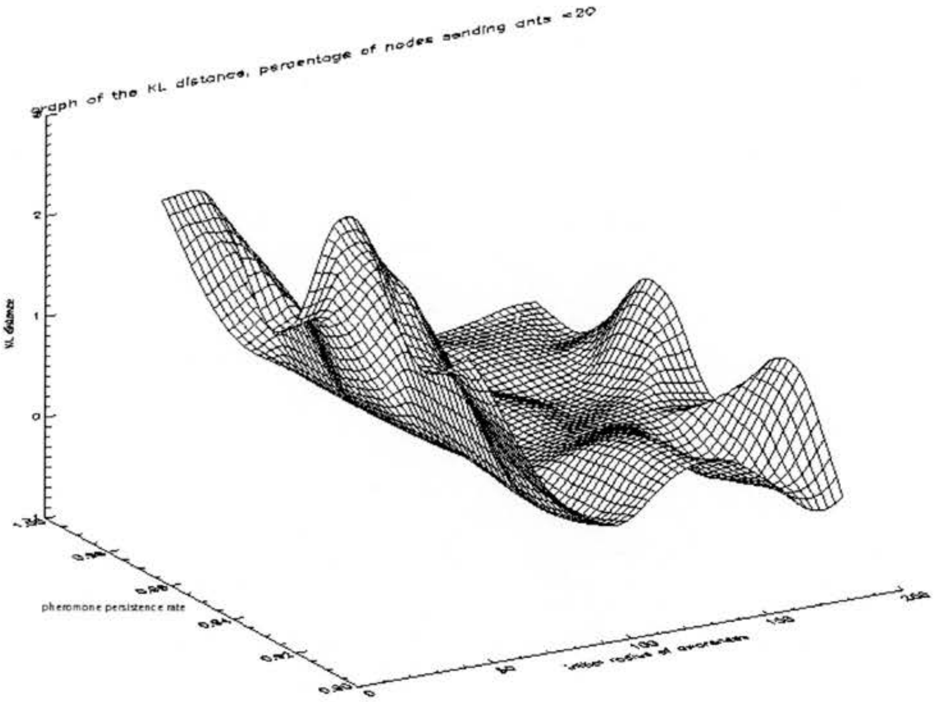


Figure 5.28: KL surface for 20 percent nodes sending out ants x axis initial radius of awareness, y axis pheromone persistence rate, z axis KL value

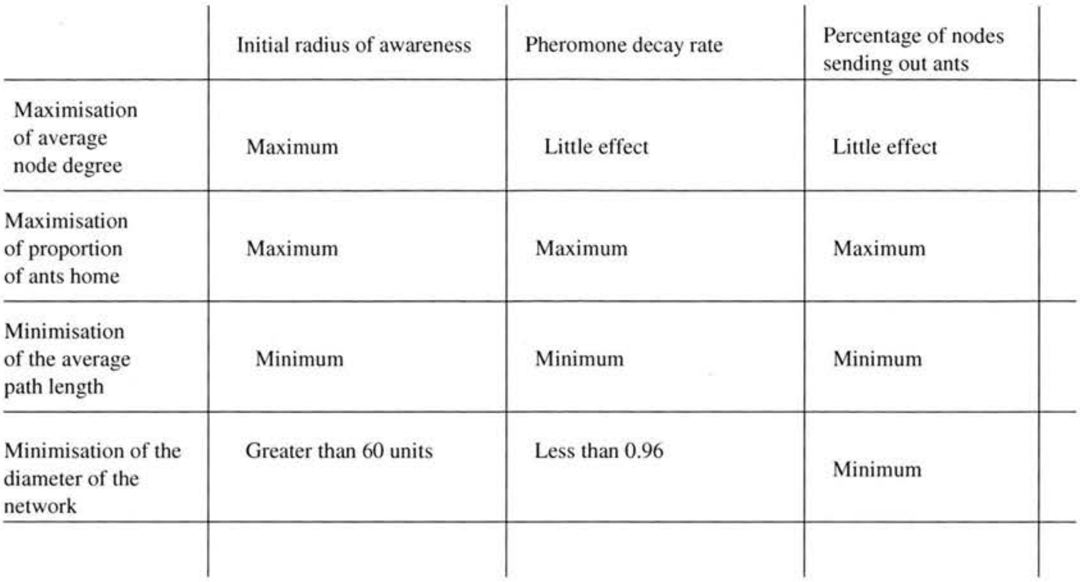
5.3.5.1 Effect of the Initial radius of awareness on the KL surface under equilibrium conditions

In this section we explore the behaviour of the D_{KL} value. The reference distribution used in the D_{KL} calculations is a Poisson distribution with the same mean value as the degree distribution being considered. The mean values of the degree distributions varied as we moved between points in the configuration space of our equilibrium experiments. In general, we see that as the initial radius of awareness of the simulation increases the D_{KL} value decreases. We also see that for very high initial radii of awareness (i.e. in excess of 150 units) the D_{KL} value increases in the case of the percentage of nodes sending out ants being 100, 80, 40 and 20 respectively. The reason for the increased randomness in the graph as the initial radius of awareness increases, is that at higher initial radii of awareness there are, by definition, more routes to choose from, and therefore the graph is more likely to be random. A possible explanation for the increase in the D_{KL} value at high initial radius of awareness and also pheromone persistence rates close to 1, is that under these conditions there is a lot of routing information being built up in the routing tables of the nodes. This is most apparent when the percentage of nodes sending out ants is 100 and 40 respectively. At 100 percent of nodes sending out ants there is a lot of pheromone being laid down in the network and the decay mechanism for pruning inefficient routes may not be working efficiently as a result. At 40 percent nodes sending out ants there is little congestion in the network, which enables routes to be traversed more efficiently and structure to be built up. We note that in the 40 percent case the maximum is at 0.98 pheromone persistence rate, and also 180 initial radius of awareness. This indicates the importance of being able to prune routes from the routing tables.

In general there is also a maximum in the D_{KL} distance when the radius of awareness is 40 units, for all percentages of nodes sending out ants (20,40,60,80,100). A radius of awareness of 40 units is less than the connection threshold for this system (which is 60 units). These large values of the KL distance are caused by disconnected components in the graph as a result of the radius of awareness being lower than the connection threshold.

5.3.5.2 Effect of the pheromone persistence rate on the KL surface under equilibrium conditions

In general we see that as the pheromone persistence rate moves from 0.9 to 1 the value for D_{KL} increases. This is because the routing information in the routing tables is pruned less when the pheromone persistence rate approaches 1. We see that the effect of raising D_{KL} by raising the pheromone persistence rate is most marked at high initial radii of awareness. We discount the high D_{kl} values at low radii of awareness as these values are beneath the connection threshold, and are therefore of limited use in the analysis of our ant peer-to-peer system.



	Initial radius of awareness	Pheromone decay rate	Percentage of nodes sending out ants
Maximisation of average node degree	Maximum	Little effect	Little effect
Maximisation of proportion of ants home	Maximum	Maximum	Maximum
Minimisation of the average path length	Minimum	Minimum	Minimum
Minimisation of the diameter of the network	Greater than 60 units	Less than 0.96	Minimum

Figure 5.29: Summary of results for equilibrium experiments

5.3.5.3 Effect of the percentage of nodes sending out ants on the KL surface under equilibrium conditions

At 100 and 80 percent of nodes sending out ants, we see that there is a very sharp fall in the value of D_{KL} as the radius of awareness increases, and also a flat response to changes in the pheromone persistence rate. An explanation for this is that congestion in the system due to the traffic levels prevents successful tours from being completed by the ants. This in turn means that little valid routing information is built up on the nodes within the peer-to-peer system. At 20 percent of nodes sending out ants we also see a sharper decay in the value of D_{KL} as the radius of awareness increases. This can be explained by considering that little pheromone is being laid down in the network so routing information in the routing tables is being eroded. The window of between 40 and 60 percent of nodes sending out ants seems to offer the greatest persistence of structure especially at high radii of awareness. This is because at these traffic levels, there is both enough pheromone being laid down and also little congestion in the system enabling the efficient exchange of routing information.

5.4 Equilibrium experiments conclusion

Through our experimental observations we have shown that the conditions needed in order to optimise our measurable quantities differ according to the quantity being optimised. We summarise the conditions in Figure 5.29.

We note that optimisation of the topological metrics (namely path length and diameter) require opposing conditions to optimisation of traffic statistics (namely the proportion of ants reaching home). An explanation for this difference it is that under conditions of route diversity (i.e. high average degree) that traffic statistics are optimised. However, not all the routes in that diverse environment are efficient. Conversely, just because the routes are efficient (i.e. have a short path length and diameter) does not necessarily mean that a large proportion of the ants in the system are able to exploit them and get through to their destination effectively. In the following chapters on burstiness, and damage, we explore further the properties of the ant peer-to-peer system under the conditions where the routes are short, but the system is not acting efficiently.

Chapter 6

Results of non-equilibrium behaviours concentrating on the effect of bursty data

6.1 Introduction to the experiments and motivation

In this chapter we focus on the effect of burstiness in traffic on the behaviour of our ant peer-to-peer system. We do this by keeping the average demand level the same while varying the long and short term dynamics in the time series which describes the demand on the system. A further intention of these experiments is to investigate how the behaviour of the ant colony optimisation (ACO) algorithm can be adapted in order to increase performance of the overall peer-to-peer system under differing bursty load conditions. For this purpose we also explore the effect of varying the pheromone persistence rate in our ant peer-to-peer system for different categories of burstiness. Specific experiments have been designed in order to support both objectives: these havc been described in Section 6.3. The time series has been generated using the traffic dynamics simulator detailed in [58]. We anticipate that as the burstiness in the traffic increases the ability of the system to adapt will be tested to a greater and greater extent. This is because burstiness in the demand function creates congestion [54] which degrades network performance. In order to describe the approach taken we have divided this chapter up into a number of sections, the structure is described below.

In Section 6.2 we introduce the concept of burstiness in data traffic, justify why wavelet models have been used, and indicate the main parameters in the burstiness model. In Section 6.3 we describe further supporting detail on the Haar wavelet which we use for generating synthetic traffic. In Section 6.4 we describe the method we use for executing the experiments on burstiness, whose results are reported in Section 6.5. Section 6.5 details methodically the effect of the pheromone persistence rate, Hurst parameter, and the demand multiplier (as defined in

Section 6.2) on the path length, degree distribution, the KL distance, number of ants dropped, and also the diameter of the graph. In Section 6.6, we bring together all of the experimental observations. Finally, in Section 6.7, we propose some further work following on from the material presented here to generate further insight.

Firstly, we justify the approach taken with respect to the warm up of the system.

6.1.1 A note on the warm up and initialisation period of the ant peer-to-peer system

The reasons for choosing the simulation lengths that we have during these experiments are stated in Section 5.1.1. We also observed that during the warm-up phase of the system, the system is far more vulnerable to disruption caused by changes in the environment or the effect of nodes entering and leaving the peer-to-peer system. For this reason the experimental results that are documented here are the result of measurements of the system taken without the warm-up phase. By examining the varying capacity of the system to cope with burstiness without the benefit of the warm-up phase we are testing the ability of the system to cope with the changes in the environment.

6.2 The modelling of burstiness

In order to consider bursty demand we must first consider what demand is. We define demand here as the rate at which new ants are released into the system, or in other words, the level of traffic that the system is demanded to cope with. So, constant demand would mean that every iteration a constant number of ants are released into the system. Bursts in demand indicate that there may be large fluctuations in the demand from iteration to iteration within the simulation run.

For this series of experiments on bursty demand it is necessary to introduce the concept of the *demand multiplier*, which helps describe the number of ants released into the system in proportion to the number of nodes within the system. This is best illustrated by example. If the total number of nodes in our system is 200, and the demand multiplier is 0.6 then every iteration 120 ants will be released in the system. Mathematically, we can define the demand multiplier as the ratio of the number of ants released each iteration to the total number of nodes in the system.

$$Demand\ Multiplier = \frac{Number\ ants\ released\ each\ iteration}{Total\ number\ of\ nodes\ in\ the\ system}$$

The source and destination nodes of each of the ants released at every iteration will be randomly chosen from the nodes in the system with the constraint that the source nodes must be capable of sending out ants. A further constraint is that the source and destination nodes must be distinct. An obvious constraint is that nodes which, due to the initial configuration of the experiments, are not able to send out ants cannot act as sources for the ants when released in the system.

In the previous equilibrium experiments the demand multiplier was constant and was equal to the proportion of nodes which were able to send out ants. The approach described here, allows us to separate out the proportion of the total number of nodes in the system able to send ants from the calculation of the number of ants sent out in a given iteration of our simulation. Burstiness in demand has been modelled as a time series which describes a series of demand multiplier values at successive iterations in our discrete time simulation. Every iteration the simulator reads in successive values for the demand multiplier from the series. In order for the demand multiplier mechanism to work we need a way of generating time series with different burstiness characteristics. In order to do this we need an easily parameterisable model of network traffic, where the parameters reflect clearly the burstiness characteristics of the time series used to represent the network traffic. There are two main ways of generating traffic time series — using wavelet approaches and Markov chains to formulate traffic models. We explore the advantages and disadvantages of each approach in the sections below and provide a justification for the choice of the wavelet approach.

6.2.1 Assessment of wavelet models

We present here the assessment for using a wavelet model in order to generate our demand time series. Firstly, there have been a number of studies [75] that show that Internet traffic exhibits strong dependence where bursts in data come in clusters. When examined in detail it is shown that the real traffic data exhibits scale invariance where the correlations in the burst structures of the traffic are exhibited over a number scales [74, 58]. The main study that showed this is [36]. The burst phenomena found in real traffic can be most clearly shown by Long Range Dependency (LRD). LRD, as shown in [72, 53], is temporal correlation in the demand function placed upon the network system, and as stated in [53], is a consequence of all purely statistically self-similar processes. Wavelets have been shown to exhibit self-similarity, and those based on the Haar Wavelet have been shown to be easily fitted to real traffic data [53]. The Haar wavelet (as defined below in Section 6.3.4) can be easily parametrised by one parameter, the Hurst exponent, H, which is explained in [36] and elaborated for the purpose of this study in Section 6.3.3.

The disadvantages of using wavelet models to generate synthetic traffic are mainly that wavelet models are less well understood than Markov models.

6.2.2 Assessment of Markov models

We present here why we decided Markov models of network traffic were less appropriate in the context of this study. The most compelling reason is that despite evidence that real traffic data does exhibit LRD, Markov models are not able to exhibit this [48]. Furthermore, Markov models such as those in [48] are described by parameters that are not easily related to the synthetic traffic time series [59, 48]. In real data there is too much correlation of arrivals for the assumption of Poisson independence of traffic to be valid [74]. Furthermore, there is an assumption that fluctuations in traffic levels will smooth out over a period of time [61].

The advantage of Markov Models is that their properties are well understood analytically [48].

6.2.3 Explanation of the main parameters in the traffic model

As defined in [53], and in Section 6.3.2, one way of modelling traffic data in computer systems is by using self-similar processes with stationary increments, specifically fractal Brownian motions generated using multi-fractal models. This method of modelling has the advantage that the properties of these self-similar processes can all be controlled with one parameter H, the Hurst exponent. It is shown in [53] that if the Hurst exponent is larger than 0.5 the associated fractal Brownian motion processes will exhibit long range dependence. The meaning of the Hurst exponent is described in Section 6.3.3. In Section 6.3 we relate the values of the Hurst exponent to real instances of traffic data.

6.3 Burstiness experiments

The main purpose of these experiments is to isolate the effect of burstiness on the system. A secondary objective is to investigate how effective the dynamic variation in pheromone persistence rate is in enabling the system to cope with that burstiness in demand. In this series of experiments we chose to restrict the number of control parameters we used. We fixed the range of awareness of each peer in our simulator to be 60 units, the grid size to be 100x100 units, and the number of nodes to be 200, and finally we chose three values for the average demand multiplier, 80 percent, 60 percent and 40 percent. The reason for choosing three values for the demand multiplier is explained later in this section. The significance of these starting conditions is as follows: the grid size and population size for the experiments have been chosen to be the same as those used in the equilibrium experiments, the values for range of awareness place the peers at the connection threshold (as defined in the chapter on equilibrium experiments Section 5.2) for their grid size and population size. In the equilibrium experiments it was found that there was a big change in the path length distribution across all configurations when the number of peers sending out ants dropped from 80 percent to 60 percent. The path length distribution reduced even more dramatically when the number of peers sending out ants was reduced further to 40 percent. By stipulating that the average demand multiplier in our bursty simulation experiments be 60 percent we are placing the system at its most responsive to either success or failure of ants to return to their source nodes successfully and, therefore, subsequent updates of the pheromone levels on paths through our peer-to-peer system. We observe, however, that in creating congestion in our peer-to-peer system by using burstiness in our demand function we may have moved the point of greatest sensitivity of our peer-to-peer system in configuration space. This is because (by definition), congestion will interrupt the flow of ants through the peer-to-peer system and therefore alter the frequency with which the renewal of pheromone levels in our system occurs. For this reason we explore burstiness in traffic with the average demand multiplier being 0.8 and 0.4 in addition to those experiments conducted with the average demand multiplier of 0.6.

6.3.1 A note on system initialisation and warm up

In the chapter on equilibrium behaviours we explored the effect of warming up the system in Section 5.1.1. In this chapter, as mentioned above, the focus is on exploring how the system responds to changes in demand function, specifically bursts in traffic. In the experiments documented in this chapter the ant peer-to-peer system is given no such warm up period. The justification for this is that we wish to explore the interplay between varying rates of change in the environment, and also varying abilities of the system to adapt. In order to most adequately do this, a cold start was required.

6.3.2 Configuration space for burstiness experiments

Two independent variables are used to generate the configuration space for our simulation experiments. Firstly, the pheromone persistence rate, and secondly, the Hurst exponent. The values of the Hurst exponent ranged from 0.5 to 0.9 in increments of 0.1. The values of the pheromone persistence rate ranged from 0.9 to 0.98 in increments of 0.02. The resulting configuration space for our experiments is depicted in Figure 6.1; at each intersection point on the grid a simulation experiment was performed.

We now consider both the generation process and the statistical properties of the demand multiplier time series, and the meanings of both the Hurst exponent and pheromone persistence rate.

6.3.3 The meaning of the Hurst exponent

Figures 6.2, 6.3, 6.4, 6.5, and 6.6 contain a number of time series generated with a range of Hurst exponent values ranging from 0.5 to 0.9. These were generated using a multi-fractal

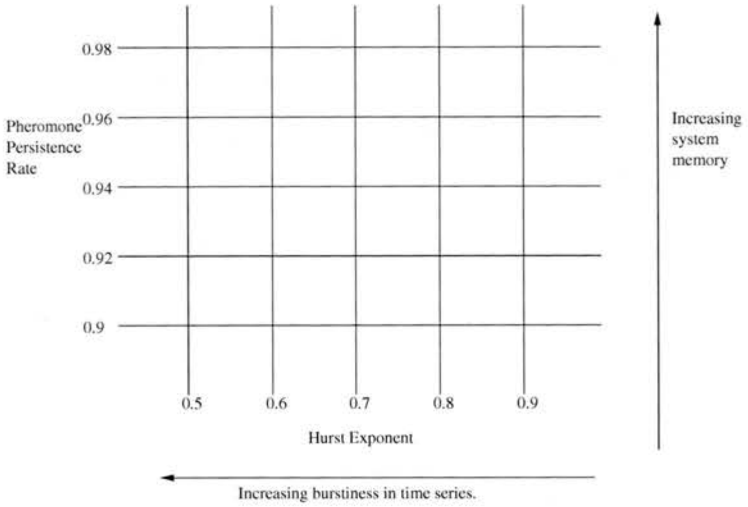


Figure 6.1: Configuration space for bursty traffic experiments

wavelet model traffic simulator, described below. The first 5000 values of each time series are plotted in the respective figures. Each of the successive values in the time series are used as values for the demand multiplier at successive iterations in the simulation runs for each of the Hurst exponent values mentioned above. In the Figures 6.2, 6.3, 6.4, 6.5, and 6.6 we have used lines to connect successive values in the time series in order to highlight variability and burstiness in the time series.

We used a tool called the Multi-fractal Wavelet Model (MWM) traffic simulator, detailed in [58], to generate all the time series. This uses a multi-fractal Wavelet Model, which in turn uses the Haar wavelet as its basis for calculation. For a description of the Haar wavelet see Section 6.3.4. A standard result for the Haar wavelet is that the variance shows a power law decay, as we move through successive scales, namely;

$$var(\psi_{j,k}) = \sigma^2 2^{(2H-1)(j-1)}(2-2^{2H-1})$$

where $\psi_{j,k}$ is the component of the wavelet at scale index j, and offset of the wavelet fragment k, $var(\psi_{j,k})$ is the variance of the time series at the jth scale, H is the Hurst exponent, and σ is the standard deviation of the wavelet at the first scale index $j = 1$. The value of the standard deviation used to generate the time series used in the simulation experiments is 0.3. We can think of the Hurst exponent as a way of describing how the variance of the wavelet changes as we move from one scale to the next. We see that as the Hurst exponent increases from 0.5 to 0.9 the changes in the time series become more dramatic and the time series becomes more abrupt and more bursty in nature. This is because when the Hurst exponent H is 0.5 the term for the variance as described above becomes independent of scale, which means there is no decay in the variance of the time series with scale. This lack of decay across scales in turn leads to

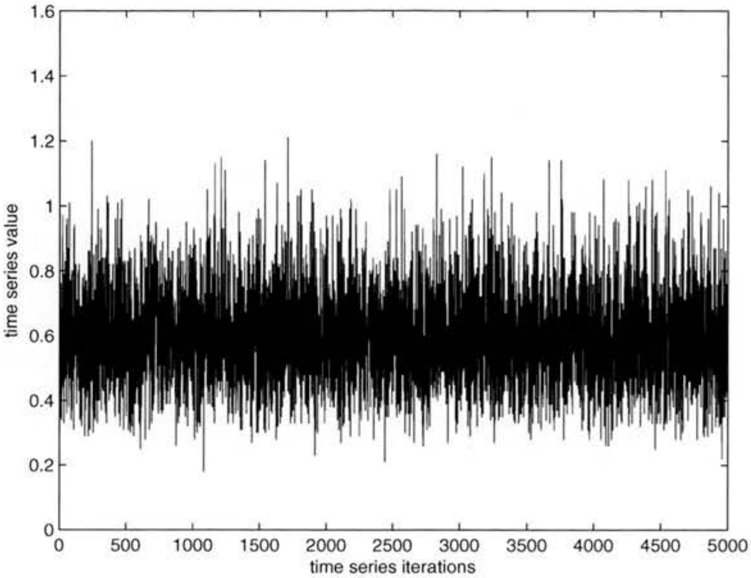


Figure 6.2: Demand multiplier time series with Hurst exponent 0.5

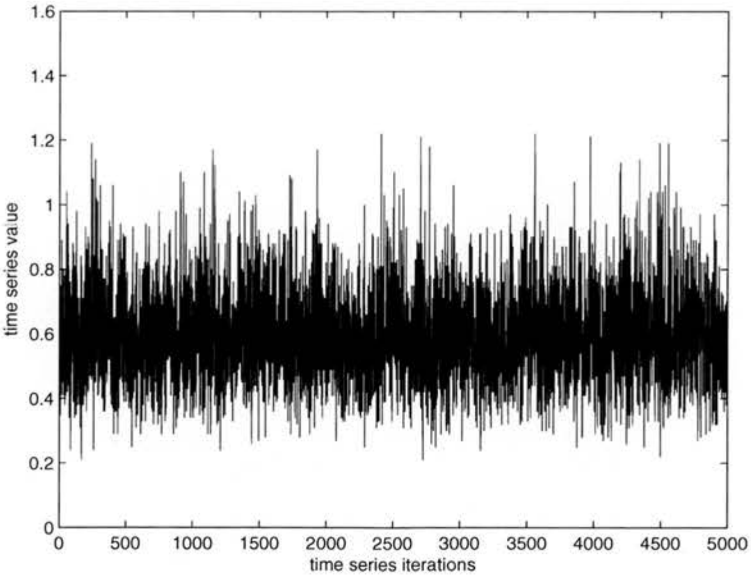


Figure 6.3: Demand multiplier time series with Hurst exponent 0.6

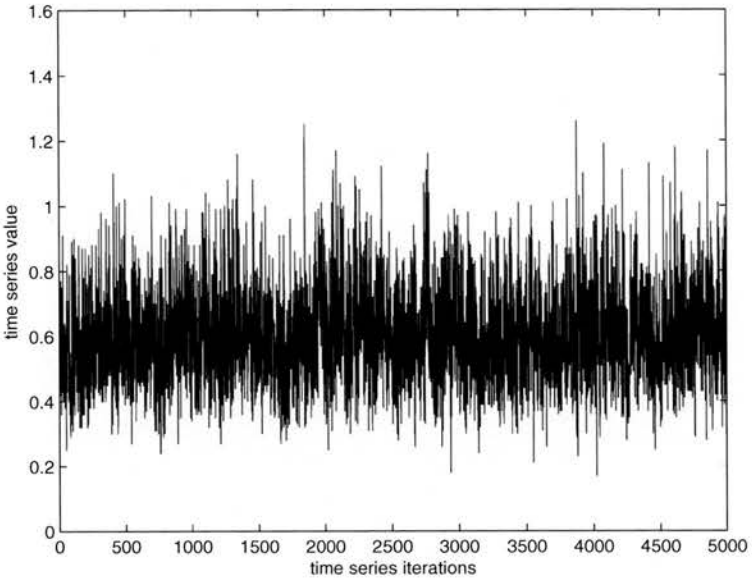


Figure 6.4: Demand multiplier time series with Hurst exponent 0.7

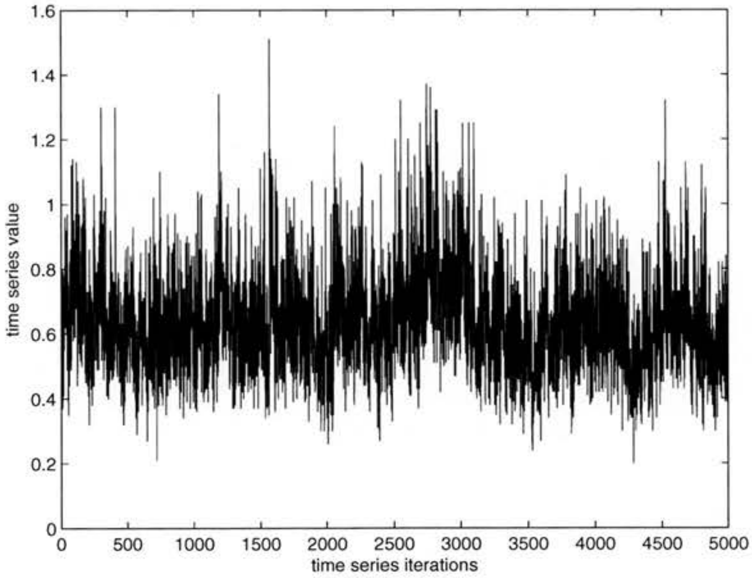


Figure 6.5: Demand multiplier time series with Hurst exponent 0.8

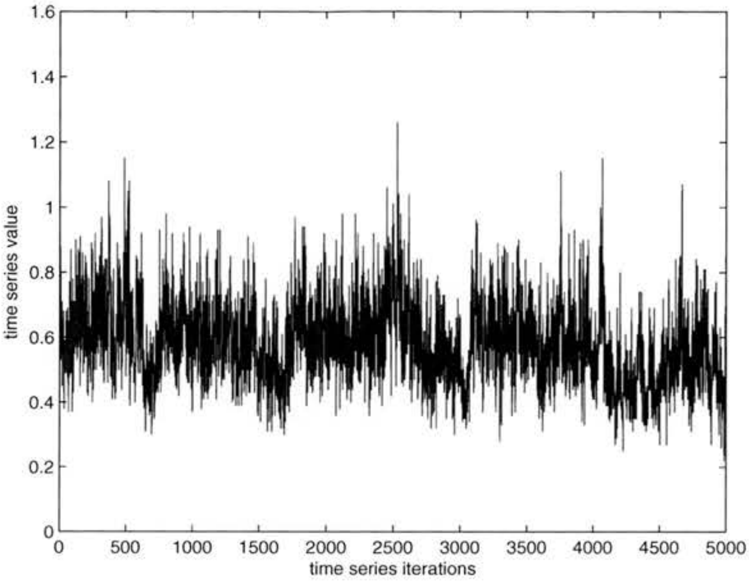


Figure 6.6: Demand multiplier time series with Hurst exponent 0.9

less abrupt bursty behaviour in the time series generated as a result. Conversely, if, due to the value of the Hurst exponent, the variance is scale-dependent the time series does exhibit abrupt changes. The Hurst exponent does not affect the mean of the time series which remains at 0.6 in these experiments. In summary, the Hurst exponent controls the abruptness in the change in the demand multiplier time series.

We must now focus on the meaning of the scale index j with respect to the number of measurements taken in the time series generated by the MWM traffic simulator. In order to do this we must consider how the calculations that generate the time series are performed in the MWM traffic simulator. A comprehensive explanation of this is given in [58]. For the purpose of explanation we provide a summary here. The MWM traffic simulator (by definition) uses a multi-fractal model to calculate the synthetic traffic time series. As explained in [58] in performing this calculation the scaling coefficients naturally form a binary tree. In moving from the top level of the tree to successive lower levels we are considering the wavelet at finer and finer scales. The wavelet in its entirety is an aggregation of all of the structures at all scales considered. The main parameter controlling this calculation is termed N_s and specifies the depth of the binary tree in calculating the wavelet. Consequently, there will be 2^{N_s} points at the lowest level of the binary tree.

From experimental measurements [57], the Hurst exponent for video signals broadcast over wide area networks has been measured to be $H = 0.84$.

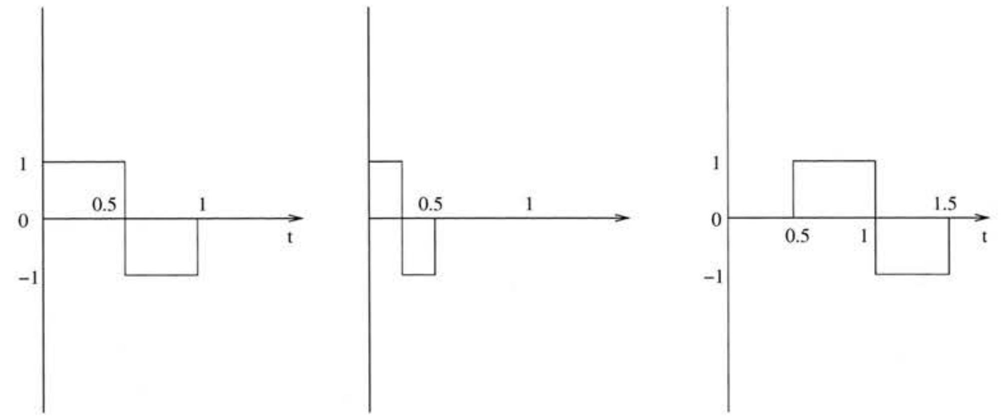


Figure 6.7: Description of the Haar wavelet and its translations and dilations

6.3.4 The Haar wavelet

In this section we state the definition of the Haar wavelet for the interested reader. The Haar wavelet, $\psi(t)$, as a function of the independent variable t, was originally described in [28] and is defined as follows:

$$\psi(t) = \begin{cases} 1 & \text{if } 0 \leq t < \frac{1}{2}, \\ -1 & \text{if } \frac{1}{2} \leq t \leq 1, \\ 0 & \text{otherwise.} \end{cases} \tag{6.1}$$

Figure 6.7 shows a graphical representation of the Haar wavelet on the left hand side. In the middle of Figure 6.7 we see the result of a dilation of the Haar wavelet, where the scale on the horizontal axis has been lengthened. On the right hand side of Figure 6.7 we see the result of a translation of the Haar wavelet where the wavelet has been shifted by half a unit to the right along the horizontal axis.

A mathematical result is that the translations and dilations of the Haar wavelet generate an orthonormal basis of functions described by

$$\psi_{j,k}(t) = \frac{1}{\sqrt{(2^j)}}\psi(\frac{t-k2^j}{2^j})$$

where j is the scale index of the wavelet which alters the vertical scale and also dilates the horizontal scale in the way depicted in Figure 6.7. Also, k is the translation index which gives an indication of the horizontal offset of the particular wavelet. By an orthonormal basis we mean, a set of functions that are all orthogonal to one another (i.e. have zero overall overlap), and also each individual function is normalised to one.

For a comprehensive description of wavelets see [42].

6.3.4.1 Fractal noises

Fractional Brownian motion, B_H [1], is defined as a statistically self-similar process with a zero mean B_H

$$B_H(0) = 0$$

and also a variance between time steps

$$E((B_H(t) - B_H(t-\Delta))^2) = \sigma^2(\Delta^{2H-1})$$

where H is the Hurst exponent, t is time, Δ is a unit change in time (or time step), and σ is standard deviation of the underlying Brownian motion process. So we can see that the greater H the wider the variance between time steps. Also we see that when the Hurst exponent, H, is 0.5 the variance of the noise process becomes independent of Δ, the unit change in time and therefore the process has no long range memory. Conversely, as the Hurst exponent H increases to a value above 0.5 the variance between the time steps does begin to depend on the unit change in time indicating some memory in the resulting noise process.

6.3.5 The meaning of the pheromone persistence rate

As described in the chapter on equilibrium behaviours (Section 5.2 on page 76 and Section 5.2.1 on page 77), the pheromone persistence rate can be interpreted as a control mechanism for determining how long each node in our peer-to-peer system remembers things for. If this memory is long (i.e. a large number of iterations), then the system may not be sensitive to perturbations in the environment that happen over short time scales. If this memory is too short (i.e. a small number of iterations), then the converse may be true and the system may be over-sensitive to short term perturbations in the environment.

6.4 Experimental setup

In this section we define the burstiness experiments which gave rise to the experimental results presented in Section 6.5, by considering the experimental procedure, experimental design and the measurements we have made during the experiments. In this way the experimental procedure is kept consistent with the experimental procedures used to explore damage and equilibrium behaviours in our ant peer-to-peer system.

Due to the stochastic nature of the ACO Algorithm, each run corresponding to a point in our configuration space had to be repeated three times in order to sample the behaviour of

[1] B_H is the notation used to describe a Brownian motion resulting from a Hurst value H

the Ant peer-to-peer system. This approach ensures consistency with the approach used when analysing the equilibrium behaviour experiments.

The burstiness experiments described here are designed specifically to investigate how the ant peer-to-peer system can cope with the change in the demand function. We do this by setting the pheromone persistence rate against the Hurst parameter as explained in Section 6.3.2.

The parameter values for the burstiness experiments are summarised in the table below.

Field name	Range of Parameter value
Percentage of nodes sending out ants	60 percent
Demand multiplier	ranges from 40 percent to 80 percent in steps of 20 percent
Pheromone persistence rate	Ranges from 1.0 to 0.9 in increments of 0.02
Range of awareness	60 units
Length of simulation run	3000 iterations
Snap-shot frequency	500 simulation iterations
Queue length of nodes	100
Pheromone threshold value	10 percent
Pheromone increment rate	3
Hurst exponent	ranges between 0.5 and 1

Table 6.1: Parameter values for burstiness simulations

6.5 Quantitative analysis of the system under conditions of burstiness

For each point in our configuration space the following measurements were taken to endeavour to understand the behaviour of the system.

- The degree distribution;
- The average degree of the graph produced;
- The average path length of the ants that returned home successfully;
- The maximum path length of the ants that returned home successfully;
- The KL distance of the degree distribution from the related Poisson distribution;
- The proportion of the total ants sent that have arrived home safely.

In all cases the initial radius of awareness of the nodes was kept at 60 units, as this is just above the connection threshold of the system (as defined in Chapter 5) given the density of the nodes on the grid. In choosing the initial radius of awareness to be at the connection threshold we have placed the system at its most sensitive. This is because any evolution of structure has to occur as a consequence of the routing decisions being made during the simulation. The system cannot rely on there being a great deal of structure at the beginning of the simulation run. This structure would be due to a large initial radius of awareness and the resultant large potential pool of neighbours. Under these conditions, any congestion caused by the burstiness in the traffic profile will have a marked effect. Ants arriving at congested nodes will be dropped from the system.

6.5.1 A note on mechanisms and methods of pheromone level modification during the simulation

As the ants move around the system, the pheromone levels in the routing tables are modified by two mechanisms. These are:

- The decay of the pheromone levels in the routing tables according to the value of the pheromone persistence rate.
- The depositing of pheromone by the ants that have successfully reached their destination and are retracing their steps.

We focus here on the second of these two mechanisms. There are a number of possible schemes by which pheromone levels in routing tables are incremented by the ants as they pass through.

In this thesis the focus has been on the exploration of the underlying dynamics of the ACO algorithm as applied to routing in peer-to-peer systems. In order to tease out the dynamics of the system a mechanism of pheromone updating that amplified the routes encoded by high pheromone levels was chosen. The version of the pheromone update rule that was chosen for these calculations is as follows:

$$\tau_{ij}(t) = (1-\rho)(\tau_{ij}(t)) + \Delta$$

In this case Δ is a constant pheromone increment. ρ is the pheromone decay rate (or the percentage of pheromone that decays in the routing entry each time-step). Finally, $\tau_{ij}(t)$ is the pheromone level in the graph at time step t between nodes i and j.

We note that the increment in the pheromone value (Δ) is constant, and does not depend on the time that has passed or the level of pheromone at the time the increment was applied. The consequence of this is that as the simulation proceeds and good routes are established the routes become less sensitive to change or perturbation. This is because the constant value Δ becomes a smaller and smaller proportion of the total amount of pheromone that has been laid down in the simulation as time goes by. Furthermore, the routing information that is laid down by the ants becomes less significant as a result. Under these conditions, the ant peer-to-peer system would take a long time to adapt to changes in the environment, as a great deal of decay in the pheromone levels in the routing tables would be needed in order for the pheromone levels deposited by ants routed as a consequence of a change in the environment to be comparable to the pheromone levels deposited by the ants routed prior to the change in the environment. This pheromone updating rule is devised in order to test the limit of the ability of the ACO algorithm to adapt to changing environments, and therefore to drive out dynamics.

An alternative update scheme is as follows:

$$\tau_{ij}(t) = (1-\rho)(\tau_{ij}(t)) + \Delta\tau_{ij}(t)$$

This scheme is kinder, as the amount of pheromone deposited by new ants is in proportion to the amount of pheromone that was there in the first place. In this case the value Δ is simply a constant of proportionality, and effectively gives a weighting to the new information (deposited by new ants) as compared to the old information (deposited previously by old ants).

We will see later (Section 7.6.3.1) that we can exploit the age of the ants returning to a node, in conjunction with a measure of confidence in the information that the ants receive in order to enhance the behaviour of ACO under conditions of variable node reliability.

6.5.2 Results for the average degree of the node in bursty conditions

In this section we document the average degree of the nodes in the ant peer-to-peer system, and investigate how this quantity varies with the pheromone persistence rate, and the Hurst

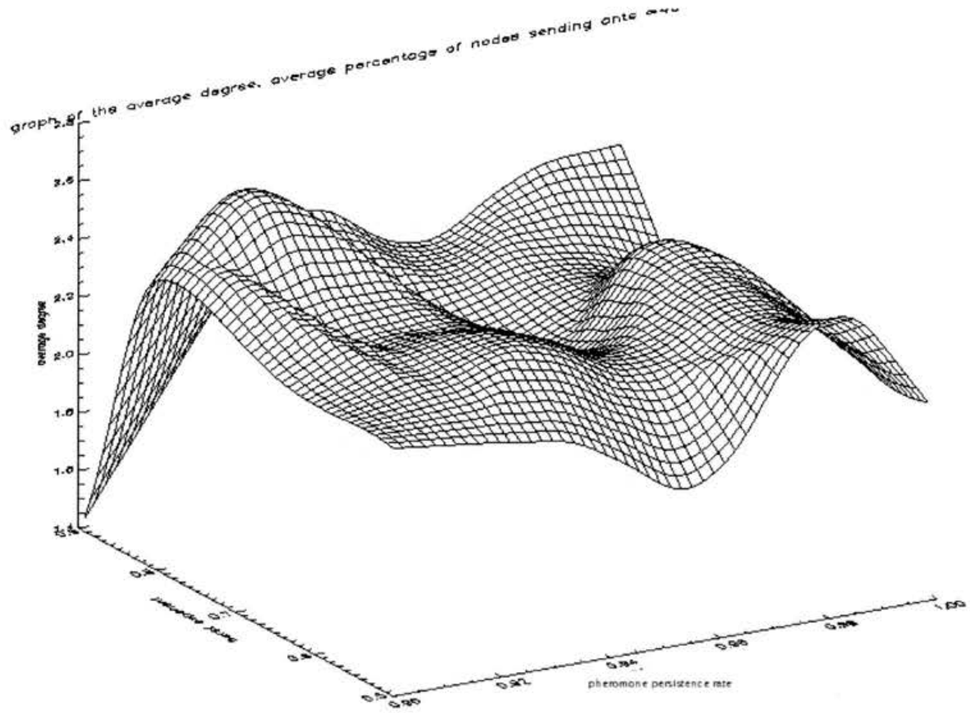


Figure 6.8: Graph of the average degree of a node for average demand multiplier of 40 percent under bursty conditions x axis pheromone persistence rate, y axis hurst exponent, z axis average degree

exponent of the demand time series.

Figures 6.8, 6.9, and 6.10 show how the average degree of the nodes in the system varies as the pheromone persistence rate varies, and also the Hurst exponent varies, as the average proportion of nodes sending out ants changes from 80 percent to 60 percent to 40 percent. We now consider the effect of the pheromone persistence rate and the Hurst exponent separately.

6.5.2.1 Effect of the pheromone persistence rate on the average degree

It is at a high values of demand multiplier (i.e. 60 and 80 percent of nodes sending out ants) and also at moderate values of pheromone persistence rates (0.94 and 0.98), and also moderate Hurst exponent values ($H = 0.5, 0.6$) that the changes in pheromone persistence rate have the greatest effect on the average degree of the system. Even then the variation in the average degree is small causing a change from 2.4 to 2.6. Outside of this region the pheromone persistence rate has little effect on the average degree.

A possible explanation for this is that under conditions of congestion (high burstiness) the movements of the ants around the peer-to-peer system are constrained. Queues in nodes in the peer-to-peer system will fill up, and nodes will begin to drop ants and because of that the rate

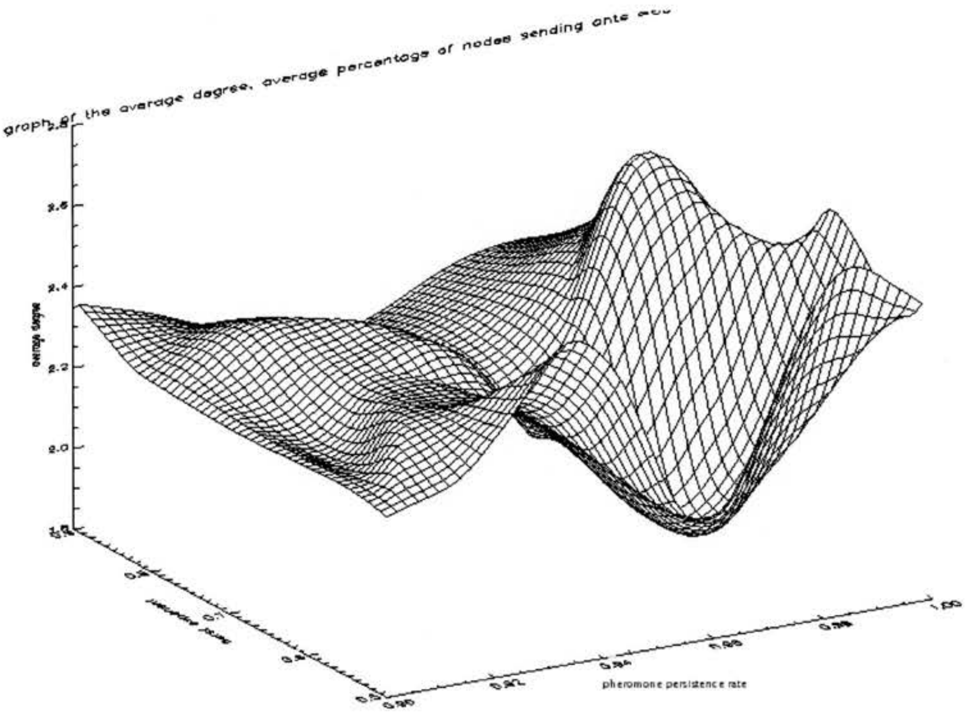


Figure 6.9: Graph of the average degree of a node for average demand multiplier of 60 percent under bursty conditions x axis pheromone persistence rate, y axis hurst exponent, z axis average degree

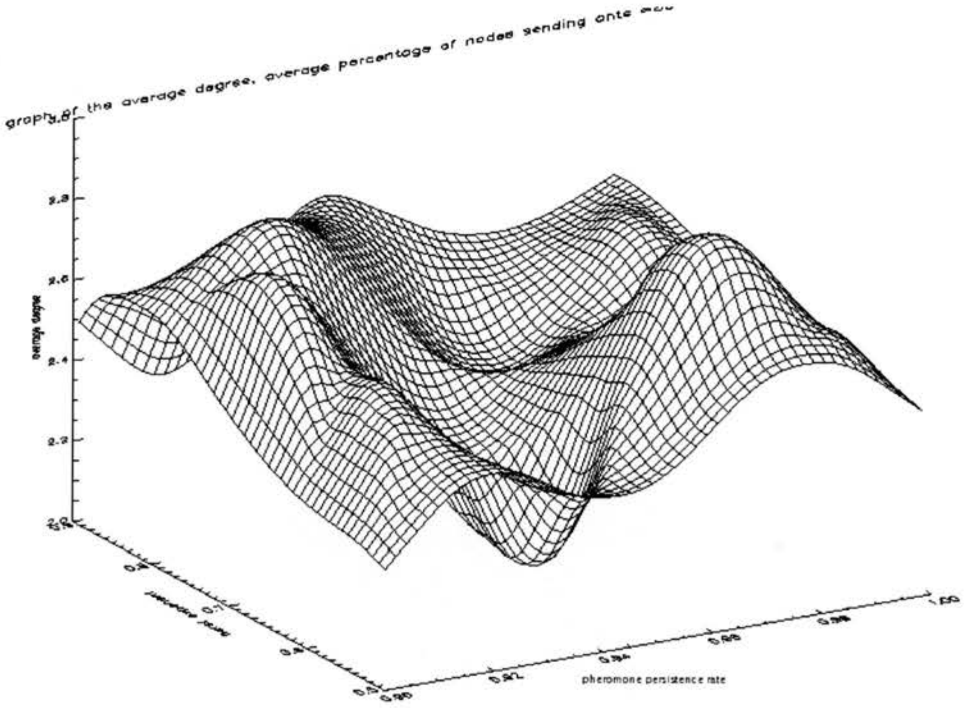


Figure 6.10: Graph of the average degree of a node for average demand multiplier of 80 percent under bursty conditions x axis pheromone persistence rate, y axis hurst exponent, z axis average degree

at which pheromone decays is irrelevant.

6.5.2.2 Effect of the Hurst exponent on the average degree

At average demand multiplier values of 40 percent the average degree has a relatively flat response to the Hurst parameter. As the average demand multiplier value increases to say 60 percent or 80 percent we see that the average degree does respond to changes in Hurst exponent. At average demand multipliers of 60 percent we see that at moderate Hurst parameter values and high pheromone persistence rate values there is a dip in the value of the average degree. We see that the average degree changes very little at other parts of the configuration space. This pattern is similar when we consider the behaviour at 80 percent of nodes sending ants. However, here we see that the dip in the average degree occurs at even more moderate pheromone persistence rates. We can consider the dip in response to the pheromone persistence rate as representing a transition in the way the system is routing traffic. At pheromone persistence rates approaching unity there is little change in the routes (as the routing information in the routing tables that is not used is not purged), and we are considering the behaviour of the routing algorithm when the routes encoded in the routing tables are static. Conversely, when the pheromone persistence rate approaches 0.9 the routes in the routing tables are dynamic, and are able to change rapidly according to circumstances. So at conditions of low burstiness, we would favour static routes, so that good routes, once established can be capitalised on. At conditions of high burstiness, we would favour dynamic routes that can adapt to the changing traffic patterns. This is reflected in the pattern of maxima in the surfaces of the average degrees of the nodes for average demand multipliers of 60 and 80 percent.

6.5.2.3 Effect of the changes in the average demand multiplier on the average degree

From Figures 6.8, 6.9, and 6.10, we see that as we decrease the average demand multiplier in the peer-to-peer system simulations from 80 percent to 40 percent the average degree of the nodes reduces slightly from 2.5 to 2.35. This effect is small, but significant, and can be explained by observing that the amount of pheromone laid down in the system reduces as the demand multiplier and hence the number of ants being released into the system reduces. This means that the decay processes that run each iteration are more able to erase any structure that is laid down in the network by the ants, which in turn has a suppressing effect on the average degree of the nodes in the network.

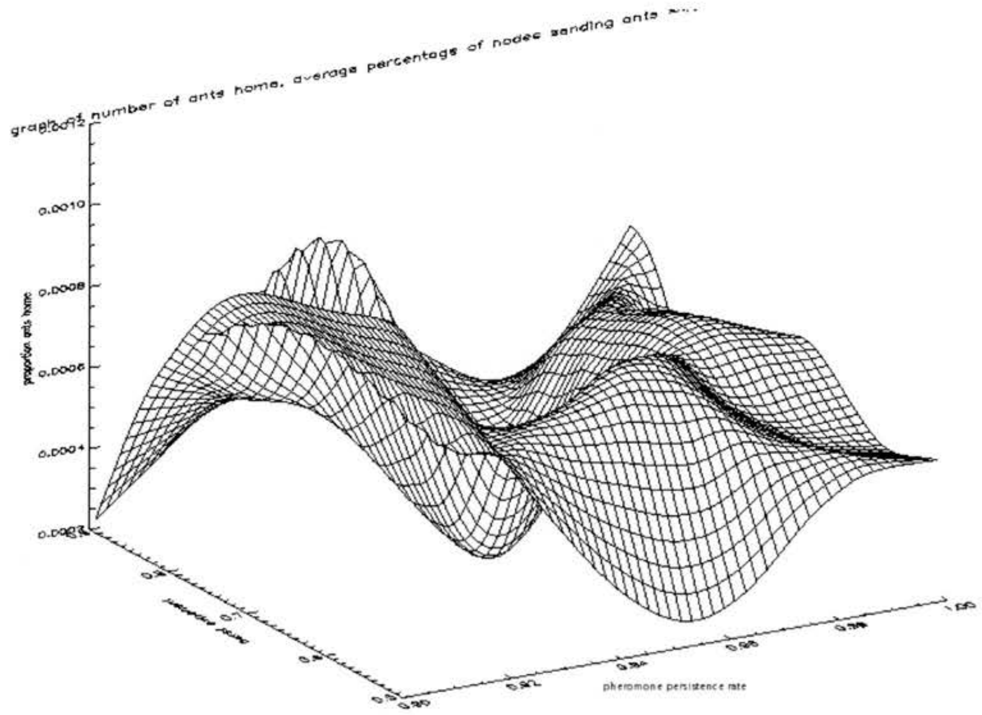


Figure 6.11: Graph of the proportion of ants getting home against Hurst exponent and pheromone persistence rate for average demand multiplier of 40 percent under bursty conditions x axis pheromone persistence rate, y axis hurst exponent, z axis proportion ants home

6.5.3 Results for the proportion of ants successfully reaching home in bursty conditions

In this section we document the proportion of ants successfully reaching home in the ant peer-to-peer system, and investigate how this quantity varies with the pheromone persistence rate, and the Hurst exponent of the demand time series.

Figures 6.11, 6.12, 6.13 show how the proportion of ants getting home successfully varies as the pheromone persistence rate and the Hurst exponent vary, as the average demand multiplier changes from 80 percent to 60 percent to 40 percent. We now consider the effect of the pheromone persistence rate, Hurst exponent, and also the demand multiplier separately.

6.5.3.1 Effect of the pheromone persistence rate

For average demand multiplier values of 80 and 60 percent, there is a slight increase in the proportion of ants reaching home as the pheromone persistence rate approaches 1 (i.e. no decay). This is because under these conditions the routing information in the routing tables persists, and the nodes can make effective routing decisions. We observe that for demand multiplier values of 80 and 60 percent, if there is no pheromone decay (i.e. pheromone persistence rate is

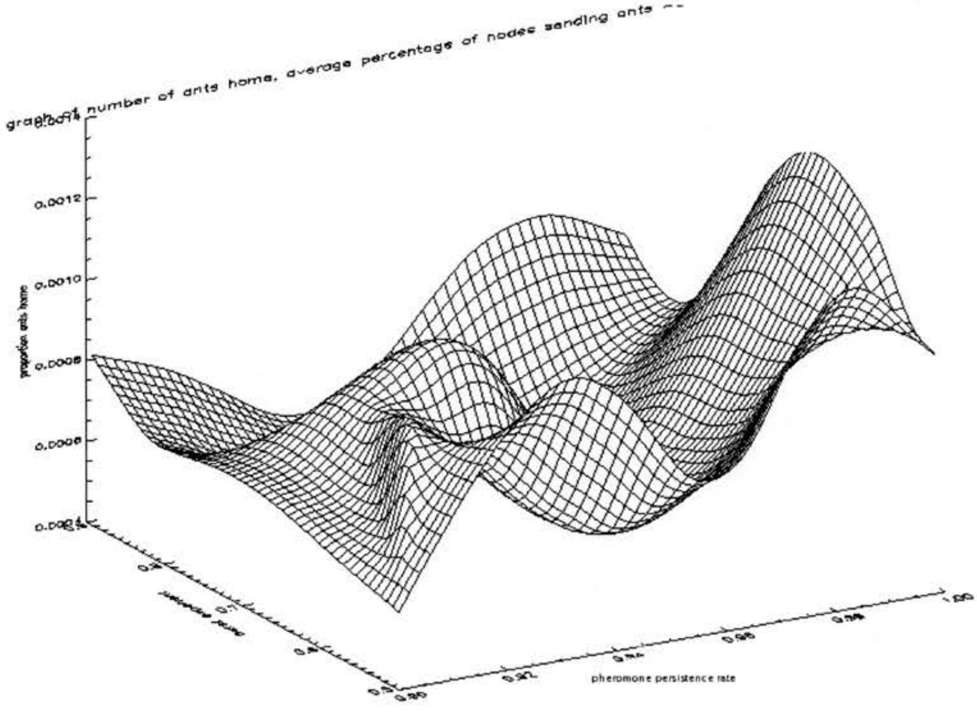


Figure 6.12: Graph of the proportion of ants getting home against Hurst exponent and pheromone persistence rate for average demand multiplier of 60 percent under bursty conditions x axis pheromone persistence rate, y axis hurst exponent, z axis proportion ants home

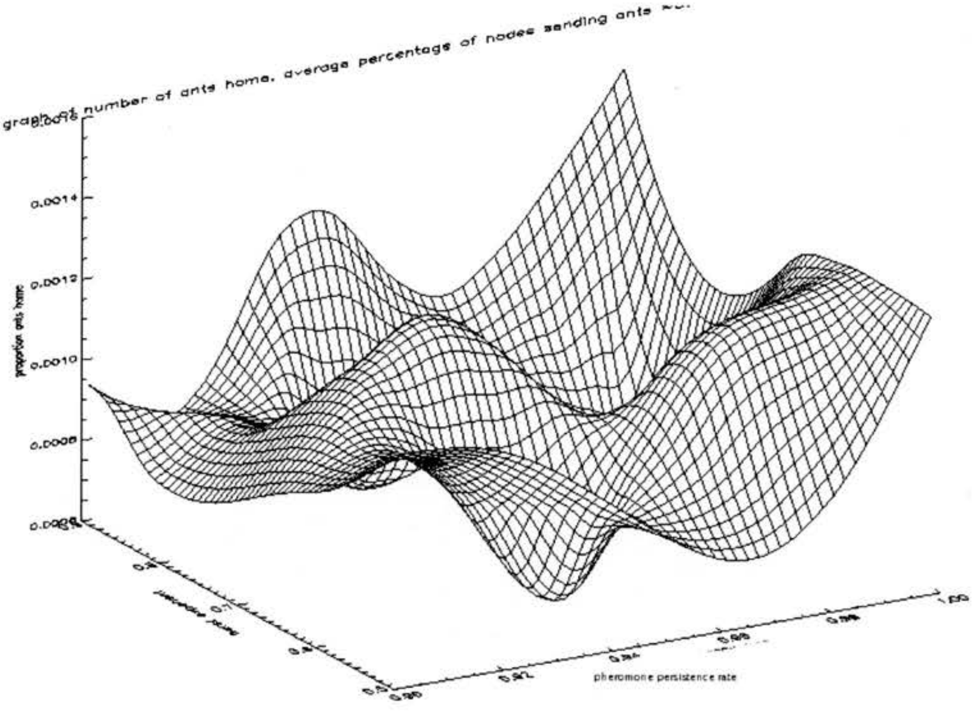


Figure 6.13: Graph of the proportion of ants getting home against Hurst exponent and pheromone persistence rate for average demand multiplier of 80 percent under bursty conditions x axis pheromone persistence rate, y axis hurst exponent, z axis proportion ants home

1) the proportion of ants reaching home is greater than elsewhere on the configuration space. We also observe, that, for 80 and 60 percent of nodes sending out ants, another line of maximum proportion of ants reaching home can be drawn from the top left hand corner (i.e. high Hurst exponent, and pheromone persistence rate - 0.9) to the bottom right hand corner (i.e. low Hurst exponent, and pheromone persistence rate = 1 or no pheromone decay). This is because the increased burstiness in the system (i.e. high Hurst exponent) causes congestion which can be alleviated by allowing a greater amount of pheromone decay (i.e. pheromone persistence rate approaching 0.9). It is under these conditions that the system is most adaptable.

6.5.3.2 Effect of the Hurst exponent

In general, at low pheromone persistence rate (i.e. pheromone persistence rate approaches 0.9) the proportion of ants getting home is invariant to changes in the Hurst parameter. However, as the system becomes more rigid, (i.e. pheromone persistence rate approaches 1) the system behaves differently at different values of the Hurst exponent. As the Hurst exponent increases from 0.5 (little burstiness) to 0.9 (a great deal of burstiness), the requisite pheromone persistence rate to create a maximum in the proportion of ants reaching home decreases. This is because as the burstiness increases a more flexible (less rigid) system is required in order to accommodate this.

6.5.3.3 Effect of the average demand multiplier

In general, as the average demand multiplier decreases from 80 percent to 40 percent the average proportion of ants reaching home decreases. One explanation is that less pheromone is being laid down as the average demand multiplier decreases. This means that there is less differentiation in the routing table entries, and therefore the routing is less directed and a smaller proportion of ants are routed successfully to their destinations.

6.5.3.4 General comment about the proportion of ants reaching home

We see that in all cases the proportion of ants reaching home is very low. An explanation for this is that the routing mechanisms within the ant peer-to-peer system are not working. There are two possible reasons for this. Firstly, congestion simply prevents ants from getting to the destination nodes, and as a result many are dropped. Secondly, the amounts of pheromone being deposited in the entries of the routing tables of the nodes by the ants may be insufficient to create enough difference between the values in the entries to effectively encode learnt routing decisions. Either way, few ants successfully reach their destination. These results tend to indicate that there is a delicate balance between the number of ants passing through the ant peer-to-peer system and also the amount of pheromone deposited on the links. Fewer ants,

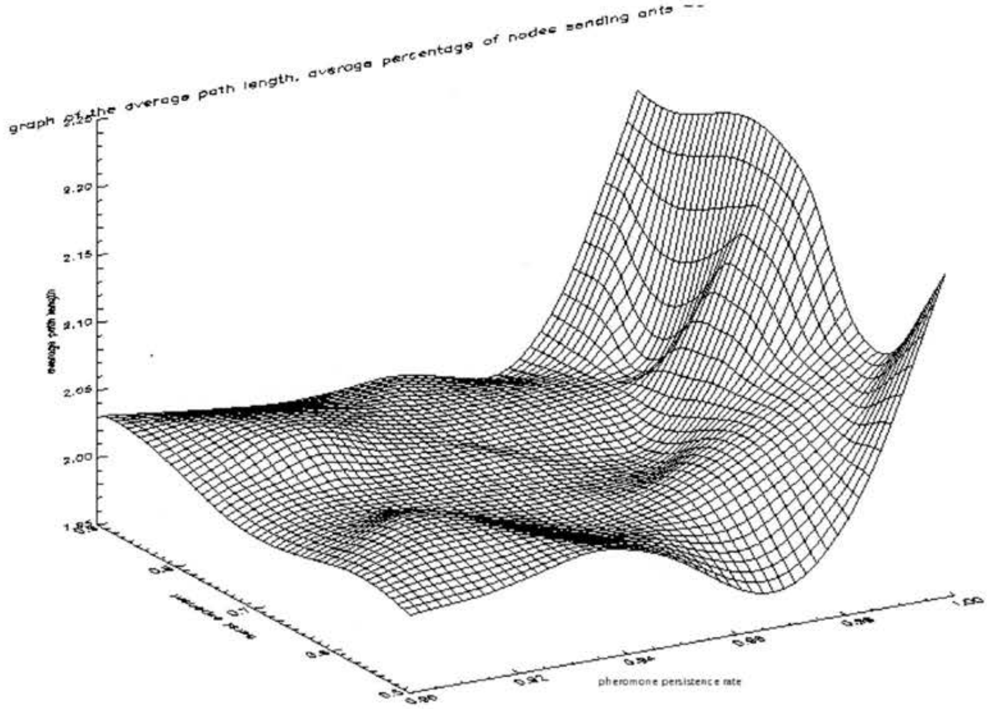


Figure 6.14: Graph of the average path length of the ants that reach home against Hurst exponent and pheromone persistence rate for average demand multiplier of 80 percent under bursty conditions x axis pheromone persistence rate, y axis hurst exponent, z axis average path length

with each ant depositing a greater amount of pheromone would have both eased any potential congestion problems, and also increased the difference in the entries of the routing tables of the nodes in the peer-to-peer system, assisting in the encoding of efficient routes within them. Both of these factors may have assisted in improving the proportion of ants reaching home in the burstiness scenarios examined.

6.5.4 Results for the average path length of the ants in bursty conditions

In this section we document the values of the average path length in the ant peer-to-peer system as we move through the configuration space detailed in Section 6.3.2, and investigate how this quantity varies with the pheromone persistence rate, and the Hurst exponent of the demand time series.

Figures 6.14, 6.15, 6.16 show how the average path length varies, as the Hurst exponent, and pheromone persistence rate varies, as the average demand multiplier changes from 80 percent to 60 percent to 40 percent respectively. We now consider the effect of the pheromone persistence rate, Hurst exponent, and also the proportion of nodes sending ants, separately.

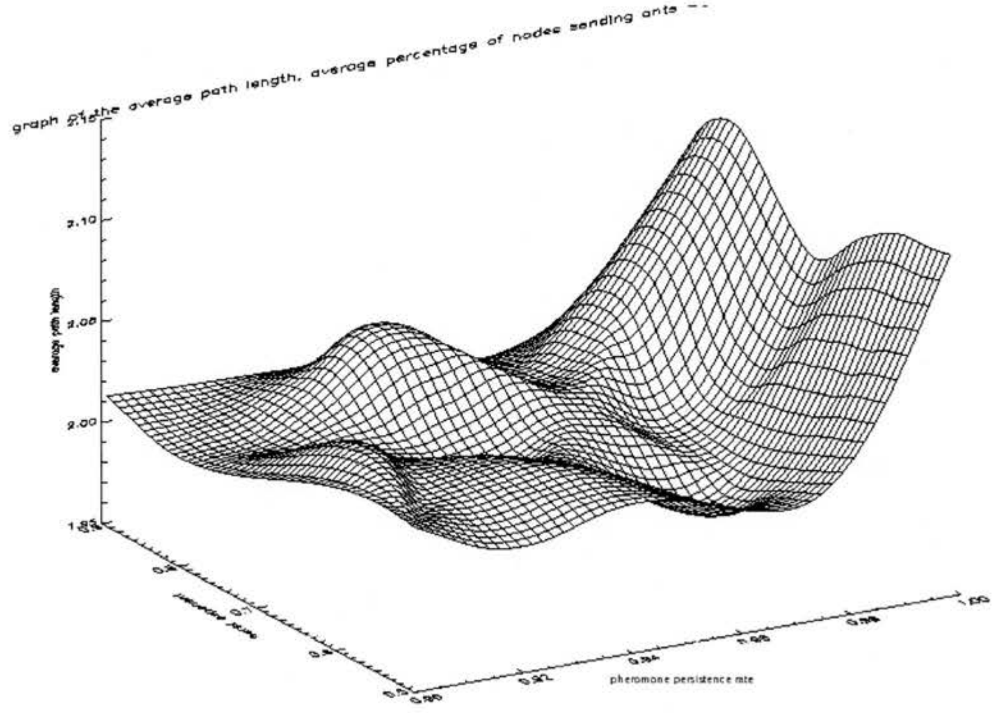


Figure 6.15: Graph of the average path length of the ants that reach home against Hurst exponent and pheromone persistence rate for average demand multiplier of 60 percent under bursty conditions x axis pheromone persistence rate, y axis hurst exponent, z axis average path length

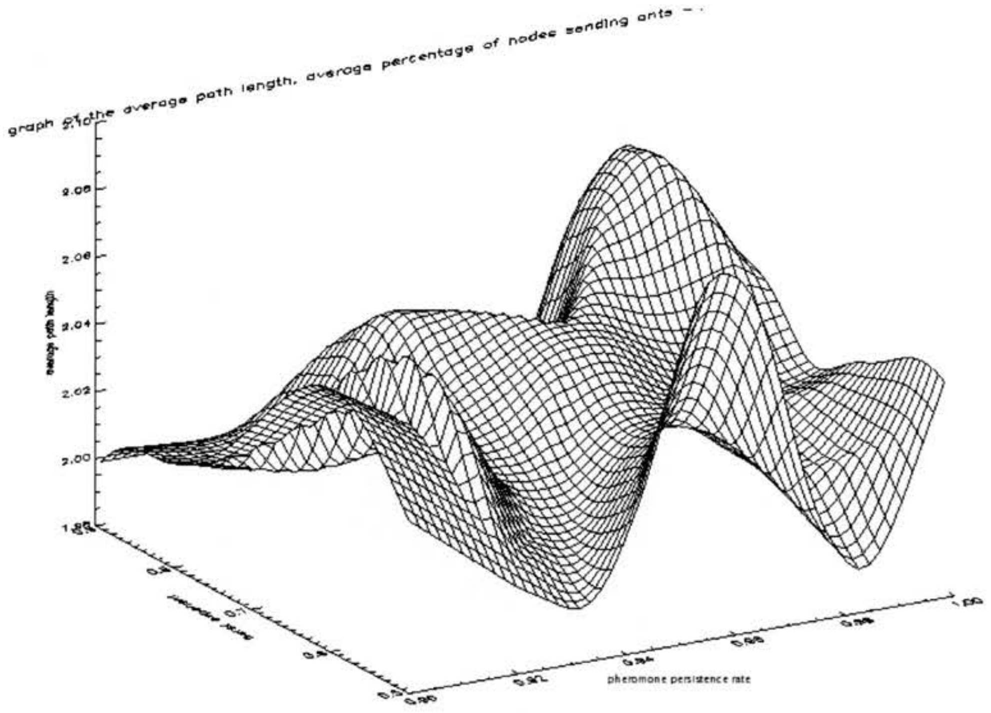


Figure 6.16: Graph of the average path length of the ants that reach home against Hurst exponent and pheromone persistence rate for average demand multiplier of 40 percent under bursty conditions x axis pheromone persistence rate, y axis hurst exponent, z axis average path length

6.5.4.1 Effect of the pheromone persistence rate on average path length under bursty conditions

In the cases of the average demand multiplier being 80 and 60 percent the response of the system to a change in the pheromone persistence rate is very flat remaining at around two hops, for pheromone persistence rates less than 0.97. Above this rate we see an upturn in the average path length as the pheromone persistence rate approaches 1. This is because under these conditions routing information does not disappear from the routing tables so longer routes can be supported by the system.

In the case of the average demand multiplier being 40 percent we see that variation of the pheromone persistence rate has a more complex effect on the average path length. The average path length remains close to 2 hops. The range of variation in the path length is smaller than for average demand multiplier value of 80 and 60 percent. We see, in general, there is an increase in the average path length as the pheromone decay approaches 1 (i.e. no pheromone decay). At low values of the Hurst exponent this gentle increase is interrupted by an intermediate peak at pheromone persistence rate $= 0.98$. In contrast, at high values of the Hurst parameter $H > 0.8$ there is a dip in the average path length at pheromone persistence rate $= 0.98$ before the increment is resumed.

A reason for the more dynamic behaviour of the system when 40 percent of nodes are sending out ants is that, because there is less pheromone being deposited, the system is more easily perturbed by the environmental conditions. So at high Hurst parameter values we need a rigid network (pheromone decay approaching 1) in order to route efficiently. At low Hurst exponent values we can explain the intermediate peak, by observing the following. If the system is too rigid (i.e. pheromone decay close to 1) then the routes cannot adapt to momentary points of congestion in the system. Similarly, if the pheromone persistence rate is too low (close to 0.9) the routing information in the node tables disappears quickly and the system is unable to route effectively.

6.5.4.2 Effect of the Hurst exponent on average path length under bursty conditions

For average demand multiplier values of 80 and 60 percent the Hurst exponent has little effect on the average path length at low pheromone persistence rates (up to pheromone persistence rate $= 0.96$). At values of pheromone persistence rate greater than this we see that the average path length increases sharply as the Hurst exponent exceeds 0.7. The reason for this is that at high values of the Hurst parameter there is congestion in the system and this blocks the short range routes in the system, and the ants are diverted to longer routes which elevates the average path length. We see the same phenomenon when the average demand multiplier is 40 percent, but to a greater extent. This is because the reduced amount of pheromone laid down by the reduced number of ants in the system renders the system even more sensitive to external perturbations. In general, for average demand multiplier values of 40 percent, as the Hurst exponent increases so does the average path length. The exception to this is when the pheromone persistence rate is low (i.e. between 0.9 to 0.94), and the Hurst parameter exceeds 0.75, we see a drop in the average path length. This can be explained by the combination of effects of a very bursty time series (high Hurst exponent) causing a great deal of congestion, a very flexible system (i.e. pheromone persistence rate close to 0.9), and also a low proportion of nodes sending out ants meaning that less pheromone is being laid down in the network. It is under these conditions that we would expect the renewal of the structure in the network to be weakest and therefore eroded most effectively. This is broadly supported by the value of the KL distance, D_{KL}, as detailed in Section 6.5.6, specifically in Figures 6.20, 6.21, 6.22, as we see that the KL distance rises as the percentage of nodes sending out ants increases and the pheromone persistence rate, and Hurst exponent value increases.

6.5.4.3 Effect of changes in the demand multiplier on average path length under bursty conditions

In general, as the average demand multiplier decreases to 40 percent the average path length becomes more responsive to the changes in initial conditions of the simulation. This is because less pheromone is being laid down, and therefore the system as a whole is more susceptible to blockages caused by congestion, which interrupt the flow of ants through the network and prevent the reinforcement of pheromone along routes therein.

6.5.5 Results for the diameter of the graph in bursty conditions

In this section we document the diameter of the graph in the ant peer-to-peer system, and investigate how this quantity varies with the pheromone persistence rate, and the Hurst exponent of the demand time series. We define the diameter of the graph as being the maximum path length traversed by the ants that successfully reached home in the ant system.

6.5.5.1 Effect of the pheromone persistence rate on diameter under bursty conditions

The general trend is that as the pheromone persistence rate approaches 1 the diameter of the graph increases. This is because as the pheromone persistence rate approaches 1 the amount of information in the routing tables increases enabling long routes to be traced by the ants as they tour the network. The effect is enhanced at high values of the Hurst parameter where there is significant congestion in the system due to burstiness removing the possibility of short routes being used.

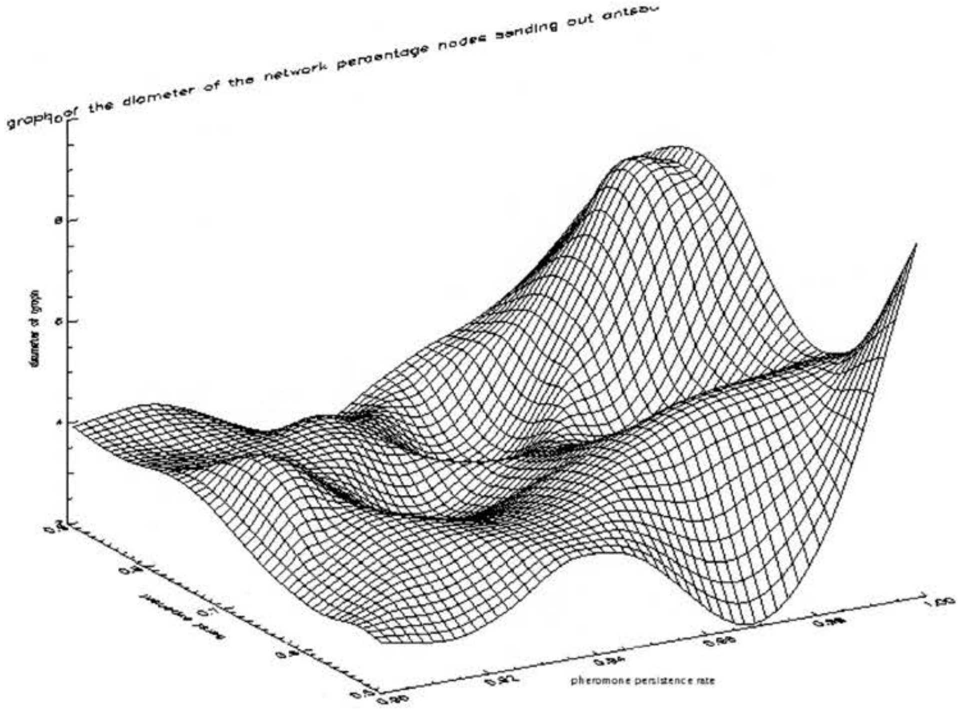


Figure 6.17: Graph of the diameter of the graph for average demand multiplier values of 80 percent under bursty conditions x axis pheromone persistence rate, y axis hurst exponent, z axis diameter

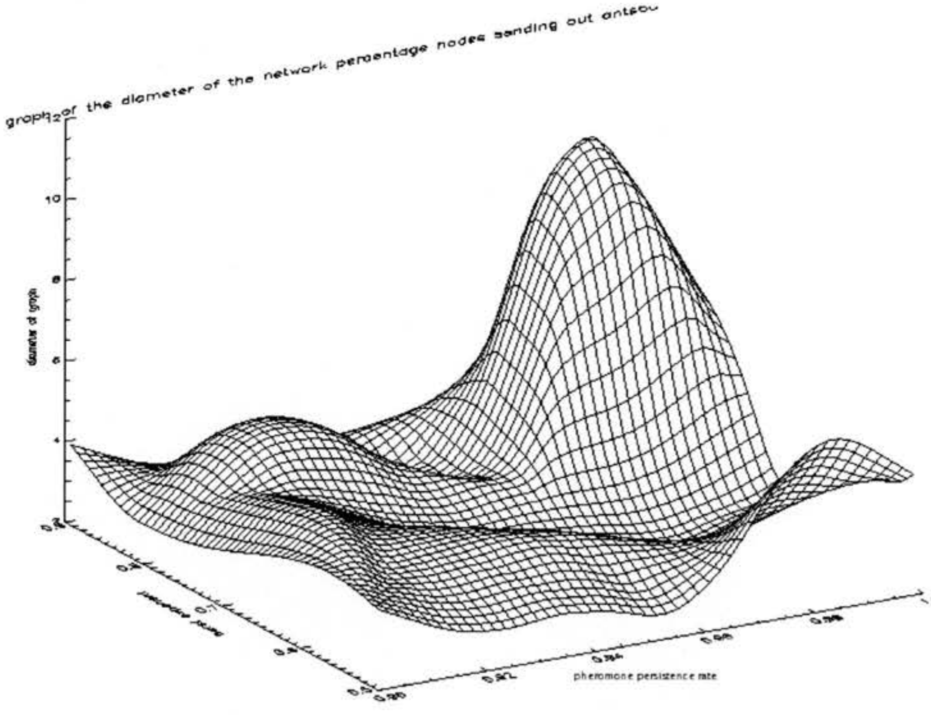


Figure 6.18: Graph of the diameter of the graph for average demand multiplier values of 60 percent under bursty conditions x axis pheromone persistence rate, y axis hurst exponent, z axis diameter

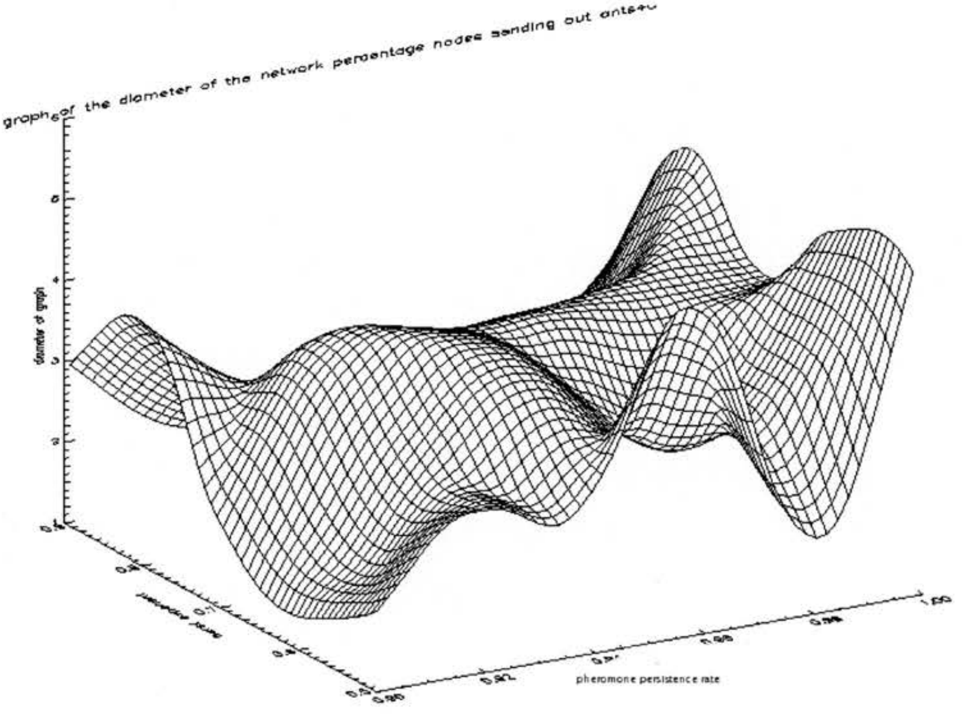


Figure 6.19: Graph of the diameter of the graph for average demand multiplier values of 40 percent under bursty conditions x axis pheromone persistence rate, y axis hurst exponent, z axis diameter

6.5.5.2 Effect of the Hurst exponent on diameter under bursty conditions

In general, as the Hurst exponent increases the diameter increases also. This is for the reason stated above, that greater congestion removes the availability of short range routes in our peer-to-peer system, although this response is flattened as the pheromone persistence rate decreases. This is because at lower pheromone persistence rate values the system is more responsive, as the entries in the routing tables can more quickly change. This means that the system can more easily route around points of congestion in the network, therefore enabling the diameter to remain small, at a range of Hurst exponent values.

6.5.5.3 Effect of the average demand multiplier on diameter under bursty conditions

As the average demand multiplier value decreases from 80 to 40 percent we see that there is a corresponding flattening of the response of diameter to changes in the Hurst parameter and pheromone persistence rate. The average diameter also decreases from around 5 to around 3. This is simply the effect of less pheromone being deposited in the routing tables and therefore less routing information being available to the ants when they arrive at the nodes.

6.5.6 Results for the KL surface

In this section we consider the variation of the Kullbeck-Leibler distance as described in the Section 3.5.2 on (page 47) of the chapter on theoretical concepts. Figures 6.20, 6.21, 6.22 show the surfaces produced as the value of the D_{KL}, as described in Chapter 3, varies with both the Hurst parameter of the time series describing the demand, and also the pheromone persistence rate. In the following subsections we consider independently the effects of the pheromone persistence rate, Hurst parameter, and also the percentage of nodes sending ants on the value D_{KL}.

6.5.6.1 The effect of the pheromone persistence rate on the KL surface

In general, as the pheromone persistence rate moves from 0.9 to 1 there is an increase in the value of D_{KL}. This can be explained by the increased amount of routing information persistent in the routing tables as the pheromone persistence rate approaches 1.

6.5.6.2 The effect of the Hurst exponent on the KL surface

We see that in general the value for D_{KL} is lower at lower values of the Hurst parameter. One possible explanation for this is that the congestion caused by bursty traffic profiles (i.e. high Hurst parameters) restricts the number of routes available for traffic to flow through. This in turn creates structure in the network. At lower values of the Hurst parameter, there is smoother

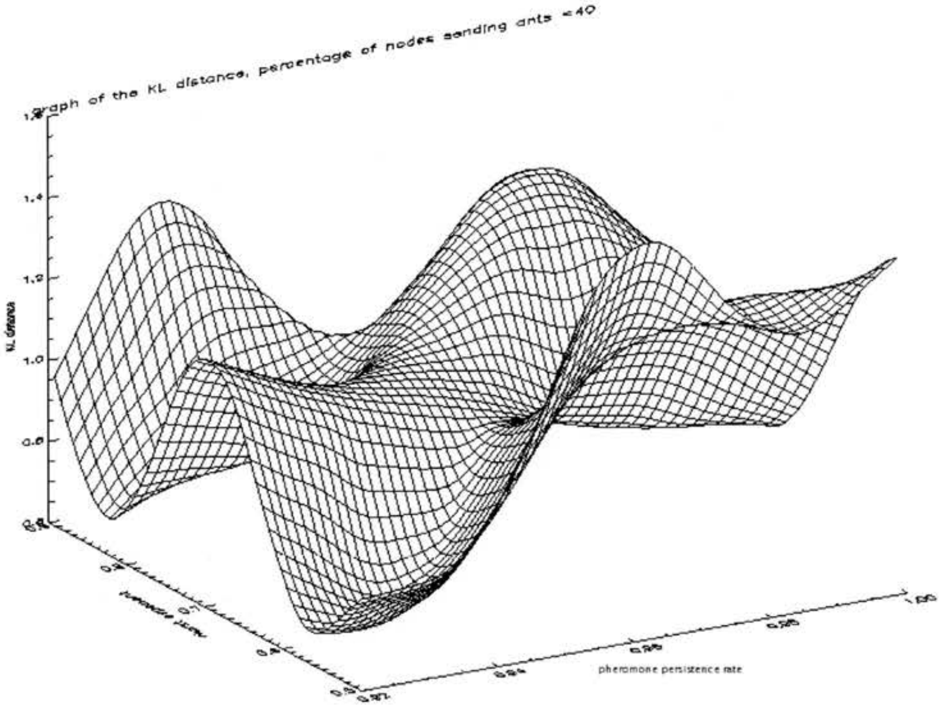


Figure 6.20: KL surface for average demand multiplier of 40 percent under bursty conditions x axis pheromone persistence rate, y axis hurst exponent, z axis KL value

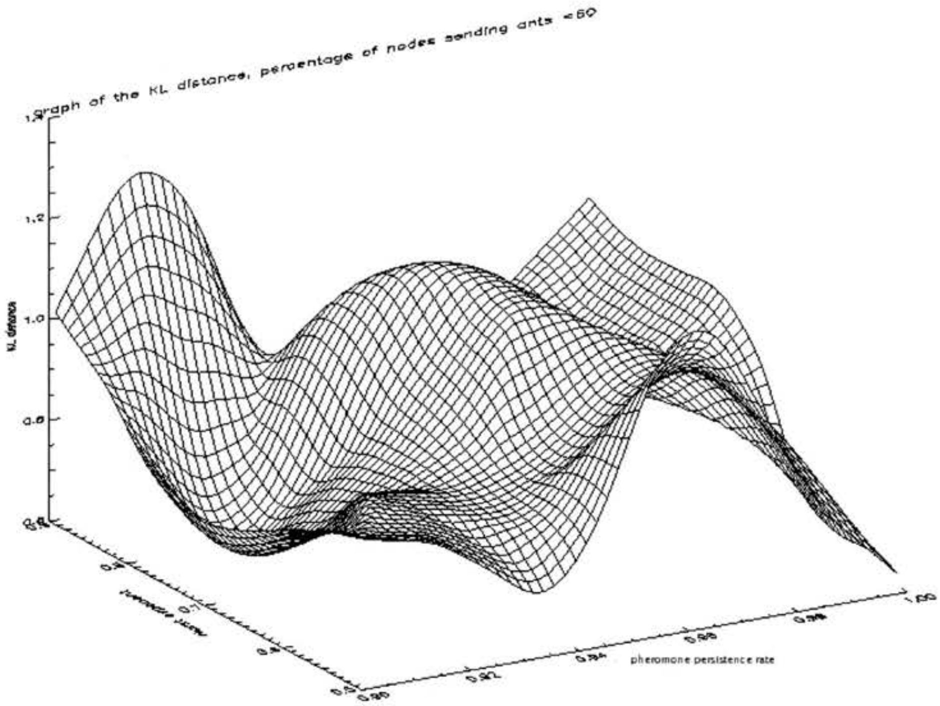


Figure 6.21: KL surface for average demand multiplier of 60 percent under bursty conditions x axis pheromone persistence rate, y axis hurst exponent, z axis KL value

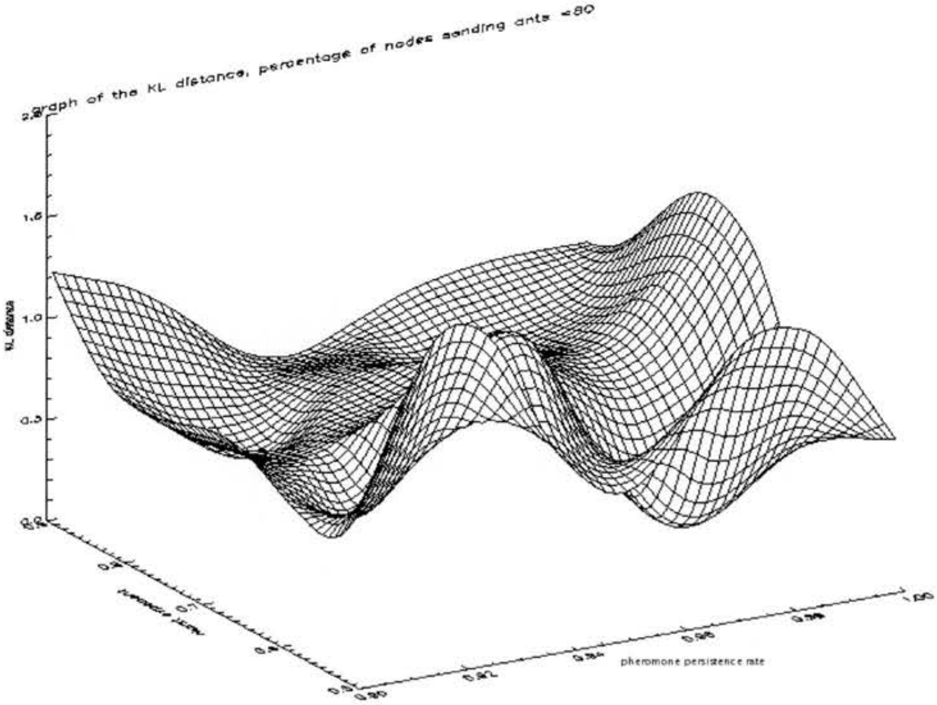


Figure 6.22: KL surface for average demand multiplier of 80 percent under bursty conditions x axis pheromone persistence rate, y axis hurst exponent, z axis KL value

flow of the ants through the network, and therefore less congestion meaning that more routes are open to discovery, and the graph has less structure.

6.5.6.3 The effect of changes in the demand multiplier on the KL surface

In general we see that as the average demand multiplier decreases from 80 percent to 40 percent there is greater variation in the KL surface, as the pheromone persistence rate and also the Hurst exponent also change. However, the average value for D_{KL} remains the same at around 1. One possible explanation could be the effect of congestion on the system, meaning that in the case of the average demand multiplier being 80 percent the value for D_{KL} is pretty constant over the configuration space. As the congestion eases, by reduction in the residual traffic level, the system becomes more responsive to characteristics in traffic level, and also variations in the Hurst exponent.

6.6 Bursty experiments conclusion

In summary, the results for the bursty experiments show that the proportion of ants getting home is very low, and the average path length of the ants that do get home is two hops. Both of these results would indicate that there is a lack of structure in the graph, and that the graph is composed of a series of disconnected components. These disconnected components are, by definition, capable of supporting only short path lengths as they encompass too few nodes to enable substantial structure in the graph to arise. Evidence to the contrary is given by the results for the diameter of the graph where we see that at pheromone persistence rates close to 1 and also high Hurst parameters diameter values exceeding 10 hops have been produced. This indicates that an ant has made ten hops between distinct nodes, source and destination, and is some evidence for long chains of nodes connected together in our network [2].

There are two possible explanations for the results presented here. These are as follows:

- The nodes are too sparsely spread on the grid and are therefore unable to form substantial connections during the simulation runs.
- Congestion in the queues of the nodes prevent ants getting to their destination and this further inhibits the creation and reinforcement of the pheromone trails in the network.

With the current data it is not possible to say which one of the two possible explanations is accurate. In the next section we discuss approaches for further optimisation of the ant peer-to-peer system.

6.7 Further work

In order to alleviate the problems stated in the previous section the following changes to the initial conditions under which the simulations have been run could be made. The suggested changes are as follows.

- Significantly reduce the number of ants being released into the system in order to prevent congestion — this can be done by reducing the average value of the demand multiplier.
- Prevent the decay of pheromone due to the decrease in the number of ants — this can be done by increasing the amount of pheromone that each ant deposits in the routing tables once it has returned home safely.
- Assist in the population of the routing tables of the nodes — this can be done by firstly increasing the initial radius of awareness of each node beyond the connection threshold, and secondly increasing the number of nodes in the system and therefore the density of nodes on the grid. Investigation of a warm up period where the node population and demand function remain constant before changes are applied to the system may also be beneficial.

[2] Although there is no indication from the data as to how many chains of this length there are in the graphs produced.

To be clear about the effect of these changes sequential execution of these changes is best, with the effect of each change then being examined by a series of experiments which interrogates the behaviour of the system under the new set of conditions generated by the change.

Chapter 7

Results for non equilibrium behaviours concentrating on damage

7.1 Introduction to the experiments and motivation

In this chapter we focus on the effect of damage in our ACO peer-to-peer system in an attempt to investigate the self-healing and self-correcting properties of our peer-to-peer system. This aspect of system behaviour is important as peer-to-peer systems experience the entering and leaving of nodes during the operation of the overall peer-to-peer system. The ability of any peer-to-peer system to cope with this dynamic aspect of system behaviour, by being able to effectively route traffic under these dynamic conditions, is key. In order to explore this aspect of peer-to-peer system behaviour we have divided this chapter into a number of sections. The overall chapter structure is indicated in the paragraph below.

7.1.1 Chapter structure

The related work for the failure and churn models has been presented earlier in this thesis in the literature survey in Section 2.5 (page 18). From this literature we see that there are a number of open questions regarding churn in peer-to-peer systems. These are reiterated in this chapter in Section 7.2. In Section 7.3 we describe a simple damage model. In Section 7.4 we describe the configuration space of that damage model, and the experiments that have been conducted to explore the response of the system to damage. We also describe the measurements taken during the experiments. In Section 7.5 we summarise the results and conclusions of these main experiments. In Section 7.6 we propose a modification to the ACO routing policy, by highlighting a shortcoming in the routing mechanism of the original ACO policy.

Firstly, we justify the approach taken with respect to the warm up of the system.

7.1.2 A note on the warm up and initialisation period of the ant peer-to-peer system

The reasons for choosing the simulation lengths that we have during these experiments are stated in Section 5.1.1. We also observed that during the warm-up phase of the system, the system is far more vulnerable to disruption caused by changes in the environment or the effect of nodes entering and leaving the peer-to-peer system. For this reason the experimental results that are documented here are the result of measurements of the system taken without the warm-up phase. By examining the varying capacity of the system to cope with node failure without the benefit of the warm-up phase we are testing the ability of the system to cope with the changes in the environment, and teasing out the underlying dynamics of the system.

7.2 Open questions regarding churn in peer-to-peer systems

From the literature on churn in peer-to-peer systems (as reviewed in Section 2.5), we saw that there are a number of open questions regarding the nature of churn such as

- what are the fundamental properties of churn?
- how similar or different are churn characteristics across different peer-to-peer platforms?
- what is the correct model for churn?
- how do we best handle churn?

Given that at the time of writing these are all open questions, in this work we have chosen a general, and easily parameterisable model for churn in peer-to-peer systems. This is described in the next section.

7.2.1 A note on system initialisation and warm up

In the chapter on equilibrium behaviours we explored the effect of warming up the system in Section 5.1.1. In this chapter, as mentioned above, the focus is on exploring how the system responds to changes in the population of nodes, specifically nodes entering and leaving. In the experiments documented in this chapter the ant peer-to-peer system is given no such warm up period. The justification for this is that we wish to explore the interplay between varying rates of change in the environment, and also varying abilities of the system to adapt. In order to most adequately do this, a cold start was required.

7.3 The modelling of damage

7.3.1 Explanation of the main parameters in the damage model

For the purposes described above, we choose to model damage in our peer-to-peer system with a two-state, discrete time Markov process, in which one state is an operational state where everything in a single node works perfectly, and the second state is our failed state where the node is in some way damaged and is non-operational. Transitions between each of these states occur probabilistically, and the state transition diagram is shown in Fig. 7.1. We are assuming that all nodes in our system are subject to the same failure model.

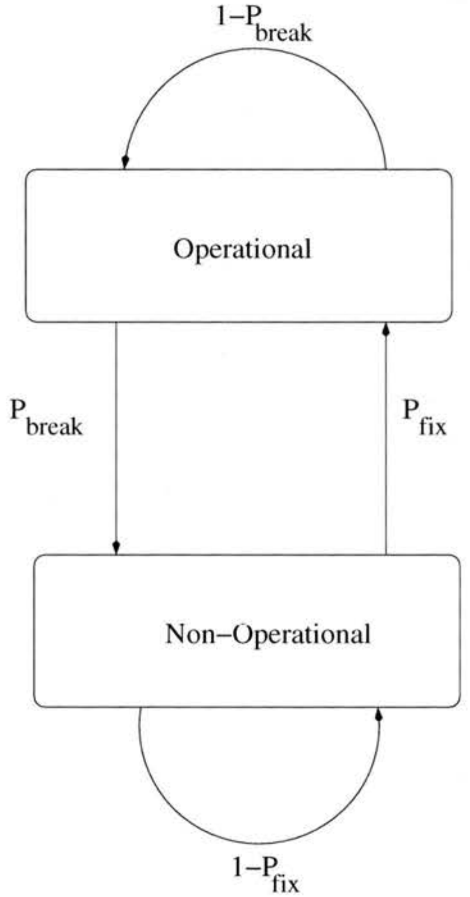


Figure 7.1: State transition diagram for damage in peer-to-peer system

From Figure 7.1 we see that there are two probabilities that parametrise the fault behaviour of the system. The first is the probability of the system breaking (making a transition from the operational state to the non-operational state), and the second is the probability of the system being repaired (making a transition from the non-operational state to the operational state). Moreover we assume that every node in the system fails independently.

By considering the Markov state transition matrix P, as defined below,

$$P = \begin{pmatrix} 1 - p_{fix} & p_{fix} \\ p_{break} & 1 - p_{break} \end{pmatrix}$$

it can be shown that the system reaches a steady state where we can specify the steady state probabilities that a node will be either operational or non-operational, ($\Pi_{operational}$ and $\Pi_{nonoperational}$) as follows

$$\Pi_{operational} = \frac{p_{fix}}{p_{fix} + p_{break}}$$
$$\Pi_{nonoperational} = \frac{p_{broken}}{p_{fix} + p_{break}}$$

The proportion of the total population of nodes that are in the operational and non-operational states can be calculated from the operational and non-operational steady state probabilities.

We can also calculate the probabilities that a node will be in the operational state n time steps after it was witnessed to be operational. This we do by calculating the Markov state transition matrix to the power n, or P^n

$$P^n = \begin{pmatrix} \frac{p_{break} + p_{fix}(1 - p_{break} - p_{fix})^n}{p_{fix} + p_{break}} & \frac{p_{fix} + p_{fix}(1 - p_{break} - p_{fix})^n}{p_{fix} + p_{break}} \\ \frac{p_{break} - p_{break}(1 - p_{break} - p_{fix})^n}{p_{fix} + p_{break}} & \frac{p_{fix} + p_{break}(1 - p_{break} - p_{fix})^n}{p_{fix} + p_{break}} \end{pmatrix}$$

So the probability of a node being in an operational state n time steps after it was witnessed to be operational is:

$$\frac{p_{break} + p_{fix}(1 - p_{break} - p_{fix})^n}{p_{fix} + p_{break}}$$

This will be of use when considering enhancements to ACO below.

In the following two sections we define the specific behaviour of the nodes of our peer-to-peer system in both the operational and non-operational states.

7.3.1.1 Explanation of the operational state

In the operational state the node is able to perform the following actions.

- The node is able to receive ants.
- The node is able to process the ants it has in its queue, by either forwarding them on to their next destination (be they backward or forward ants) or extract information from their visited lists (in the case that they have safely returned to their origin).
- The node is able to update the pheromone levels in its routing tables, both in terms of pheromone increments when the ant deposits pheromone after having successfully completed a tour, and also in terms of pheromone decay as the iterations pass by.

- The node is able to successfully remove ants from its queue once they have been successfully processed.

7.3.1.2 Explanation of the non-operational state

In the non-operational state we model the following failure behaviour.

- The node is not able to receive new ants.
- The ant that is at the beginning of the node's ant queue is deleted from the system and the information contained in that ant's visited list is discarded.
- Other ants in the node's queue remain there and are not processed until the fix event occurs and a transition to the operational state is executed.
- The pheromone levels in the routing table of the node continue to decay, because none of the routes the pheromone levels represent have been renewed.

The main effects to consider are, firstly, that the information in the routing table of the failed node will become dated as the routes are not being refreshed with current information about the environment. Secondly, the information in the ants' visited lists will also become dated as the visited lists will refer to nodes that may no longer be operational.

7.3.1.3 Justification of the node's permissible actions in the non-operational state

It may seem that there is an inconsistency in the definition of behaviour in the node's non-operational and operational states. This is because in the non-operational state the node is unable to process ants in its queue but the pheromone levels in the tables of the non-operational nodes still decay with every iteration that the node is in the non-operational state. The reason for choosing this behaviour is that the experiments documented here are intended to highlight the effect of node failure on the overall properties of the system.

We observe that if this decay of pheromone levels did not occur in the routing tables of a node that had become non-operational, then it is possible that the non-operational node may be benefiting from being in the non-operational state. This is because the pheromone levels of all the other nodes that are in operational states are subject to normal decay processes. By preventing the decay processes in the routing tables of the non-operational nodes, we would be placing the non-operational nodes at an advantage with respect to the rest of the system while they are in that state, as the routes whose pheromone levels do not decay due to the node being in the non-operational state are effectively being strengthened in comparison to those routes which do decay. This is not the intention of the damage experiments.

7.3.2 How mean time to failure, and mean time to repair relate to the probabilities of our model

The underlying assumption of this model is that each of the nodes fail independently of one another. The mean time to failure (MTTF), and mean time to repair (MTTR) are functions of the probabilities to break (P_{break}) and repair (P_{fix}) in our two state model. In this section we explore how.

The MTTF and MTTR can be thought of as the number of iterations that our system spends in the operational and non-operational states respectively. Failure occurs only when a transition from the operational state to the non-operational state is induced. Similarly, repair occurs when a transition from the non-operational state to the operational state occurs. Let us consider the MTTF, but in doing so we are minded that the same analysis holds for the MTTR.

In order to make a transition from the operational state to the non-operational state an event of probability p_{break} has to occur. The probability that this occurs at the kth iteration $p_{breakafterk}$ is

$$p_{breakafterk} = ((1 - p_{break})^{k-1})(p_{break})$$

This constitutes a geometric probability distribution. The MTTF is the mean of this geometric probability distribution, and we can work this out from a series that we sum to infinity.

$$MTTF = \sum_{k=1}^{\infty} p_{breakafterk}$$

This can be shown to be

$$MTTF = \frac{1}{p_{break}}$$

with a variance of

$$Variance = \frac{1 - p_{break}}{p_{break}^{2}}$$

By the same argument we can show the MTTR to be

$$MTTR = \frac{1}{p_{fix}}$$

$$Variance = \frac{1 - p_{fix}}{p_{fix}^{2}}$$

There are two limiting cases. Firstly, if it is certain that breakage will occur so $p_{break} = 1$ then the transition from the operational state to the non-operational state occurs on the next iteration of our discrete time simulation. Similarly, if breakage never occurs so $p_{break} = 0$

then the MTTF becomes infinite, and the system remains in the operational state and does not perform a transition to the non-operational state. Corresponding limiting cases apply to the MTTR and the probability of the node being fixed.

7.3.3 Justification for using a discrete time Markov model

As stated above the intention of this chapter is to describe the behaviour of the system under failure. The requirement for a model to describe damage in our system has been to find a model with a small number of simple parameters, whose effects are clear, and also to find a model that can easily be controlled. A further requirement was also that the model could describe the statistical behaviour of failure, and serve as the basis of a design for an implementation of such behaviour. The failure model used here has these properties, and has enabled the focus of the experiments to remain on the behaviour of the peer-to-peer system under failure. The failure model also has some underlying assumptions. Firstly, we assume that the nodes fail independently of one another. Secondly, we assume that the probability of failure and repair of the nodes is constant throughout the system and applies equally to all the nodes, once set by each point in our configuration space. Thirdly, we assume that the probabilities of failure and repair of our nodes remain constant throughout the lifetime of our simulation. In summary in our simulations, the probabilities of failure and repair are homogeneous both in time and in space.

A further advantage of the simple parametrisation of this failure model is that changes in the basic probabilities driving the model can be used to model a number of phenomena in the real world. Examples of more elaborate failure models may include, reliability growth models where the failure properties change either over time or in response to a repetition of events. Provided the assumptions stated above are not violated, the effect of these elaborations can easily be mapped to changes in the values of the two parameters in our basic model here, namely p_{fix} and p_{break}. For instance, if the reliability of the system increases for some reason then the probability of failure of a node reduces. Similarly if we wish to examine system healing behaviours we can vary the probability of repair. The basic dynamics of the system in response to these new probability measures will remain the same irrespective of the cause of the changes to those probability values.

A further justification for using a discrete time Markov chain to model failure is that we could increase the range of non-operational and operational behaviours incorporated in our model, by increasing the number of states in our Markov chain. This would be done by having a set of states which were operational, and another set which were non-operational, but each state represented a different condition of the machine (i.e. a different set of functions were permissible in each state of the machine). In this way we could model degraded operation of the machine, where the range of operational behaviours of the machine gradually narrows until a non-operational state is reached, when the machine has technically failed. This would not be possible if simple mean times to failure and mean times to repair were used to specify the damage behaviour of the system. However, mean times to failure, and mean times to repair could both be calculated from the resultant more complicated Markov chains.

Since the underlying peer-to-peer ACO system is complex it was decided to avoid unforeseen interactions as far as possible by keeping the failure model simple. The ACO algorithm implemented in this work is a stochastic distributed routing algorithm which has load balancing and self-healing properties. In this chapter we are adding the further complication of a stochastic damage model to our system. It is important to limit this complexity in order that the composite system containing our stochastic routing algorithm, and our stochastic damage model remain easily and clearly parameterisable.

7.3.4 The effect of damage on the system

When a node becomes non-operational in our peer-to-peer system this has a number of effects. The first effect is that the flow of ants (both those going forward to the destination, and backward ants returning to the source) is interrupted. The second effect is that the control mechanism for the ACO peer-to-peer system is also interrupted. This is because the backward ants that are returning to the source may not be able to retrace their steps, which means the source ceases to get feedback about the validity of the routes of the ants the source node has sent out. Finally, thirdly, there is an effect in terms of contamination of information. Even those ants that do make it back to the source nodes may be in possession of inaccurate information about the validity of the routes they have passed through. The issue of invalidity of information is explored later in this chapter.

Here, we seek to explore the response to damage by our ACO peer-to-peer system by a number of experiments. To reiterate, the version of ACO used here is the basic form where only the pheromone trails are used to regulate routing of ants in the peer-to-peer system.

7.4 Damage experiments

As stated above the main purpose of these experiments was to isolate the effect of the volatility in the environment caused by the damage and healing cycle. In this series of experiments we chose to restrict the number of control parameters we use. We fixed the range of awareness of each peer in our simulator to be 60 units, the grid size to be 100 by 100 units, and the number of nodes to be 200, and finally we chose two values for the demand multiplier, 60 and 40 percent. The pheromone persistence rate took 3 values 0.92,0.96, and 0.98.

The significance of these starting conditions is as follows; the grid size and population size for the experiments have been chosen to be the same as those used in the equilibrium experiments in order to assist comparison with the equilibrium experiments.

In the equilibrium experiments it was found that there was a big change in the path length distribution across all configurations when the number of peers sending out ants dropped from 80 percent to 60 percent. The path length distribution fell further when the number of peers sending out ants was reduced further to forty percent. By stipulating that the percentage of peers sending out ants for our damage simulation experiments is sixty and also forty percent we are placing the system at its most responsive to either success or failure of ants to return to their source nodes successfully and therefore update the pheromone path.

7.4.1 Configuration space for damage experiments

The experimental framework outlined above leaves two independent variables which can be used to generate the configuration space for our damage simulation experiments. Firstly, p_{break} the probability that the node makes a transition from the operational to the non-operational state, and secondly, p_{fix}, the probability that the node makes a transition from the non-operational to the operational state. The values of probability of node failure and probability of node repair ranged from 0.1 to 0.9 in steps of 0.2. The configuration space can be visualised using Figure 7.2.

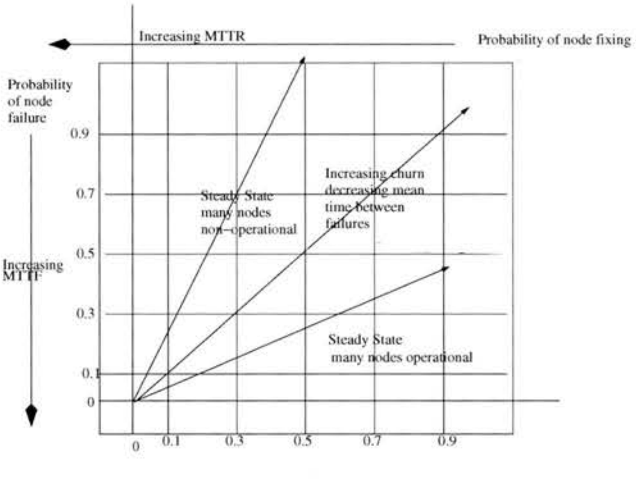

Increasing MTTR
Probability of node fixing
Probability of node failure
Increasing MTTF
0.9
0.7
0.5
0.3
0.1
0
0
0.1
0.3
0.5
0.7
0.9
Steady State many nodes non−operational
Increasing churn decreasing mean time between failures
Steady State many nodes operational

Figure 7.2: Configuration space for damage in peer to peer system

The reason for choosing the probability range to omit 0 for both p_{fix} and p_{break} is that if either probability is 0 the corresponding state (either non-operational or operational respectively) becomes absorbent and all the nodes end up in that state over a period of time. This behaviour would not support the purpose of these experiments which is to explore the effect of dynamic failure on our ACO peer-to-peer system, and the ability of the peer-to-peer system to cope with it.

7.4.1.1 Ways of considering the configuration space of our damage model

There are two observations to make about the damage model configuration space. Firstly, the larger the sum of the probabilities $p_{fix} + p_{break}$ the greater the amount of churn in the system. By considering the lines in our configuration space which are perpendicular to the line $p_{fix} = p_{break}$ we can connect points in the configuration space for which the churn in the system is the same. By considering lines that pass through the origin at a given gradient we can connect points in our configuration space for which the proportion of operational and non-operational nodes in the steady state behaviour is the same. As the gradient of the line passing through the origin rotates from horizontal to vertical the proportion of the nodes that are operational in our steady state varies from 100 percent to 0 percent. Both of these concepts are depicted in Figure 7.2. We can specify the mean time between failure (MTBF) of each node to be:

$$MTBF = MTTF + MTTR$$

The concept of churn is related to the mean time between failures as the smaller the mean time between failures, the greater the churn in the system. To reflect this one definition of churn could be

$$Churn = \frac{1}{MTBF}$$

This measure of churn would give an indication of the proportion of the total population of nodes that change state (from operation to non-operational, or vice versa) each simulation iteration.

In the context of the two-state, discrete time Markov model used here we see that this measure of churn has a simple relationship to the parameters of the Markov model.

$$Churn = \frac{p_{break} p_{fix}}{p_{break} + p_{fix}}$$

Consideration of this expression reveals that this measure of churn is at its greatest when both the probability of node breaking and node fixing are unity, and also if either the probability of node breaking or fixing are zero then the value for churn reduces to zero as well.

We can think of a given configuration in our configuration space as being a unique combination of values for churn and also proportions of operational and non-operational nodes in the steady state.

7.4.2 Experimental procedure for damage volatility experiments

Due to the stochastic nature of ACO, experiments initiated using each point in our configuration space had to be repeated three times in order to sample the behaviour of the system. Further the values of the measurements were averaged across all three for each point in the configuration space.

7.4.3 List of measurements taken during the experiments

The following measurements were taken during the experiments. The reader is referred to Section 5.3 for a more comprehensive definition.

- The average degree of the graph produced.
- The path length distribution.
- The Diameter.
- The KL distance of the degree distribution from the related Poisson distribution.
- The proportion of the total ants that make it home to the source.

7.5 Results and conclusions of experiments with ACO

In this section we detail the results for the damage experiments in our ant peer-to-peer system. We consider two sets of results: those when the demand multiplier is 60 and 40 percent respectively, for a range of pheromone persistence rate values.

7.5.1 Results for the average degree

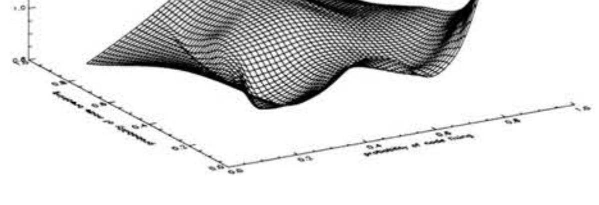


Figure 7.3: Graph of the average degree of the system for demand multiplier of 40 percent, pheromone persistence rate 0.92, x axis probability of node fixing, y axis probability of node breaking, z axis average degree

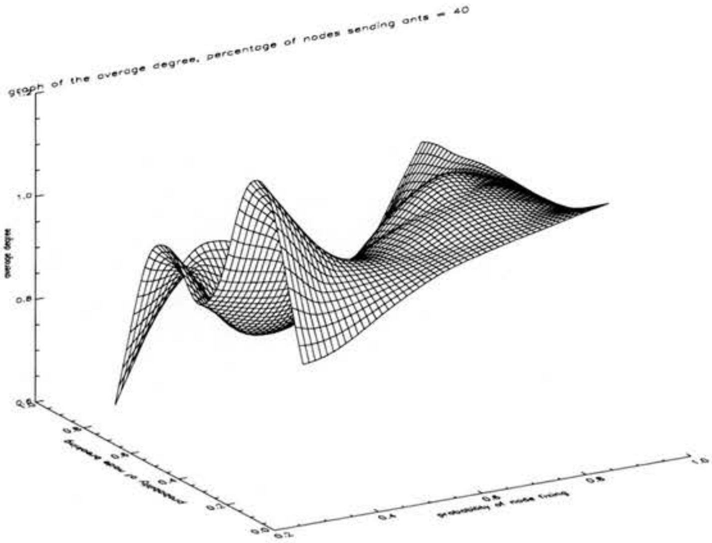


Figure 7.4: Graph of the average degree of the system for demand multiplier of 40 percent, pheromone persistence rate 0.96, x axis probability of node fixing, y axis probability of node breaking, z axis average degree

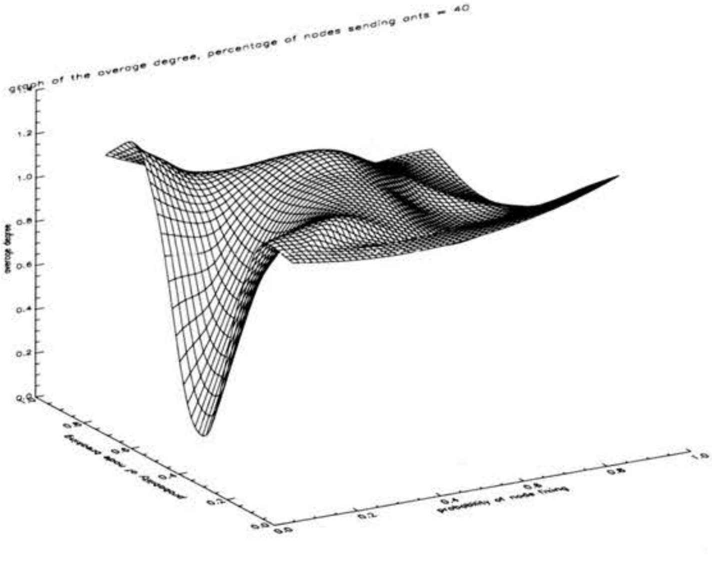


Figure 7.5: Graph of the average degree of the system for demand multiplier of 40 percent, pheromone persistence rate 0.98, x axis probability of node fixing, y axis probability of node breaking, z axis average degree

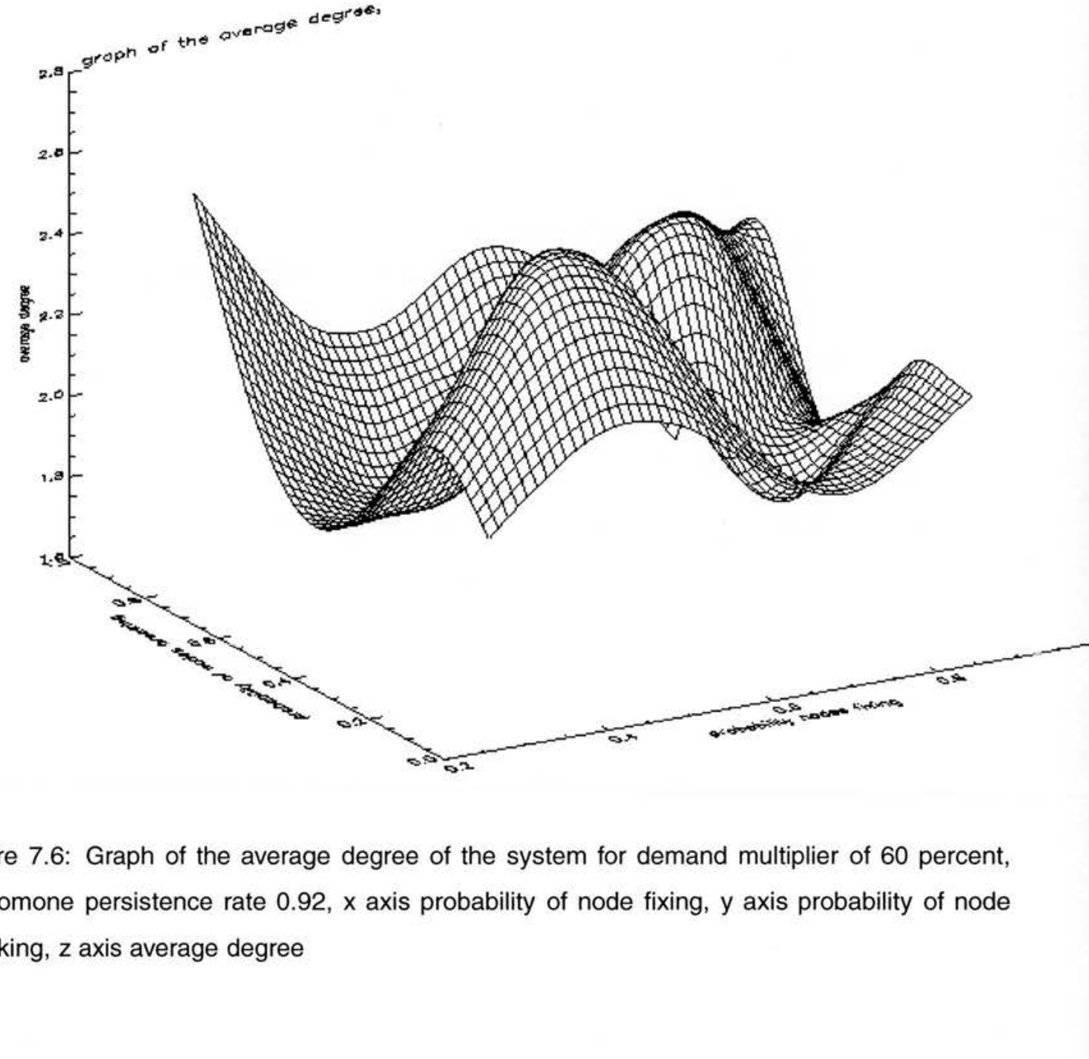


Figure 7.6: Graph of the average degree of the system for demand multiplier of 60 percent, pheromone persistence rate 0.92, x axis probability of node fixing, y axis probability of node breaking, z axis average degree

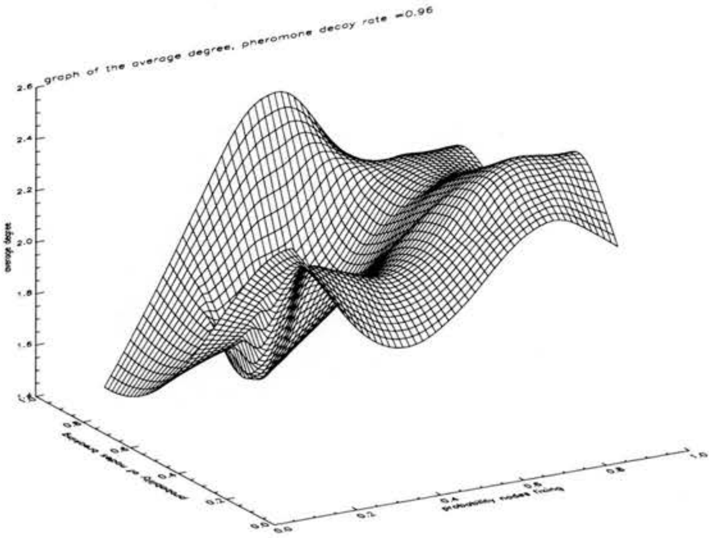


Figure 7.7: Graph of the average degree of the system for demand multiplier of 60 percent, pheromone persistence rate 0.96, x axis probability of node fixing, y axis probability of node breaking, z axis average degree

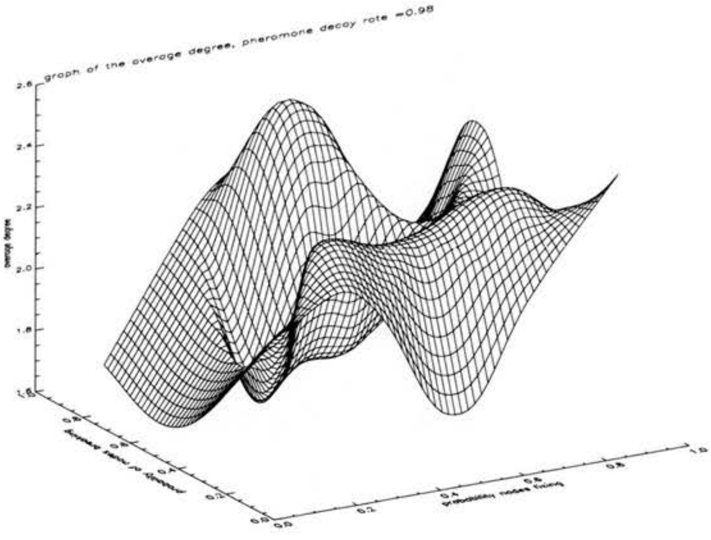


Figure 7.8: Graph of the average degree of the system for demand multiplier of 60 percent, pheromone persistence rate 0.98, x axis probability of node fixing, y axis probability of node breaking, z axis average degree

We see from Figures 7.6, and 7.7, 7.8 that for values of demand multipliers of 60 percent the average degree is about 2 for all values of probabilities of node failure and node repair. We see that as the probability of the nodes breaking increases the average degree collapses to zero. This trend is the same when the demand multiplier is forty percent (see Figures 7.3, and 7.4, 7.5). Here we see that the average degree is around unity, and only rises when the probability of node repair is high and the probability of node failure is low.

7.5.2 Results for the average path length

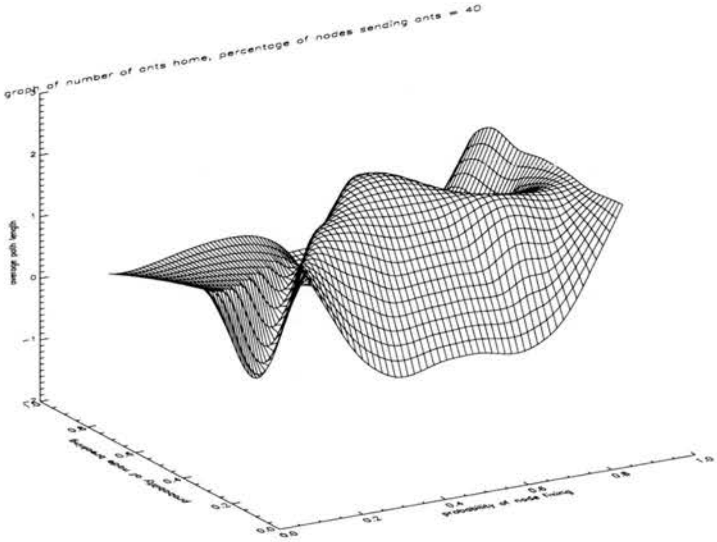


Figure 7.9: Graph of the average path length of the system for demand multiplier 40 percent, pheromone persistence rate 0.92, x axis probability of node fixing, y axis probability of node breaking, z axis average path length

We see from figures 7.9, 7.10, 7.11, 7.12, 7.13, 7.14 that, in general, as the probability of node failure increases beyond 0.2 the average path length collapses to zero. We see that as the pheromone persistence rate increases from 0.92 to 0.98 this pattern remains roughly the same.

7.5.3 Results for the proportion of ants reaching home

From the results of the proportion of ants reaching home (Figures 7.20, 7.19, 7.18, and 7.17, 7.16, 7.15) we see that, in general, the number of ants getting home is very low. We see that in all cases of pheromone values and demand multiplier values, the proportion of ants getting home becomes zero when the probability of node failure exceeds 0.4. When the probability of node repair approaches unity there is a sharp increase in the proportion of ants reaching home.

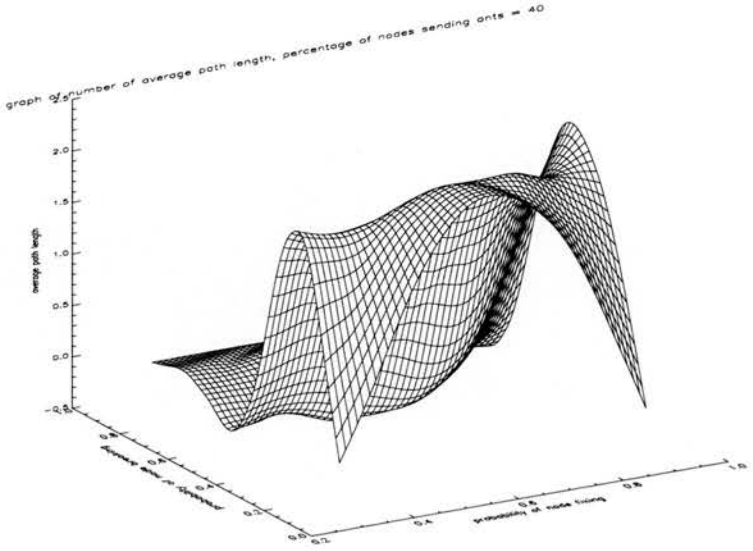


Figure 7.10: Graph of the average path length of the system for demand multiplier 40 percent, pheromone persistence rate 0.96, x axis probability of node fixing, y axis probability of node breaking, z axis average path length

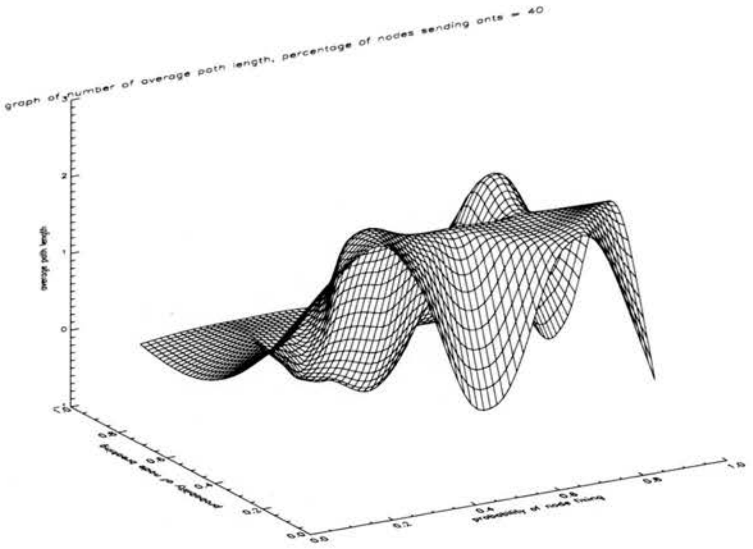


Figure 7.11: Graph of the average path length of the system for demand multiplier 40 percent, pheromone persistence rate 0.98, x axis probability of node fixing, y axis probability of node breaking, z axis average path length

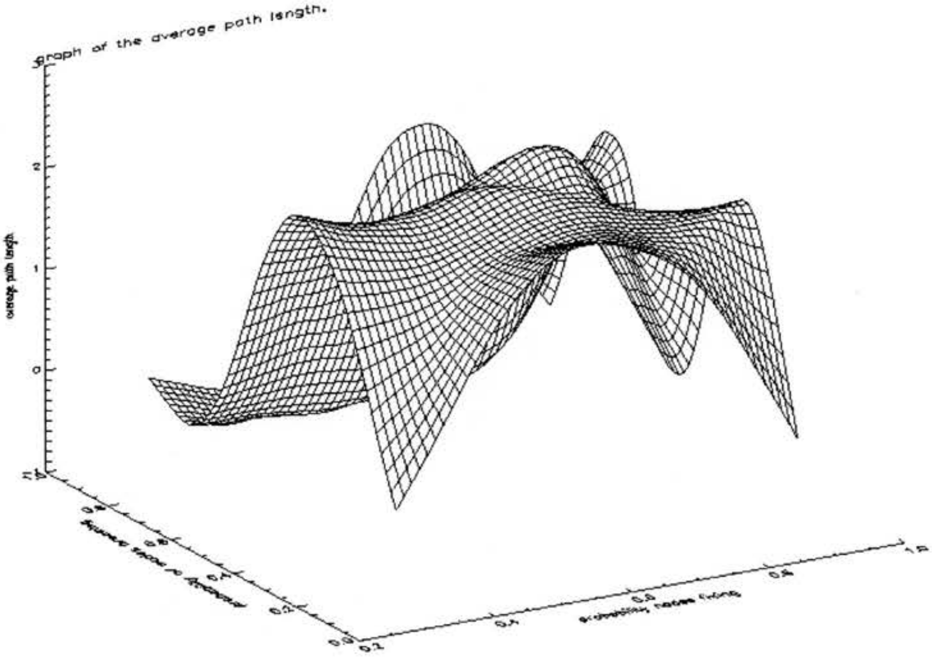


Figure 7.12: Graph of the average path length of the system for demand multiplier 60 percent, pheromone persistence rate 0.92, x axis probability of node fixing, y axis probability of node breaking, z axis average path length

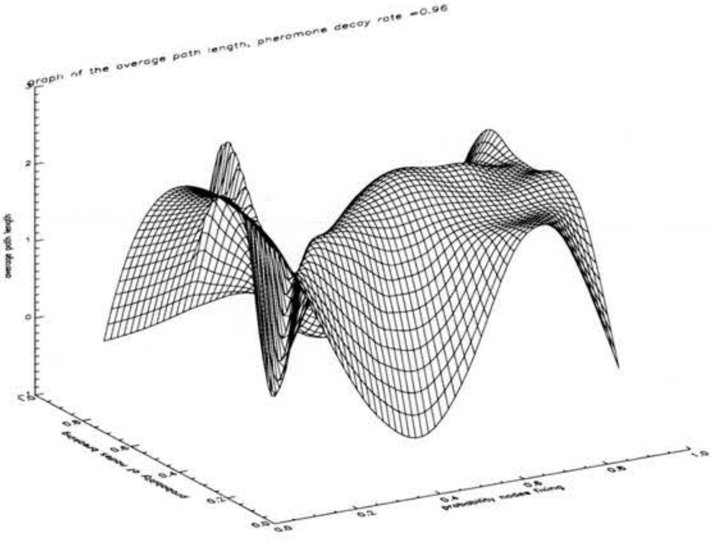


Figure 7.13: Graph of the average path length of the system for demand multiplier 60 percent, pheromone persistence rate 0.96, x axis probability of node fixing, y axis probability of node breaking, z axis average path length

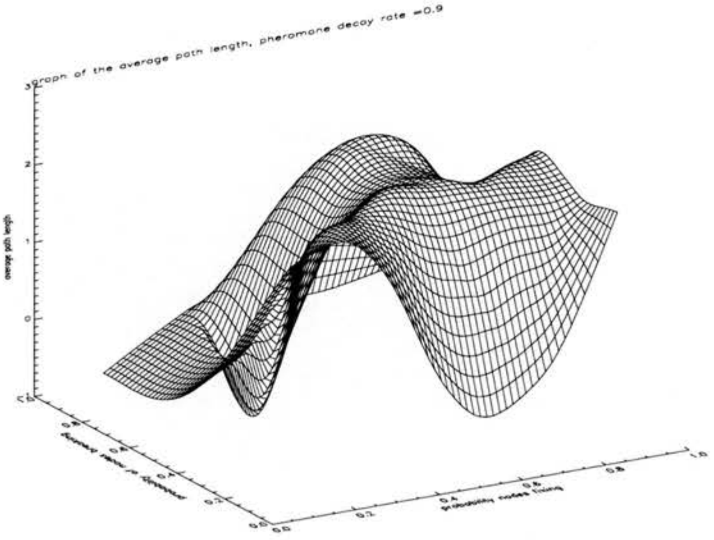


Figure 7.14: Graph of the average path length of the system for demand multiplier 60 percent, pheromone persistence rate 0.98, x axis probability of node fixing, y axis probability of node breaking, z axis average path length

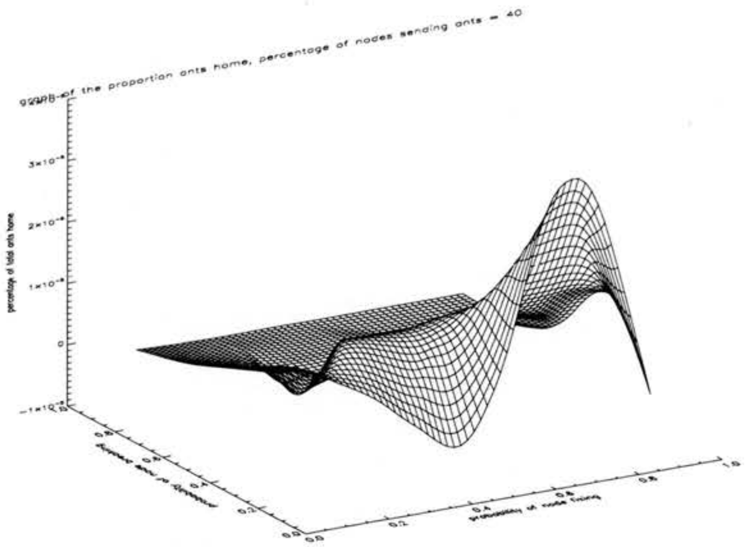


Figure 7.15: Graph of the proportion ants getting home for demand multiplier of 40 percent, pheromone persistence rate 0.92, x axis probability of node fixing, y axis probability of node breaking, z axis proportion of ants home

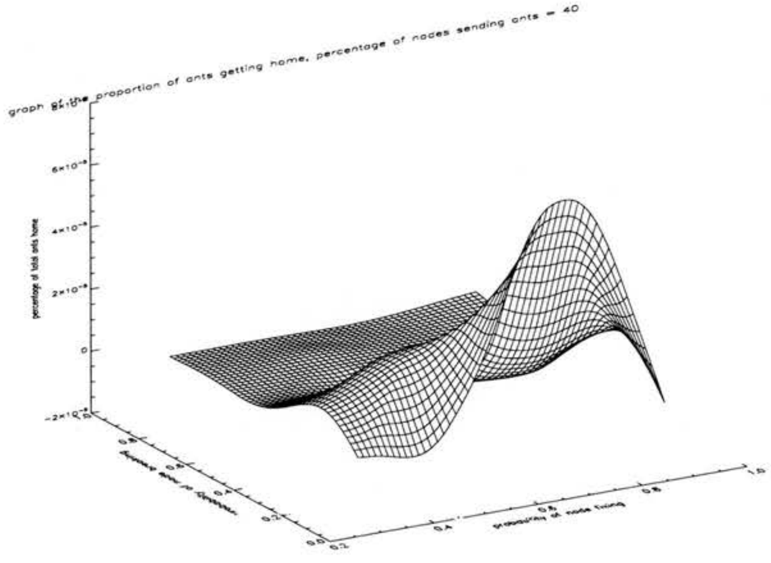


Figure 7.16: Graph of the proportion ants getting home for demand multiplier of 40 percent, pheromone persistence rate 0.96, x axis probability of node fixing, y axis probability of node breaking, z axis proportion of ants home

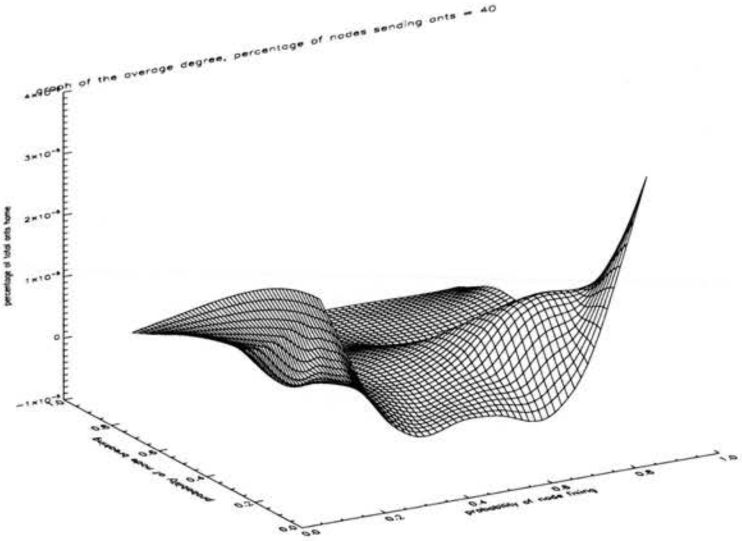


Figure 7.17: Graph of the proportion ants getting home for demand multiplier of 40 percent, pheromone persistence rate 0.98, x axis probability of node fixing, y axis probability of node breaking, z axis proportion of ants home

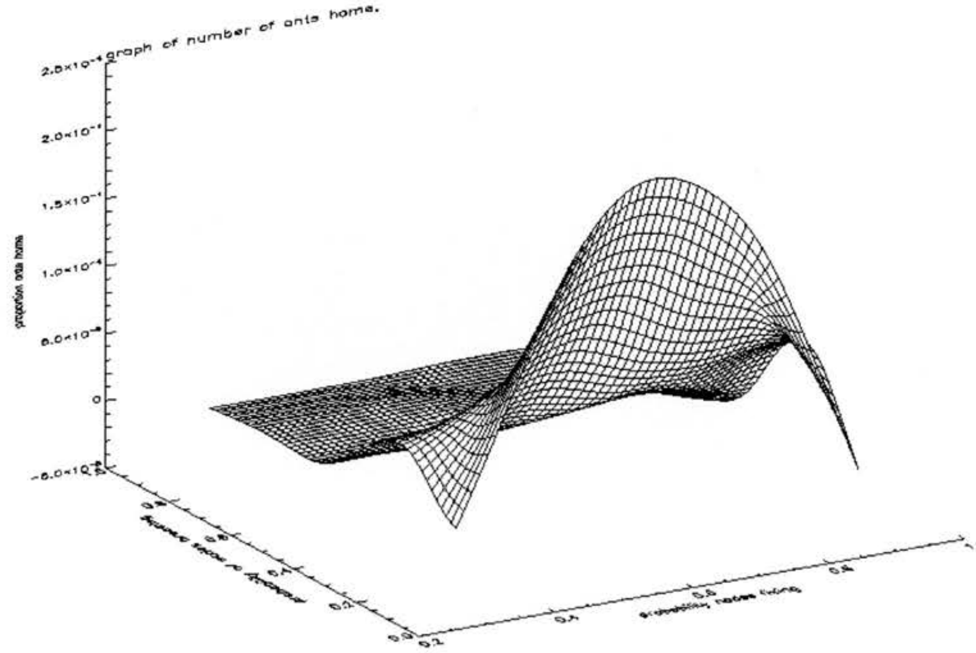


Figure 7.18: Graph of the proportion ants getting home for demand multiplier of 60 percent, pheromone persistence rate 0.92, x axis probability of node fixing, y axis probability of node breaking, z axis proportion of ants home

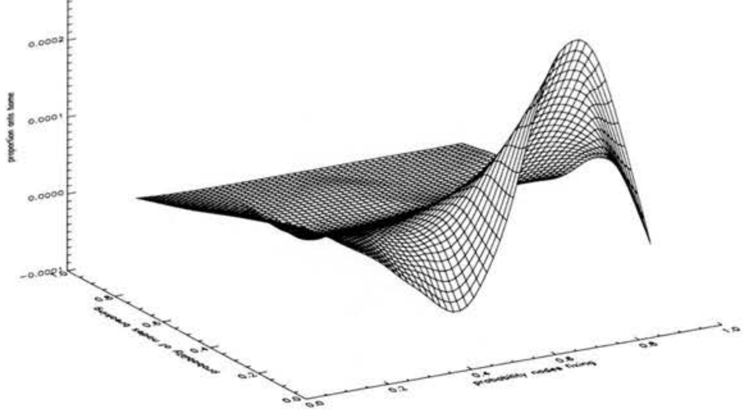


Figure 7.19: Graph of the proportion ants getting home for demand multiplier of 60 percents, pheromone persistence rate 0.96, x axis probability of node fixing, y axis probability of node breaking, z axis proportion of ants home

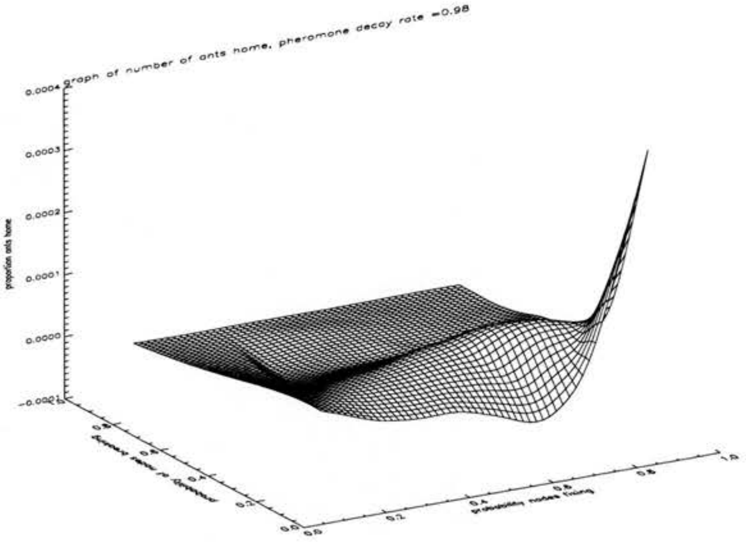


Figure 7.20: Graph of the proportion ants getting home for demand multiplier of 60 percent, pheromone persistence rate 0.98, x axis probability of node fixing, y axis probability of node breaking, z axis proportion of ants home

7.5.4 Results for the diameter of the graph

In this section we consider how the diameter (that is the maximum path length) of the graph changes with the values of the initiating parameters of the simulation runs.

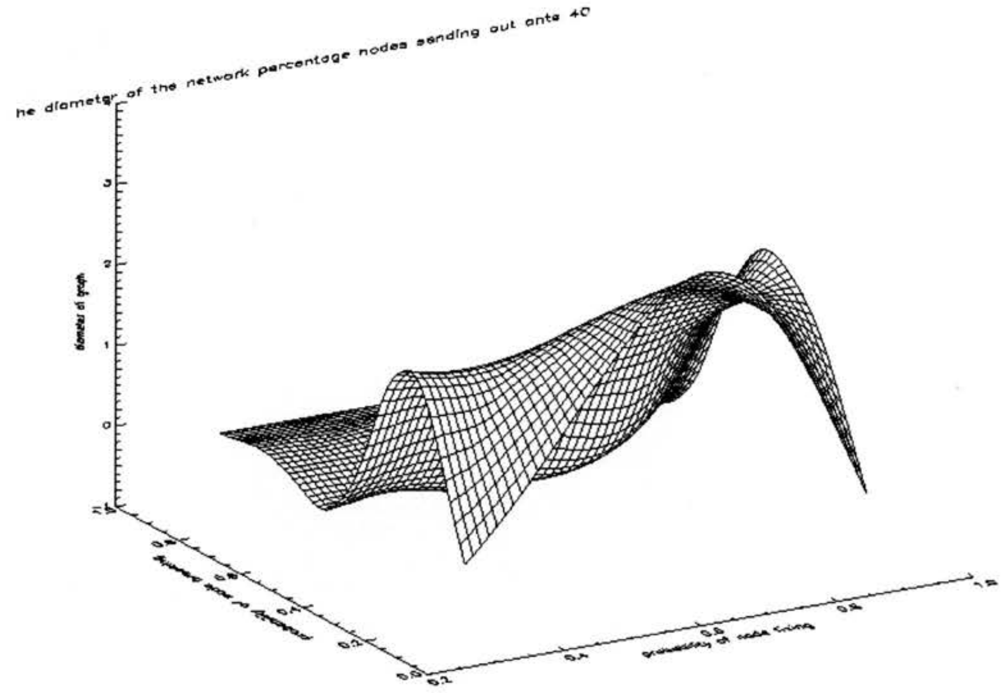


Figure 7.21: Graph of the diameter of the system for demand multiplier 40 percent, pheromone persistence rate 0.92, x axis probability of node fixing, y axis probability of node breaking, z axis diameter

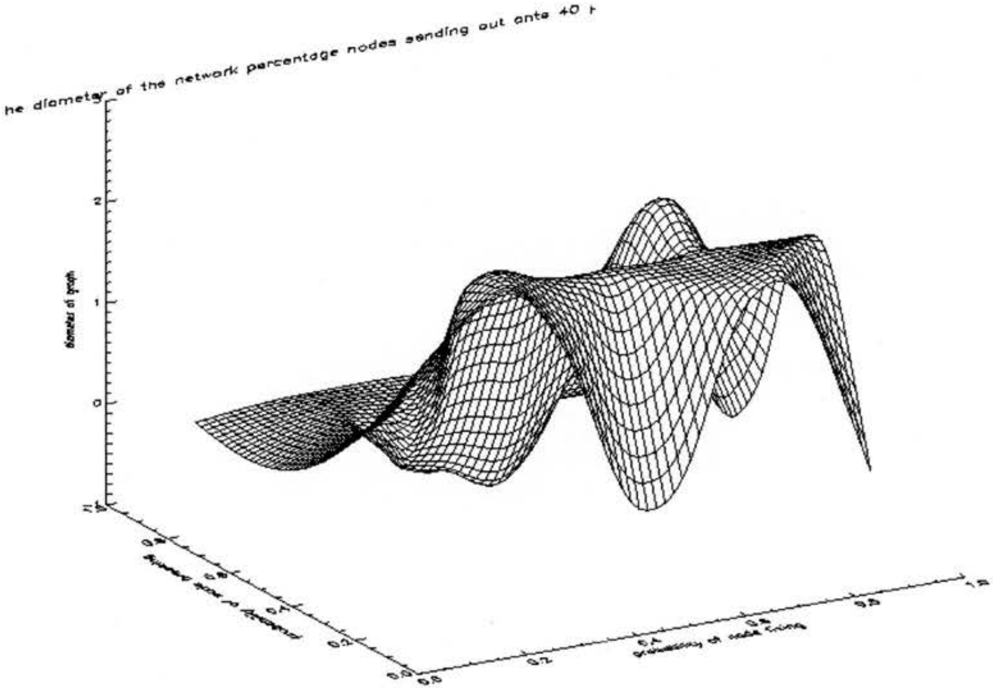


Figure 7.22: Graph of the diameter of the system for demand multiplier 40 percent, pheromone persistence rate 0.96, x axis probability of node fixing, y axis probability of node breaking, z axis diameter

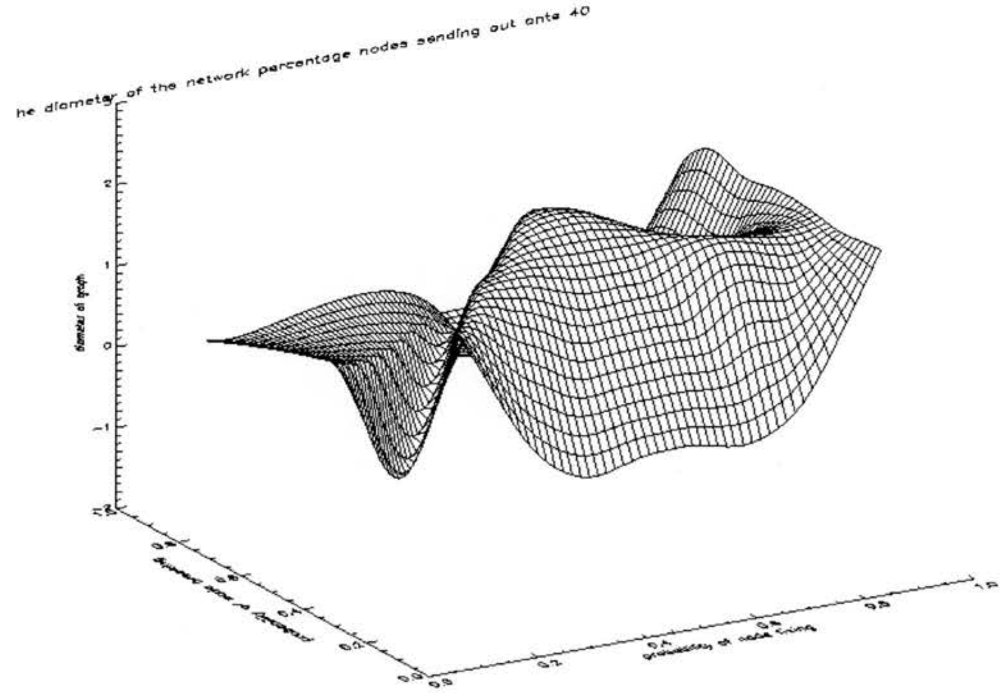


Figure 7.23: Graph of the diameter of the system for demand multiplier 40 percent, pheromone persistence rate 0.98, x axis probability of node fixing, y axis probability of node breaking, z axis diameter

From Figures 7.26, 7.25, 7.24, and 7.23, 7.22, and 7.21 we see that, in general, as the probability of node repair shrinks to zero and the probability of node failure increases, the diameter of the graph reduces to zero. A reduction in the pheromone persistence rate from 0.98 to 0.92 has the effect of lessening the area in configuration space within which the diameter shrinks to zero. In general, the maximum diameter remains the same (at 2 units) across all scenarios measured in the configuration space. Further, it seems that when the probability of node failure is greater than 0.6 the diameter of the graph shrinks to zero.

7.5.5 KL surface for the ant peer-to-peer system under churn conditions

In this section we consider how the Kullbeck-Leibler distance D_{KL}, as defined in Section 3.5.2, varies with the probabilities of node failure and repair, the pheromone persistence rate, and also the percentage of nodes sending out ants. The KL surfaces generated under varying values of pheromone persistence rates and demand multipliers are documented in Figures 7.32, 7.31, 7.30, and 7.29, 7.28, 7.27.

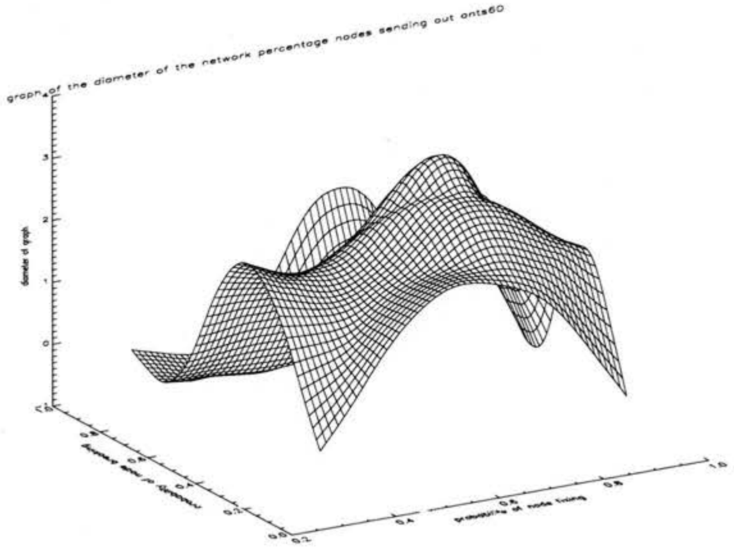


Figure 7.24: Graph of the diameter of the system for demand multiplier 60 percent, pheromone persistence rate 0.92, x axis probability of node fixing, y axis probability of node breaking, z axis diameter

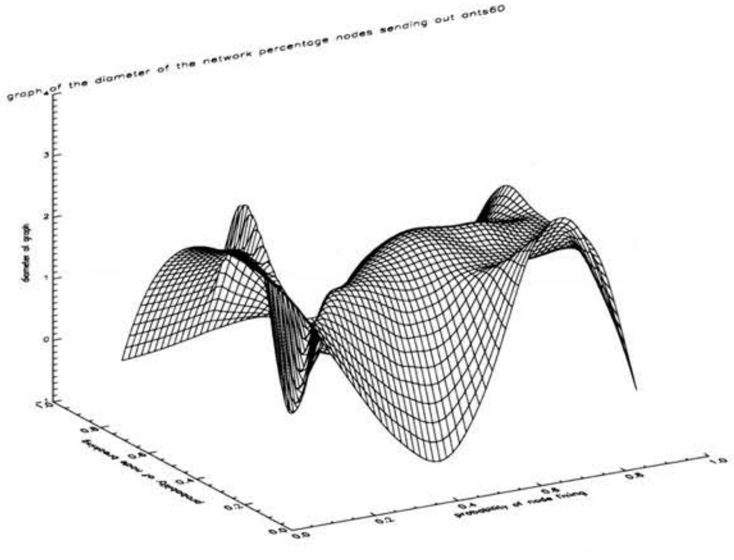


Figure 7.25: Graph of the diameter of the system for demand multiplier 60 percent, pheromone persistence rate 0.96, x axis probability of node fixing, y axis probability of node breaking, z axis diameter

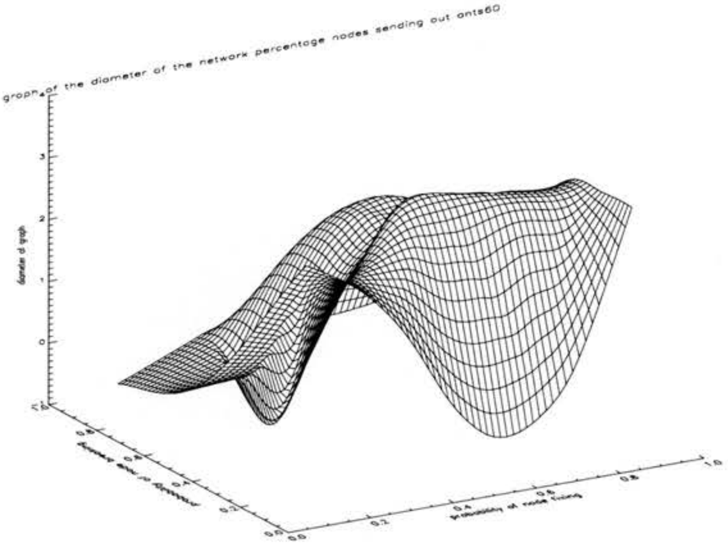


Figure 7.26: Graph of the diameter of the system for demand multiplier 60 percent, pheromone persistence rate 0.98, x axis probability of node fixing, y axis probability of node breaking, z axis diameter

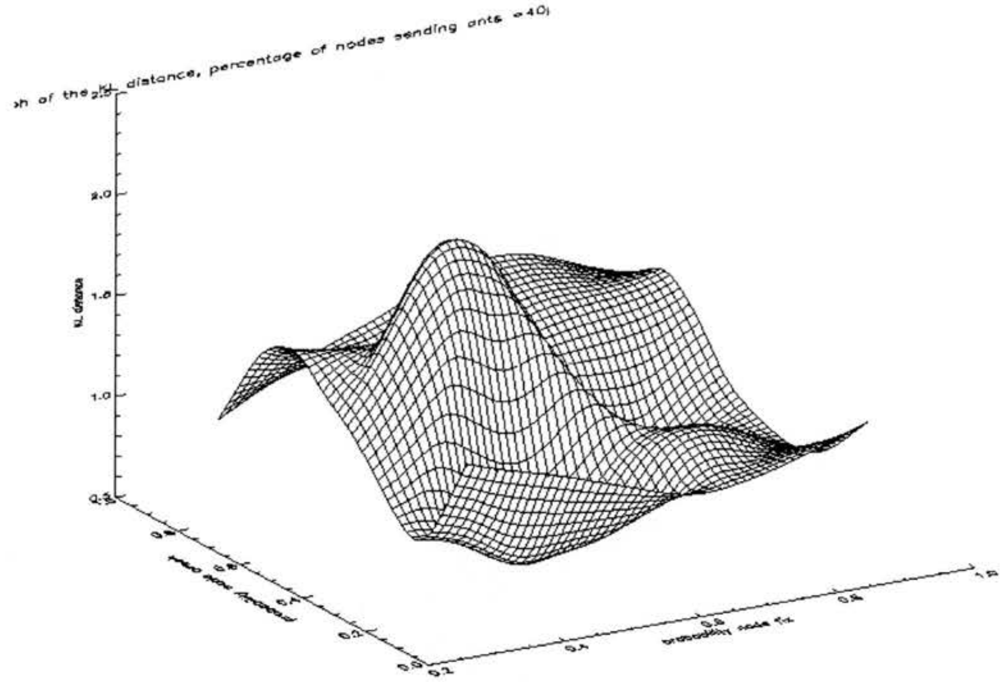


Figure 7.27: Graph of the KL surface for 40 percent demand multiplier, pheromone persistence rate 0.92 under churn conditions, x axis probability of node fixing, y axis probability of node breaking, z axis KL value

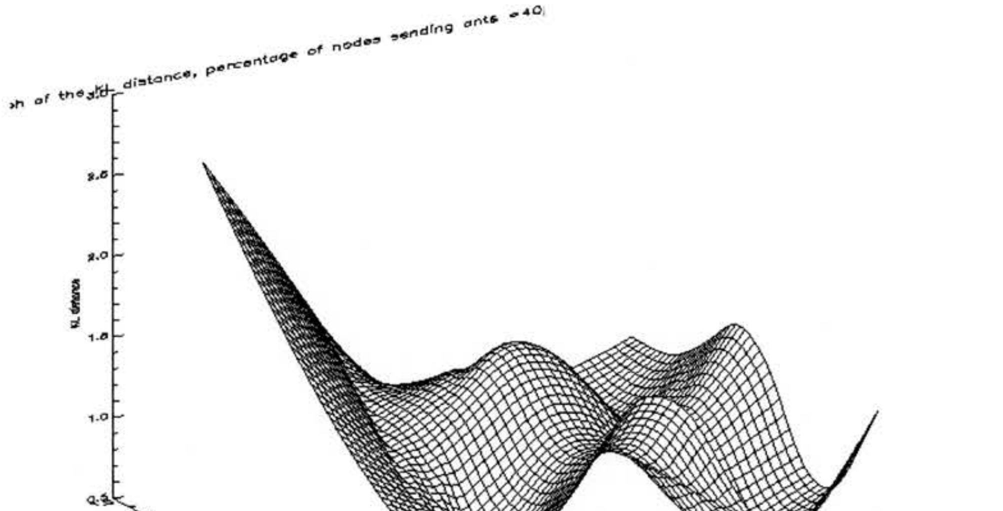


Figure 7.28: Graph of the KL surface for 40 percent nodes demand multiplier, pheromone persistence rate 0.96 under churn conditions, x axis probability of node fixing, y axis probability of node breaking, z axis KL value

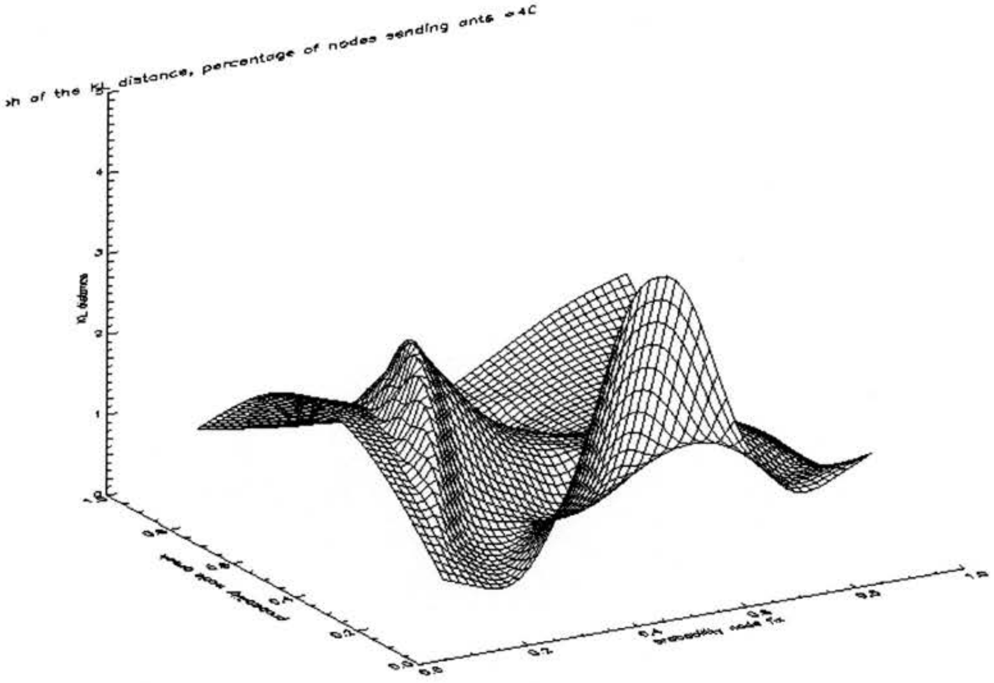


Figure 7.29: Graph of the KL surface for 40 percent demand multiplier, pheromone persistence rate 0.98 under churn conditions, x axis probability of node fixing, y axis probability of node breaking, z axis KL value

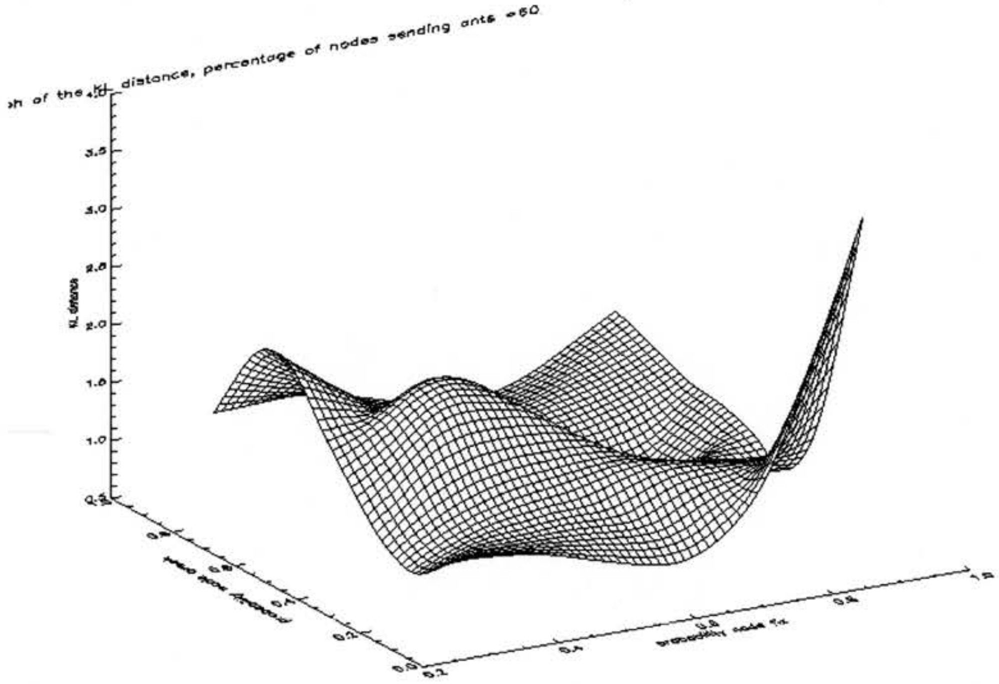


Figure 7.30: Graph of the KL surface for 60 percent demand multiplier, pheromone persistence rate 0.92 under churn conditions, x axis probability of node fixing, y axis probability of node breaking, z axis KL value

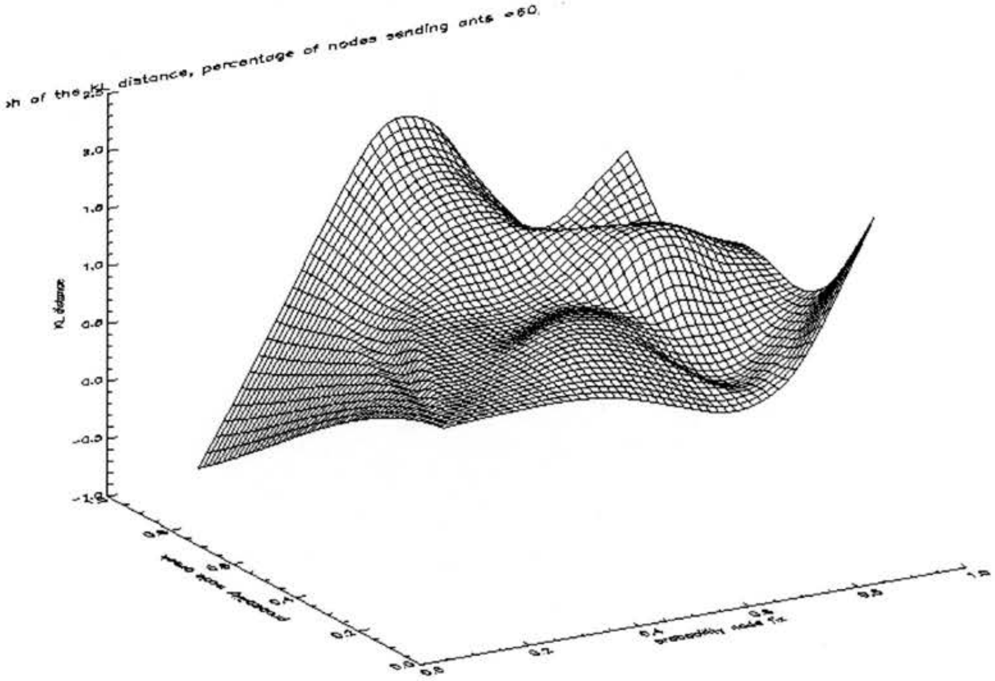


Figure 7.31: Graph of the KL surface for 60 percent demand multiplier, pheromone persistence rate 0.96 under churn conditions, x axis probability of node fixing, y axis probability of node breaking, z axis KL value

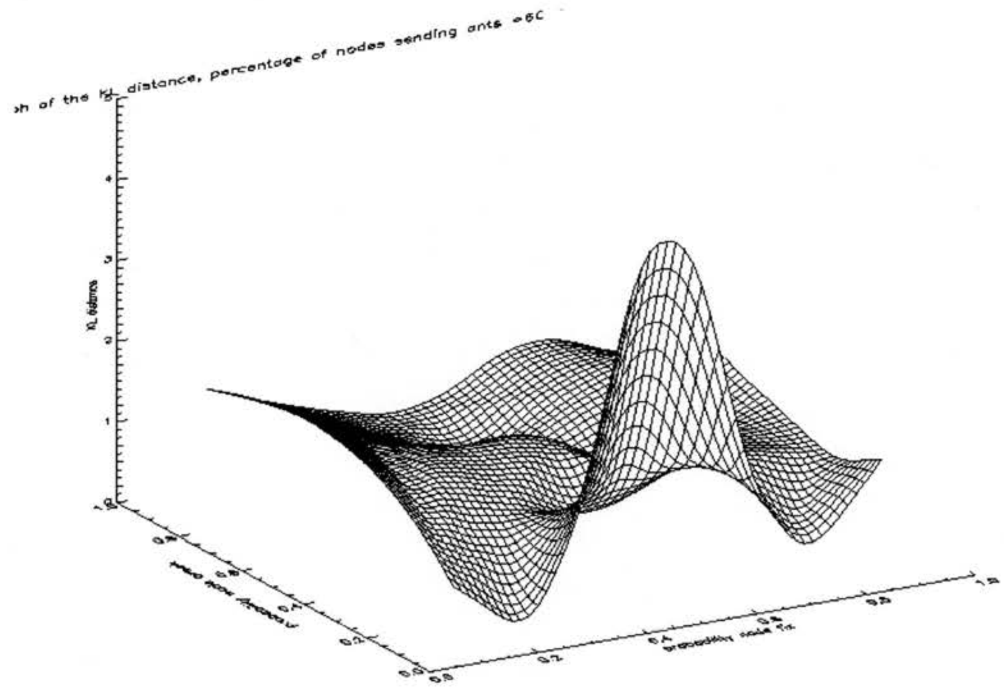


Figure 7.32: Graph of the KL surface for 60 percent demand multiplier, pheromone persistence rate 0.98 under churn conditions, x axis probability of node fixing, y axis probability of node breaking, z axis KL value

7.5.5.1 The effect of the probability of nodes breaking on the KL distance

In general, we see (from Figures 7.32,7.27,7.28,7.29) that as the probability of node failure increases there is an (in some cases gentle) increase in the KL measure. An explanation for this is that as the probability of node breaking increases, the average number of nodes operational at any time decreases, meaning that there are fewer links in the peer-to-peer graph. This increases the difference of the peer-to-peer graph from the random graph and the value D_{KL} increases.

7.5.5.2 The effect of the probability of nodes fixing on the KL distance

From the figures we see that mostly there is a flat response in the value of D_{KL} when changing the probability of node repair. There are a number of exceptions to this. Firstly, at low probabilities of node failure and high pheromone persistence rate (0.98), we see that the value D_{KL} experiences a maximum at intermediate values of the probability of node repair. It is under these conditions that ACO is able to cope with the churn in the system. When the probability of node repair is low, then over a period of time the nodes will become non-operational. When the probability of node repair is high, the majority of nodes in the network remain in an operational state. It is only at intermediate values of the probability of node repair that the churn in the peer-to-peer system forces new routes to be explored by the ants, and therefore appreciable structure is created in the graph.

7.5.5.3 The effect of the pheromone persistence rate on the KL distance

As the pheromone persistence rate changes from 0.98 to 0.96 and then 0.92, (Figures 7.29, 7.28, 7.27 and 7.32, 7.31, 7.30) the system becomes more responsive to changes in the environment, specifically churn in the system. So, in general, as the pheromone persistence rate reduces there is more volatility in the value of D_{KL}. We see that a maximum in D_{KL} emerges when the probability of node repair is small and the probability of node failure is large, at lower pheromone persistence rates. This is because under these conditions the proportion of operational nodes is small and also there is a great deal of churn in the system. This means that the structures that do form in the peer-to-peer graph will be small localised structures, as the churn in the peer-to-peer system will prevent long range connections. Under these conditions the distance from the associated random graph becomes large.

7.5.5.4 The effect of the percentage of nodes sending out ants on the KL distance

In all cases, reducing the proportion of nodes sending ants from 60 to 40 percent (compare Figures 7.29, 7.28, 7.27 against Figures 7.32, 7.31, 7.30) has a flattening effect on the KL surface at all pheromone persistence rates measured. This can be explained by observing that less pheromone is laid down when the proportion of nodes sending out ants reduces. This in turn means that the decay processes in the simulation tend to take over and this erases significant structure in the peer-to-peer graph, bringing it closer to the associated random graph and reducing the D_{KL} value.

7.6 Proposed modifications to ACO routing strategies

As mentioned above there are a number of effects on the system when a node becomes non-operational. Disruption of the flow of ants, and also disruption of control information are two such effects. We are motivated here by the third effect which is the invalidity of routing information in our routing tables, caused by non-operational nodes in our peer-to-peer system.

7.6.1 A proposed extension to ACO

In the standard implementation of ACO for telecommunications routing, the routing tables are maintained only by the variation of the amount of pheromone in the routing table entries. The mechanisms for renewal and decay of the pheromone values are seen as sufficient self-regulation mechanisms to support the identification of efficient routes in static environments. We can interpret the strength of pheromone on a particular route as being a measure of the efficiency of that particular route. However, as discussed above when considering the effect of damage to the system the issue of information quality (or the reliability of the information about that route) also must be considered and taken into account.

7.6.2 The effect of information invalidity

Having described the effect of damage on the system we focus here on the effect of invalidation of information due to damage. Let us now consider what happens when an ant returns to its source having completed a tour of our peer-to-peer system and the routing table of the source node is updated with the information that the ant has gained during its tour. In the ideal situation where there is no damage in our system all the nodes in the tour that the ant has taken are live. This means that, in the ideal situation, all of the routing information that updates the routing table of the source node is valid.

However, in the case where there is system damage the routing information that the ant returns to its source may not be entirely valid. This is because nodes that the ant has visited during its tour may have become non-operational in the time the ant has taken to return to its source. In this way the quality of the information that is deposited at the source node by the ant when it returns is related to the volatility of the environment in which the tour was made, and the length of the tour. However, a more direct measure of the quality of the information is in terms of the time taken for the ant to reach its destination. This is because if an ant gets stuck at a node due to the node going into a non-operational state, the ant does not make any progress along its path towards the destination node. However, due to the fact that time is passing the information is still becoming more and more out of date. It is the information that is deposited at the source node that is used in subsequent iterations to make forward routing decisions for the next batch of ants to be sent out. The effect of this mechanism is that the probability of routing subsequent ants down on routes that may be invalid is increased. If this is the case then those further ants are likely to be dropped by the system, as they will end up at nodes which are non-operational, and this will be reflected in the proportion of ants which do not make it home to the source.

7.6.3 ACO and the filtering of information

The main mechanism that ACO has for pruning routes in our peer-to-peer system is pheromone decay. This is a blunt instrument, as all the pheromone levels in the routing tables decay by the same proportion each iteration, unless they are renewed by the safe return of ants to their source node. The mechanism of pheromone decay does not distinguish between information that is recently gathered and information that is older in the routing tables of our nodes. Nor is any distinction made about the the age of the information at the point the information is added to the routing table by the ant. Here we detail how we would enhance the mechanism of routing table information update. In summary this is done by adding a mechanism that makes the routing decisions sensitive to the age of the information returned to the nodes by the ants. We explain the details below.

We can compare the latency of the information regarding the operational state of a node with the mean time to failure of that node and develop a scoring system. If latency of routing information is much smaller than the MTTF then the information held regarding that route is in good confidence. If latency of routing information is much greater than the MTTF then the routing is held in little confidence. By using the result of this comparison we can more selectively route ants within our peer-to-peer system.

The simulations performed in this work are discrete event simulations, so each time step a returning ant moves forward just one node along its visited list back toward its source. The position of the node in the ant's visited list gives us an indication of the latency of the information. This means that entries in the visited list of the ant have probabilities of being correct that depend on where the entry is in the visited list. If the entry is at the end (namely one of the first nodes to be visited) then that entry will be very old and will have a lower probability of being correct currently. Conversely, if the entry is at the beginning of the list, it will be more recent and will have a higher probability of being correct.

We can extend this analysis further by considering MTBF. This is because after the MTBF we can reasonably expect the node to have been broken and then fixed again. So in general the confidence that we ascribe to the node latency information decays as the latency increases, but then is reset to close to its initial value once the latency of the information exceeds the MTBF.

7.6.3.1 Information quality: A secondary metric

As stated in the previous section, when an ant returns to its source node from a successful tour it possesses information about its environment, and specifically information about the operational status of the nodes it has visited during its tour. If many ants return to a given source node having all followed different routes to different destination nodes, but many of them using the same small set of intermediate nodes on the way, we can accrue information about the operational status of these intermediate nodes at the source node. The key information to be kept and maintained is how many iterations ago the intermediate nodes were assessed to be operational. We can use this information to assign a confidence to the pheromone level that an ant deposits at a given routing table entry.

7.6.4 Proposal of a routing algorithm using information quality of the operational status of intermediate nodes

In this section we present a routing algorithm which takes information confidence into account. The purpose of this algorithm is to create a scheme whereby the forward routing policy of the nodes in the peer-to-peer system takes into account not only the efficiency of the routes available, but also their confidence. The algorithm has two parts. Firstly, an updating phase where information regarding the operational status of the surrounding nodes of a given node is maintained. Secondly, a forward routing policy which uses the certainty of the node operational information to create effective routing. Both of these phases are explained below. For the purpose of explanation we use a number of definitions.

- Random Ant — any ant which can be moving forward (toward its destination) or backward (toward its source).

- Argument Ant — a backward ant which has arrived at a node and is on its way from its destination back to its source.

- Ant Recipient Node — a node which is in receipt of an argument ant. This may be any node not just the source.

- Visited List — this is the list of nodes that the ant has visited during its tour. Note that this list is built up on the way from source to destination node. The list is then read in reverse order in order to determine the path back to the source node.

- Node latency information — the number of iterations that have passed between the present iteration, and the iteration in which the node in question was witnessed to be operational on the return path.

7.6.4.1 Updating of surrounding node operational status

This section describes actions performed when the Argument Ant arrives at the Recipient Node. The purpose of this part of the algorithm is to maintain information regarding the operational status of the nodes surrounding the Ant Recipient Node. It is anticipated that the node latency information is stored in the routing tables of the nodes in the peer-to-peer system. So, in contrast to the previous case where only pheromone levels are stored in the routing tables of the nodes, for this algorithm each entry in our routing table will have two values stored within it. Firstly the strength of the pheromone, between this node and the next-hop node, and secondly the node latency information for that particular route. The node latency information is simply an integer.

The process by which a node's operational status is updated is detailed below.

- Ant Recipient Node receives the Argument Ant.
- Ant Recipient Node interrogates the Visited List of the Argument Ant and collects information about how long ago (how many iterations have passed) all the nodes that the Argument Ant passed through were witnessed to be operational i.e. the node latency information.
- This node latency information is then tallied against prior node latency information for all the nodes in the routing table of the Ant Recipient Node.
  - If the Ant Recipient Node has seen the node in the Visited List more recently than the Argument Ant it has received then there is no change.
  - If the Ant Recipient Node has seen the node in the Visited List less recently than the Argument Ant has then the node latency information in the routing table of the Ant Recipient Node is updated.

7.6.4.2 Forward routing policy

On receiving an ant the node receiving the ant has to make a decision about the forward routing of that ant. The objective of this policy is to optimise two quantities. Firstly, the efficiency of the route, which is reflected in the strength of pheromone. Secondly, the certainty of the route, this is reflected in the node latency information.

7.6.4.3 Metrics for the forward routing policy

The simplest possibility for the forward routing probability is to combine the strength of the pheromone for an onward routing segment with a measure of the confidence in the information we have for that onward routing segment. We can calculate the confidence in the onward routing information from the expression above derived from our n-step Markov transition matrix, where the value of n is the Node latency information as described above. After the $MTBF$ there is a high probability that the node will have been fixed again after it has been broken. To reiterate, the confidence we then have in the information that a node is operational n time steps after the node has been observed to be operational is:

$$\frac{p_{break} + p_{fix}(1 - p_{break} - p_{fix})^n}{p_{fix} + p_{break}}$$

When routing forward ants we can use the product of the confidence measure stated here and the strength of pheromone for that onward route as the basis for our routing.

7.6.4.4 An aside on the two-state discrete time Markov model

We now state some relevant and standard results for the two state Markov Damage Model Section 7.3. Consider the transition matrix:

$$P = \begin{pmatrix} 1-\alpha & \alpha \\ \beta & 1-\beta \end{pmatrix}$$

It can be verified by induction that

$$(\alpha+\beta)P^n = \begin{pmatrix} \beta & \alpha \\ \beta & \alpha \end{pmatrix} + (1-\alpha-\beta)^n \begin{pmatrix} \alpha & -\alpha \\ -\beta & \beta \end{pmatrix}$$

therefore

$$P_{11}(n) = \frac{\beta}{\alpha+\beta} + \frac{\alpha}{\alpha+\beta}(1-\alpha-\beta)^n$$

where $P_{11}(n)$ is the probability that the Markov chain is in state 1 n steps after it was observed to have been in state 1. The interested reader is referred to [71].

This is exactly the probability that we need for the enhanced version of the ACO routing algorithm as described in Section 7.6.4, and it can be shown to be the expression we use in the algorithm by replacing α with p_{fix} and β with p_{break}.

7.6.4.5 Circumstances under which the enhanced ACO routing algorithm applies

Here we consider the circumstances under which the enhancements to ACO would be necessary. As mentioned above in Section 7.4.1 the points in our configuration space have unique values for both churn and the steady state probabilities of nodes being in the operational and non-operational states. When considering whether the enhancements to ACO routing in our peer-to-peer system would be valuable we are best informed by considering scenarios that differ in the amount of churn in our system.

If the churn value is low in our system then we may expect that the ACO algorithm will be capable of dealing with change in the system at this frequency. This is because nodes that are in the non-operational state will be incapable of processing incoming ants which means that ants routed to non-operational nodes will not be forwarded on, and the ant will therefore not ultimately return to their source node. This has the consequence that the route leading to a node which has entered the non-operational state will not have pheromone added to the corresponding routing table entries, and therefore will be pruned from the system by the normal decay processes. In summary we would expect ACO to be able to cope with low churn in the system.

If the churn value is high in our system we can assume that the probability of the nodes being in the non-operational state is determined by the steady state probability as described above in Section 7.3. This means that the confidence that a given node has that the surrounding nodes are operational is uniformly the steady state probability. In this scenario there is therefore no value in using a measure of confidence for the operational state of the node in the routing algorithm, as the confidences we ascribe to information regarding the operational state of neighbouring nodes are all the same, and can therefore not be used as a differentiator in routing decisions. If the churn value is intermediate then we may use the enhanced routing algorithm detailed in Section 7.6.4. This is because both of the following conditions are true. Firstly, the ACO algorithm will not be able to cope as the churn is not low, and secondly, the confidences we have in the operational statuses of the neighbouring nodes will not be the same from node to node as the churn is not high. In summary, the ACO algorithm will not be able to cope and also there is value in using confidences about the operational status of the neighbouring nodes.

7.7 Conclusions and further work

In this section we consider both the conclusions of the experiments and also the further work stemming from it. In order to do this we must first sum up the experimental results.

7.7.1 Damage experiment conclusions

In all the experiments documented in this chapter we see that the system has a tolerance for churn as long as the following boundary conditions are adhered to: $P_{fix} > 0.6$ and $P_{break} < 0.4$. Outside of this region of the configuration space of our damage model we see that there are significant reductions in the parameter values of average degree, average path length, the proportion of the ants reaching home, and also the diameter of the graph. These results indicate that there is only so much churn that the ant peer-to-peer system (based on the original version of the ACO routing algorithm) can handle. Further, we see that the proportion of ants reaching home in general is low, but, very quickly deteriorates as the churn in the system increases. The deterioration is more dramatic under these failure scenari os, than in the burstiness experiments. We conclude that this is the effect of information invalidity (as described above) in the routing tables.

7.7.2 Further work for damage experiments

From the conclusions in the previous section we can derive a number of changes that could be made to the system, and also experiments in order to generate further insight into the behaviour of the system under conditions of churn. These are listed below, and it is intended that each change could be performed sequentially, in order that the effect of each change could be assessed on the system in order that further adjustments to the experimental conditions be made if needed.

- Considerably reduce the value of the demand multiplier used, and also the amount of pheromone laid down by each ant passing through a node when conducting further experiments. In this way we seek to avoid overall congestion of the system, and enable the ACO reinforcement mechanism of routing information in the node's routing tables.

- Increase the initial radius of awareness of the nodes, at the beginning of the simulation runs to values in excess of the connection threshold. In this way we would be able to create a diverse population of possible routes in the routing tables of the nodes which assists in alleviation of congestion, and also the building of a network resilient to node failure.

- Focus on a smaller range of parameter values for our failure model. We have seen from the experiments conducted here that the system breaks down outside the region of $P_{fix} > 0.6$ and $P_{break} < 0.4$. Further experiments could be conducted at the border of this region in order to examine how the system breaks down. Also the effect of changing the balance between the number of ants and the amount of pheromone deposited on the region of operation could be assessed.

An additional avenue is to explore the effect of elaboration of the failure model on the peer-to-peer system. Two possible directions present themselves:

- Firstly, the effect of the variation of the values of p_{fix}, and p_{break} over time could be investigated. As mentioned above one profile could be in terms of reliability growth models, where the reliability of the system changes over time, perhaps reflecting the increased proficiency with which the peer-to-peer system learns to recover from failure over time.

- Secondly, the inclusion of further states in the damage model, where there are more than one operational and non-operational states would enable a more accurate model of the set of node behaviours to be modelled. As mentioned above, each state could indicate a precise list of functions that a node was able to perform. This would profoundly affect the conditions under which transitions between operational and non-operational modes of behaviour occurred.

Both of these enhancements to the failure model would lead to a far more complicated configuration space for the model. Just as with the changes to the experimental conditions, these enhancements would have to be implemented incrementally, with the effect of each change being base-lined against the previous version of the model.

Chapter 8

Conclusions and Further work

8.1 Introduction

In this chapter we draw the work contained in this thesis together. Specifically, we summarise the motivation behind the work, and relate this to the main contributions contained in this thesis. We substantiate the main contributions by interpreting the experimental results, which leads into a general discussion of the contrasts between congestion and failure scenarios in networks. In conducting the work in this thesis insights from a number of different perspectives have had to be combined. This provides us with a number of avenues of further work to derive further insight. The structure of the chapter (and particular sections) are indicated below.

8.1.1 Chapter structure

This chapter is structured as follows. Section 8.2 discuss the motivations for the thesis, and touches on the implications of the simulation model used. Section 8.3 indicates the main contributions of the thesis from the theoretical work through to the experimental work. We also give an interpretation of the experimental results in this section. We present a discussion of the contrasts of system failure due to fault and congestion in Section 8.4. Future work is then presented in Section 8.6.

8.2 Motivation

This thesis has been motivated by the desire to both characterise and understand the way in which peer-to-peer systems self-organise. However, the main question this thesis is concerned with (as stated in the introduction) is how the behaviour of the ant-based peer-to-peer system can best be measured using a simulation-based approach, and how these measurables can be used to control and optimise the performance of the ant-based peer-to-peer system in conditions of equilibrium, and also non-equilibrium (specifically varying levels of bursts in traffic demand,

and also varying rates of nodes entering and leaving the peer-to-peer system).

The reason for this choice of scenarios is that both congestion and damage cause the peer-to-peer system to fail, in subtly different ways. This failure manifests itself both with respect to the values of topological metrics and also the proportion of the total traffic that is routed effectively. The inclusion of the equilibrium demand scenarios in this thesis enables us, through comparison with the bursty and damage scenarios, to consider the effect of changes in the environment, and also changes in the system, on the system.

The dynamic nature of peer-to-peer systems (with nodes continually entering and leaving) places great demands on the routing mechanisms used to route traffic within them. The application of complex adaptive systems (such as genetic algorithms or neural networks or as used here, ant colony optimisation) to this dynamic routing task is new at the time of writing. The main open question is under which conditions do these complex adaptive systems operate well in the context of peer-to-peer systems A further question is why do complex adaptive systems operate well or badly in given conditions. The peer-to-peer system we consider in this thesis uses ant colony optimisation (ACO) as the basis for the traffic routing calculations within the nodes of the system.

Models of the way in which networks form have been presented especially with respect to the internet. The most well established one is that of preferential attachment. Preferential attachment is contingent on two assumptions being true:

- The network has to be growing in terms of the number of nodes within it.
- The probability of an existing node forming a new connection to another node must be proportional to the degree of the node from which the connection is being formed.

Both of these assumptions cannot be relied upon to be true when considering the way in which peer-to-peer systems operate. A further motivation for the work in this thesis has been the need to develop models that do not make assumptions about the way in which networks grow and can be used in the description of evolution of networks in peer-to-peer systems.

8.2.1 Implications of the simulation model

In order to investigate the behaviour of the ant colony optimisation algorithm in peer-to-peer routing a simulation model was developed with the intention of capturing the main categories of behaviour of the system. This simulation model was implemented in the form of the simulator detailed in Chapter 4. In order to adequately describe the nature of the operation of the ant peer-to-peer system a total of 20 parameters have been used in the routing calculations that are performed during the course of the simulation runs. There are three main categories of control, as described in Chapter 4.

- Overall system control — here we configure the global variables in the system such as the number of nodes and grid size.
- Node control — here we configure the queue lengths, ranges of awareness and pheromone threshold of the nodes.
- Ant control — here we configure the amount of pheromone that is deposited by the ants at the nodes in the system.

The implications of the simulation model are that the configuration space of the ant peer-to-peer system is huge. This brings the possibility that there may be more than one point of optimum performance in the configuration space. However, this is further complicated by the fact that the ACO algorithm is stochastic in nature so average values for the values of measurables have had to be generated using multiple simulation runs starting from the same point in configuration space.

The following sections detail both the main contributions and conclusions from the research work presented in this thesis.

8.3 Thesis overview

This section provides a summary of the work presented in the thesis outlining its major contributions to the field and emphasising the work's novelty.

The first major contribution is the theoretical framework which describes the general mechanisms at work within a peer-to-peer system, irrespective of the method of routing the traffic. The framework also sets out a method for relating the values of the initialising parameters of the peer-to-peer system to the outcome of the self-organisation (in terms of network topology, and network traffic profiles). Within the framework we distinguish between the processes which happen on a short time scale, and those that happen over a long time scale.

We identify the main parameters that drive the routing decisions in the ant peer-to-peer system. We also identify some generic measurables which enable specification of the state of the peer-to-peer system. The framework allows for the variation of load placed on the system, and also the existence of churn (that is nodes entering and leaving the peer-to-peer system) within it. We model churn using a reliability model which we applied to all the nodes in the peer-to-peer system. The reliability model indicates the probabilities under which nodes transition from operational to non-operational states. In many cases churn arises because nodes voluntarily enter and leave the peer-to-peer system. This voluntary entering and leaving of nodes would be modelled by the same mechanism. This framework can be applied to any peer-to-peer system, in order to understand its behaviour.

The second contribution of this thesis is the definition of a method for determining the distance (KL-distance) of a given topology from a reference topology (for example, a lattice, tree, or random graph). We do this by considering the degree distribution (or the distribution of the number of connections between nodes in the peer-to-peer system) of the topology in question, and compare this to the degree distribution of the reference topology. We know the functional forms of the reference topologies and can measure the distance between the reference topologies and network topology using a divergence measure. We show that this method can be applied to experimental data using a random graph as the reference topology, and can be considered to be a measure of structure or self-organisation when using a random graph as the reference distribution. This method can be extended to include other forms as reference topologies (e.g. lattices or small world graphs).

We present the results of the equilibrium experiments which document the behaviour of the ant peer-to-peer system under a range of levels of traffic (note the traffic levels remain the constant throughout a given simulation) by considering both network topology, and the proportion of traffic successfully routed. The overall results of the simulations were analysed by relating them back to the initial conditions used to initiate the simulations (namely the initial range of awareness, the pheromone decay rate, and the percentage of nodes in the system sending out ants every simulation iteration). The contribution is that experiments show that the conditions under which the topological metrics are optimised (path length and diameter of the graph) are opposite to those under which the traffic metrics are optimised. We also show that the KL-distance can be used to create a surface relating the distance from randomness of the peer-to-peer graph topology to the values used to initiate the simulations (namely the initial range of awareness, the pheromone decay rate, and the percentage of nodes in the system sending out ants every simulation iteration).

We present the results of the burstiness experiments contrast with the results of the equilibrium experiments in the following respects. There is a greater variation in the values of the topological metrics (average degree, average path length, and diameter) in the simulation scenarios shown. The proportion of ants reaching home is similar to those for the equilibrium scenarios. The contribution is that we see that, in general, the values of the measurables can be optimised by altering the value of the pheromone persistence rate so that it is closer to unity as the burstiness of the demand reduces, and vice versa. This can be explained by observing that when there is little pheromone evaporating the routing tables are rigid. In other words, the routes (through the peer-to-peer system), encoded by the routing are constant. The constant route scenario is most efficient when there is a constant demand placed on the system. However, when there is variation in the traffic levels, we need to allow the routes in the routing tables to adapt, in order to enable the pruning of bad routes to make room for efficient routes on a dynamic basis. In the ACO scenario this is done by allowing pheromone decay.

The results of the damage experiments contrast with those of both the burstiness, and equilibrium experiments. The contribution is that the damage experiments show that even though the damage model is a simple one, churn in the system reduces the proportion of ants successfully arriving home in comparison to the equilibrium and bursty scenarios. This indicates that the network structure which has arisen in the equilibrium and bursty experiments is compromised by the churning of the nodes in the system. An explanation for this is the effect of inaccurate routing information in the peer nodes due to uncertainty of the operational state of the peer nodes. Examination of the topological metrics (average path length, diameter, and average degree) reveals that in all cases there is an abrupt transition from a value reflecting structure in the system (when mean time to repair (MTTR) is low, and the mean time to failure (MTTF) is big) to a value reflecting little structure in the system (as the rate of churn in the system increases and the steady state proportion of nodes in the operational state decreases). This indicates that there is a threshold churn level beyond which the ant peer-to-peer system cannot function (i.e route traffic adequately). From the experiments we see $p_{fix} > 0.6$ and $p_{break} < 0.4$ defines a region where the ant peer-to-peer system can function. Outside of this region the values of the measurables collapse, and there is no significant structure.

Finally, we conjecture that a routing algorithm which uses confidence in the routing information in the routing calculation, used to determine the next hop for the ants in the peer-to-peer system, would improve the behaviour of the system under fault conditions in nodes. We define such an algorithm in this thesis in Section 7.6.4.

8.3.1 Overview and summary of experimental results

In summary, the experiments show that there is a complex interplay between each of the variable values that initiated the simulation runs (range of awareness, pheromone decay rate, demand multiplier, Hurst exponent, and the probabilities of node failure and node repair). In general, the proportion of ants that return home is very small under all scenarios measured. We conjecture that this is because of high levels of traffic n the system causing congestion which prevents the ants from reaching their destinations. As the ants are prevented from reaching their destinations the mechanism of feedback of information in terms of reinforcement of pheromone is prevented. This in turn prevents valid routing information from being disseminated through the nodes in the peer-to-peer system. Five trends emerge from the experiments.

1. From the equilibrium experiments we see that the topology of the network becomes more feature-rich (reflected by an increase in the average degree) as the initial radius of awareness of the nodes increases.

2. We see from the burstiness experiments, that other topological metrics (such as the path length) can be improved by reducing the demand multiplier value for the simulation

which in turn increases the diameter of the network. We see that when considering burstiness in the demand function the proportion of the ants reaching home is in the same order of magnitude as in the equilibrium scenarios. We observe a second trend that is that by moving the value of the pheromone persistence rate closer to one as the burstiness decreases (and vice versa) we can improve the behaviour of the system (as reflected in the values for the average degree, proportion of nodes reaching home, and average path length).

3. We see that optimising these quantities (the average degree, proportion of nodes reaching home, and average path length), under bursty conditions, tends to increase the value of the diameter of the graph.

4. In the case of the damage experiments the main trend is that there is a small window within which the values of the metrics used to measure the peer-to-peer system remain within acceptable values. Outside this window the values of the topological and traffic metrics approach zero. In general we see that the probability of node repair has to be high, and the probability of node failure has to be low in order for there to be an appreciable path length and diameter in the system.

5. Contrasting the damage and bursty scenarios we also see that the proportion of ants returning home is at least an order of magnitude lower in the damage scenarios than in the equilibrium and bursty scenarios. We conjecture that this change is due to the invalidity of information in the routing tables.

We note, however, that under all experimental conditions measured the proportion of ants returning home is very small. In Section 8.3.2 we give an overall interpretation of the experimental results contained in this thesis. We suggest ways of enhancing the performance of the ant peer-to-peer system and also avenues for further work in Section 8.6. In Section 8.4 we compare node fault, and node congestion scenarios.

8.3.2 Interpretation of the experimental results

In both the damage, and also the burstiness, experiments the initial radius of awareness has been set to be the connection threshold (as defined in Section 5.2.2). The intention of using this value for the initial radius of awareness was to test the hypothesis that the ant peer-to-peer system could evolve a network, through discovery, starting from a sparse substrate graph. From the experiments we see that the ant peer-to-peer system is not working efficiently, and one possible cause of this is that the network is not forming under the conditions of the simulation runs. This leads us to conclude that a more dense initial substrate graph may be required in order to enable the network of the ant peer-to-peer system to form.

An alternative interpretation is that ants are being prevented from getting to their destination by congestion in the network in which case far fewer ants need to be used in the course of the simulation runs. However, the amount of pheromone being deposited by each ant must increase in order to counteract the effects of evaporation of the pheromone in the routing tables.

From the experimental results we are unable to determine exactly the cause of the lack of adequate traffic routing. However, we clearly see that under the conditions of the simulation experiments, the ant peer-to-peer system has entered a brittle state where few ants are returning successfully to their source node.

8.4 A comparison of node fault and node congestion scenarios

Both fault and congestion conditions prevent a node from being able to process traffic effectively. However, congestion and fault conditions in nodes have different properties and effects on the overall system. We contrast here the scenarios of fault and congestion events from the following points of view:

- The probabilities the events occur,
- The effects on routing table information,
- The nature of any cascading effects in our system,
- The distribution of the nodes that are not processing ants.

8.4.1 Probabilities of fault and congestion

In the node fault scenario used in this thesis, since we assume all node failures are independent, we can calculate from the steady state probabilities of our Markov model the probability of a node being operational, and from this calculate the proportion of the nodes in our population that are operational and non-operational. At steady state these probabilities remain constant. In the case of congestion failure it is not so simple to calculate what the proportion of the total number of nodes that are congested is. However, we can, in principle, use the number of ants sent, average queue length, and average tour length of an ant in order to calculate the probability that a node is congested in the peer-to-peer system.

8.4.2 Effects of faults and congestion in our peer-to-peer system on the routing information held in routing tables

As described above, when a node fails it ceases to process ants, and the ants in its queue are all suspended in the queue, until the repair event happens. While the ants are suspended the information contained in the visited lists of the ants is becoming out-of-date. The reason for this is that while the ants are suspended the rest of the peer-to-peer system is continuing to change (i.e. the operational state of the remainder of the nodes in the system is changing). When the ants are released from their suspended state there is a decreased level of certainty that the nodes in the visited list of the ants remain in an operational state. By comparing the number of iterations a node is operational against the MTTR we can determine how dated the information in the visited list of an ant is on average (that is, how many iterations an ant has been in a suspended state). Specifically, we can assume that the number of simulation iterations the node has been in a non-operational state for is the MTTR as defined in Section 7.3.2. We can use this information to ascribe a confidence to the routing information held in the routing tables of our nodes, by assuming a two-state reliability model for the node and using the expression for confidence in Section 7.6.4.3. .

In the congestion scenario where nodes cease to process ants purely for congestion reasons there is no clear way of ascribing a confidence to the routing information held in the routing tables of our nodes. In fact congestion prevents ants from returning to the source node. In other words, it is not easy to reason about which nodes referred to in the routing tables of a central node will be congested and which will not.

In summary, in both the failure and congestion scenarios it is not possible to say with certainty which nodes are processing ants, and which are not. Furthermore it is not possible to say with certainty whether the reason for lack of ant processing is for congestion or failure reasons. The distinction is that, as described above, we can calculate a confidence value for the routing information in the failure case. This can in principle be used to modify the routing table entries when a central node recovers.

8.4.3 Cascading effects due to fault and congestion

Let us consider what happens when one node ceases to process ants. As described above it stops processing ants, and through the reinforcement mechanism of ACO, the routes leading to the non-operational node are not renewed. As a consequence of this the system finds alternative routes, and the traffic that would have gone through the node is diverted to neighbouring nodes. The chances that the neighbouring nodes will then become non-operational due to congestion is then increased. So even though we have chosen to model node failure as independent events, this may result in node congestion cascading throughout the system. This effect does not, however, mean that the assumption of independent node failure events is invalid.

A similar cascading process will be observed in the case of congestion at nodes. If a node becomes congested and ceases to process ants then routes to the congested node will not be renewed and the traffic to the congested node will be diverted to neighbouring nodes in a similar way to the scenario above.

8.4.4 Distribution of nodes not processing ants due to either fault or congestion

The final aspect to consider is the distribution of the non-operational nodes in our peer-to-peer system. In the failure case, the distribution of the non-operational nodes is random throughout the network. Each node has a set probability of being operational or non-operational as described by the steady state argument detailed in Section 7.3. In the case of the congestion scenario we can conclude that the nodes most likely to become congested are those of high degree. This leads to a far more localised distribution of nodes with non-operational states, than in the failure case. In both the node fault and node congestion scenarios the node that has initially begun to cease processing ants may also become surrounded by a collection of neighbouring nodes which have become congested due to the load balancing properties of the ACO algorithm. However, this is a secondary effect.

8.5 Assessment of the degree to which the main question of the thesis has been answered in the work

In this section we assess the effectiveness of the work in answering the main question of the thesis as stated in Section 1.1. We re-state the question here.

The main question this thesis is concerned with is how the behaviour of the ant-based peer-to-peer system can best be measured using a simulation-based approach, and how these measurables can be used to control and optimise the performance of the ant-based peer-to-peer system in conditions of equilibrium, and also non-equilibrium (specifically varying levels of bursts in traffic demand, and also varying rates of nodes entering and leaving the peer-to-peer system).

The work in this thesis shows that insight can be derived into the behaviour of the peer-to-peer system under both equilibrium and non-equilibrium conditions by considering the topological metrics of the system (such as path length, degree distribution, and cluster distribution) in combination with traffic metrics (such as the proportion of the ants home). Furthermore, we see that the system does respond to changes in the pheromone persistence rate, and that varying the pheromone persistence rate in response to the changing nature of the system's environment may provide a mechanism for optimising the performance of the system. We see however, that the effect of varying the pheromone persistence rate cannot compensate for varying degrees of confidence in the entries in the routing tables (as created in the node failure scenario). In the failure scenario a finer method of control must be used which takes into account the age of the information, and guages the confidence in the information accordingly.

The usefulness of the measurement framework detailed in this thesis is limited, and this is reflected in the degree of insight generated by the experiments documented in this thesis, as further detail needs to be accounted for in order to gain a more complete picture of the response of the ant-based peer-to-peer system to its own self-organisation and also change in the external environment. This supplementary information falls into three categories:

- Detailed statistical measurements of the path length, degree distribution and cluster distribution. By this we mean observations like the maxima and minima, standard deviations and higher moments of the distributions concerned. We are also concerned with KL distances using distributions other than the poisson distribution as the reference distribution. In this way we could generate a more complete picture of the nature of the distributions and also the similarities of the topology of the system to reference topologies.
- Correlation measurements between the topological and traffic metrics. Here we are concerned with effects that occur in synchronisation with one another, such as abrupt changes in average cluster coefficient and average degree values. In this way we can understand the trade-offs that the system is having to make as the self-organisation occurs.
- Time-dynamics of the traffic and topological statistics. Here we are concerned with how the values of these metrics change with time as the self-organisation proceeds. Specifically we are interested in the timing of abrupt changes in the values of the metrics (i.e. the number of simulation iterations at which they occur). This insight would enable us to determine the maximum rate of change in the environment that the self-organising system could cope with.

8.6 Future work

The work presented in this thesis can be extended in a number of directions. In this section we indicate the major directions for further work. We conjecture two possible causes of the low traffic metrics observed in the experiments detailed in this thesis. It may be that either the ants are being lost due to congestion, or that the initial radius of awareness and node density are insufficient for a rich network.

The first thing to consider is the parameter values that initialised the equilibrium, bursty, and damage experiments. Alteration of these parameter values without fundamentally changing the underlying model, would enable a more precise understanding of the response of the ant peer-to-peer system to constant levels of traffic, burstiness in traffic, and also churn. We now detail the possible new conditions for each of the three sets of experiments.

In all cases (equilibrium, burstiness, and damage) an examination of the behaviour of the ant peer-to-peer system under far smaller values of demand multiplier, with each ant depositing a greater level of pheromone, would be valuable. Such a study would prevent the overloading of the network in the peer-to-peer system, and enable the freer movement of ants within it.

Once the effects of this change have been understood the effect of changing the node density on the grid could be examined. In this case the key observation would be evidence for the existence of a threshold minimum density below which the network does not form. Conversely, it would be instructive to consider what happens at the opposite extreme when the node density is very high.

For the damage experiments there is a further enhancement, which is the alteration of the range of churn values that the experiments are performed over. One would expect there to be a transition at certain churn levels between the system generally operating, and the network and system breaking down (this is likely to be characterised by a steady deterioration in the traffic statistics for the system as a whole). Under these transition conditions it would be instructive to examine the effect of the variation in the pheromone decay rate on the topological and traffic metrics.

Focusing on the failure aspect of the system, we see that a more realistic simulation model could be created by the use of a more elaborate failure model. Possible enhancements could be the use of intermediate states denoting conditions of partial failure or operation where the permitted operations of the nodes are significantly reduced. Cascade failure could be simulated by coupling the probabilities of a node failing, to an aggregate metric such as the number of nodes, or the number of neighbouring nodes, already in a failed state. This would break the assumption used in this thesis that the nodes in the ant peer-to-peer system fail at the same rate, and independently. Reliability growth models could be used to reflect the increased efficiency of the system as the system learns to cope with failures more effectively.

In this thesis we have discussed the effect of the confidence in information in the routing tables of nodes in failed states. We also suggested it may be possible to exploit these varying confidence values in routing algorithms, and traffic routing calculations. An algorithm has been defined to do just that. This algorithm could be implemented and tested against routing algorithms which do not use measures of the confidence in routing information in calculation of routing decisions. A criterion of success for this algorithm would be that the use of confidence measures enables routing of a greater proportion of traffic under conditions of higher churn.

There is a connection between the way in which networks rearrange themselves and statistical mechanics [20]. By considering the degree distribution we can infer the probability rule for the formation of a new connection of a node with existing degree K. This analysis is done by considering how re-wiring one of the connections of a node of degree K to an alternative destination node, affects the degree of the source node, the original destination node of the link, and the final destination node of the connection. The analysis for this is detailed in [19]. Note that this analysis is contingent on the validity of the assumption that capacities of all edges in the graph are equivalent. We note that the graph is a directed graph.

We can consider the process of re-wiring in a network to be similar in concept to a redistribution of balls in boxes. By imposing the constraint that any given redistribution of the balls in the boxes preserves the total number of balls in the system we can derive a relationship between the probability distribution that a given box has a given number of balls, and the probability that a further ball would be added to a box with a given number of balls in it. By transformation we see this provides us with a relationship between the overall degree distribution of a graph and the probability that new connections will appear between individual nodes of degree K. The relationship is as follows:

$$f(K) = \frac{(K+1)P(K+1)}{P(K)}$$

where K is the degree of a node in the peer-to-peer system, $f(K)$ is the degree distribution of the individual node, and $P(K)$ and $P(K+1)$ refer to the overall degree distribution of the network.

This model gives us a simple relationship between the microscopic and macroscopic properties of a network, under conditions where the total number of edges in the network remains constant. In the case of the ant peer-to-peer system this is not valid, as the capacities of the links in the peer-to-peer system vary. Further work in this instance would involve the investigation of a simple model with the probability of a new link being formed of $f(k)$ where all the edges in the graph are of the same importance (or capacity). These models would help in generating insight into the way in which the macroscopic structure of a network is related to the microscopic structure, and vice versa. A further step would be to exploit this insight by programming microscopic growth patterns to aggregate into graphs with desired macroscopic topological properties.

We have shown in this thesis that it is possible to measure the distance of a given topology from a reference topology by the use of the KL-distance (D_{KL}). A wider application of the distance D_{KL}, would enable topological control by using this distance to guide and correct growth processes in networks thereby supporting of the goal to create graphs with desired topological properties. Specifically, this could be done by taking repeated measurements of the value of D_{KL} as candidate changes to the network topology are evaluated. Only those candidate changes with a favourable effect on the D_{KL} value would be implemented in order to generate a new graph. In principle finer tuning of the network topology could be performed by applying the same divergence metric to the cluster distribution of the network. The cluster distribution of a network is created by plotting the value of the cluster coefficient against the number of nodes with that cluster coefficient value. This method could be used in combination with dynamic programming in order to continually monitor, and guide network growth.

8.6.1 A practical note on the time and resources required by simulation experimentation

As described above, the experimental results documented in this thesis were generated using computer simulation. The simulation calculation is very computationally intensive, and for the system interrogated here with 200 nodes, and the values of the other parameters as described, each simulation run took 2-3 days of CPU time to complete (so a typical results graph would take 10 - 15 days to generate), running on a beowulf cluster using linux as the operating system. Due to the stochastic nature of ACO, each point in the configuration space needed to be sampled at least three times (preferably five times). Given the number of points in the configuration space, this means that the generation of the results in the chapter on equilibrium, and non-equilibrium behaviours required processing, and validating and (in the case of the KL distance) post-processing data from approximately 3,000 experiments. This amounted to roughly three months of CPU time to complete the simulation experiments, and additional post-production analysis time. The time required to perform the simulation experiments is the reason for not performing further experimentation to try to improve the simulation results documented in this thesis. The nature of the simulations is such that they slow down as the simulation proceeds. There are two reasons for this. Firstly, the number of ants in the system increases as the simulation proceeds. Secondly, the routing tables in the nodes grow, as the list of possible destination nodes grows due to the feedback of the ants in the system. The growth of the routing tables as the simulation proceedes increases the time taken for the routing calculation to be performed at each node, as more and more information needs to be processed. This means that the simulation proceedes it slows. Performing similar simulations to investigate the scaling behaviour on such an ant system would require even more time than required in the first tranche of experiments. This is because the number of possible destinations in a larger system is greater, there are more ants, and also the routing tables in the nodes will grow at a greater rate in a larger system due to the feedback of routing information generated by the ants. In general the ant routing algorithm scales as $O(n^3)$, where n is the number of nodes in the ant system.

Appendix A

Appendix

A.1 Purpose

In this appendix we detail the file formats and the meanings of the fields in both the input and output files of both release 1 and release 2 of the simulator described in Chapter 4.

A.2 Simulator release 1 input files

In this section we detail the input files for the release 1 of the simulator.

A.2.1 Configuration file for release 1 of the simulator

Field name	Field format	Meaning of field
Number of nodes	Integer	Total number of nodes in the system
Grid X size	Integer	The width of the grid upon which the simulation will be run
Grid Y size	Integer	The height of the grid upon which the simulation will be run
Number of iterations	Integer	Number of iterations in the simulation
Number of iterations between snapshots	Integer	Number of iterations between snapshots.

continued on next page

Table A.1: Definition of simulator release 1 configuration file

Field name	Field format	Meaning of field
Percentage of nodes that can send ants	Float	Percentage of nodes that can initiate ants in the system. All nodes can route ants in the system irrespective of whether they are able to send them or not.
Pheromone persistence rate	Float	This ranges between 0 and 1. If it is 1 then there is no decay as 100 percent of the pheromone is retained in the routing tables of the nodes.
Pheromone increment rate	Float	This indicates the amount of pheromone that is deposited in the routing table of the node when an ant returns safely to its source.
Range	Integer	This is the initial range of awareness of the nodes used when we first compute which nodes can see which others.
Threshold	Float	This controls the pruning mechanism for the routing table. If the threshold is ten percent then every time a snapshot is taken the bottom ten percent (by pheromone value) of routes in the routing table are removed from the routing table.
Initial pheromone value	Float	This is just the initial pheromone value placed in the routing tables when the simulation is initialised. This value is only put in the routes generated when the nodes first compute which other nodes they can see at the beginning of the simulation
Results path	String	This is just the path indicating where the files to be saved will be kept.
Queue lengths	Integer	The maximum queue lengths of the nodes in question.
Batch size	Integer	This is the number of ants processed by each node on each iteration.

Table A.1: Definition of simulator release 1 configuration file

A.2.2 Node information file for release 1 of the simulator

Field name	Field format	Meaning of field
X-coordinate of node	Integer	Self-explanatory
Y-coordinate of node	Integer	Self-explanatory
Send ant capability	Boolean	Set to true if the node in question is capable of sending ants. False otherwise. Note all nodes can route ants unless congested.

Table A.2: Definition of configuration file for constant demand

A.3 Simulator release 1 output files

In this section we detail the output files of the release 1 of the simulator.

A.3.1 Degree distribution file for release 1

Field name	Field format	Meaning of field
Degree	Integer Array	Total number of connections to a given node
Number of nodes with given degree	Integer Array	The number of nodes of the population in the peer-to-peer system which have the degree specified in the degree column
Total number of nodes in the simulation	Integer	Self-explanatory
Range	Integer	The range of awareness of the nodes in the simulation.
Pheromone persistence rate	Float	The pheromone persistence rate used in the calculations in the simulation

continued on next page

Table A.3: Definition of degree distribution output file

Percentage of nodes that can send ants	Integer	This is self-explanatory.

Table A.3: Definition of degree distribution output file

A.3.2 Path length distribution file for release 1

Field name	Field format	Meaning of field
Path length	Integer	Path length of a tour
Number of tours with the given path length	Integer Array	Number of ants which have made tours with the given path length.
Total number of nodes in the simulation	Integer	Self-explanatory
Range	Integer	The range of awareness of the nodes in the simulation
Pheromone persistence rate	Float	The pheromone persistence rate used in the calculations in the simulation
Percentage of nodes that can send ants	Integer	This is self-explanatory.

Table A.4: Definition of output file for path length distribution

A.3.3 Key statistics file for release 1

Field name	Field format	Meaning of field
Total ants in the system	Integer	Self-explanatory
Total number of ants dropped	Integer	Number of ants which have been dropped by the system because the nodes they were passing through were congested and so could not accept further ants
Total number ants home	Integer	The total number of ants that have made it home to the source node
Total number of nodes	Integer	Self-explanatory
Range	Integer	The range of awareness of the nodes in the simulation
Pheromone persistence rate	Float	The pheromone persistence rate used in the calculations in the simulation
Percentage of nodes that can send ants	Integer	This is self-explanatory
Average degree	Float	The average degree of all the nodes in the system to date
Average path length	Float	The average path length of the tours that the ants have made during the course of the simulation.

Table A.5: Definition of format of key statistics file

A.4 Simulator release 2 input files

In this section we detail the input files of the release 2 of the simulator.

A.4.1 Configuration file for constant demand

Field name	Field format	Meaning of field
Number of nodes	Integer	Self-explanatory

continued on next page

Table A.6: Definition of configuration file for constant demand

Grid X size	Integer	The width of the grid upon which the simulation will be run.
Grid Y size	Integer	The height of the grid upon which the simulation will be run.
Number of iterations	Integer	Self-explanatory
Number of iterations between snapshots	Integer	Self-explanatory
Percentage of nodes that can send ants	Float	Percentage of nodes that can initiate ants in the system. All nodes can route ants in the system irrespective of whether they are able to send them or not.
Pheromone persistence rate	Float	This ranges between 0 and 1. If it is 1 then there is no decay as 100 percent of the pheromone is retained in the routing tables of the nodes.
Pheromone increment rate	Float	This indicates the amount of pheromone that is deposited in the routing table of the nodes when an ant returns safely to its source.
Range	Integer	This is the initial range of awareness of the nodes used when we first compute which nodes can see which others.
Threshold	Float	This controls the pruning mechanism for the routing table. If the threshold is ten percent then every time a snapshot is taken the bottom ten percent (by pheromone value) of routes in the routing table are removed from the routing table.
Initial pheromone value	Float	This is just the initial pheromone value placed in the routing tables when the simulation is initialised. This value is only put in the routes generated when the nodes first compute which other nodes they can see at the beginning of the simulation.

continued on next page

Table A.6: Definition of configuration file for constant demand

Results path	String	This is just the path indicating where the files to be saved will be kept.
Queue lengths	Integer	The maximum queue lengths of the nodes in question.
Demand mode	Integer	This indicates the demand mode of the simulation. If the demand mode is -1 then we are in a constant demand simulation.
Demand multiplier	Float	This indicates the number of ants that will be sent out each iteration as described in the chapter on busty demand.
Probability of node failure	Float	This is the probability that the node makes a transition from the operational state to the non-operational state
Probability of node resurrection	Float	This is the probability that the node makes a transition from the non-operational state to the operational state
Prune queue mode flag	Integer	When this flag is set to -1 only the head ant is cleared from the node when the node enters the non-operational state. If the flag is zero then the entire queue is deleted on entering the non-operational state.
Batch size	Integer	This is the number of ants processed by each node on each iteration.

Table A.6: Definition of configuration file for constant demand

A.4.2 Configuration file for bursty demand

Field name	Field format	Meaning of field
Number of nodes	Integer	Total number of nodes in the system.
Grid X size	Integer	The width of the grid upon which the simulation will be run.
Grid Y size	Integer	The height of the grid upon which the simulation will be run.

continued on next page

Table A.7: Definition of configuration file for bursty demand

Number of iterations	Integer	Number of iterations in the simulation.
Number of iterations between snapshots	Integer	Number of iterations between snapshots.
Percentage of nodes that can send ants	Float	Percentage of nodes that can initiate ants in the system, all nodes can route ants in the system irrespective of whether they are able to send them or not.
Pheromone persistence rate	Float	This ranges between 0 and 1. If it is 1 then there is no decay as 100 percent of the pheromone is retained in the routing tables of the nodes.
Pheromone increment rate	Float	This indicates the amount of pheromone that is deposited in the routing table of the nodes when an ant returns safely to its source.
Range	Integer	This is the initial range of awareness of the nodes used when we first compute which nodes can see which others.
Threshold	Float	This controls the pruning mechanism for the routing table. If the threshold is ten percent then every time a snapshot is taken the bottom ten percent (by pheromone value) of routes in the routing table are removed from the routing table.
Initial pheromone value	Float	This is just the initial pheromone value placed in the routing tables when the simulation is initialised. This value is only put in the routes describing generated when the nodes first compute which other nodes they can see at the beginning of the simulation.
Results path	String	This is just the path indicating whether the files to be saved will be kept.
Queue lengths	Integer	The maximum queue lengths of the nodes in question.

continued on next page

Table A.7: Definition of configuration file for bursty demand

Demand mode	Integer	This indicates the demand mode of the simulation. If the demand mode is 1 then we are in a bursty demand simulation.
Path to demand information	String	This indicates the path to the file containing the successive values for the demand multiplier to be used in the simulation.
Number of demand measurements	Integer	This is the number of demand measurements in the demand information file.
Probability of node failure	Float	This is the probability that the node makes a transition from the operational state to the non-operational state.
Probability of node resurrection	Float	This is the probability that the node makes a transition from the non-operational state to the operational state.
Prune queue mode flag	Integer	When this flag is set to -1 only the head ant is cleared from the node when the node enters the non-operational state. If the flag is zero then the entire queue is deleted on entering the non-operational state.
Batch size	Integer	This is the number of ants processed by each node on each iteration.

Table A.7: Definition of configuration file for bursty demand

A.5 Simulator release 2 output files

In this section we detail the output files for the release 2 of the simulator.

A.5.1 Degree distribution file for release 2

Field name	Field format	Meaning of field
Degree	Integer Array	Total number of connections to a given node.

continued on next page

Table A.8: Definition of degree distribution output file

Number of nodes with given degree	Integer	The number of nodes of the population in the peer-to-peer system which have the degree specified in the degree column
Total number of nodes in the simulation	Integer	Self-explanatory
Range	Integer	The range of awareness of the nodes in the simulation.
Pheromone persistence rate	Float	The pheromone persistence rate used in the calculations in the simulation.
Percentage of nodes that can send ants	Integer	This is self-explanatory.
Number of healthy nodes	Integer	The number of nodes in the peer-to-peer system that are in the operational state.

Table A.8: Definition of degree distribution output file

A.5.2 Path length distribution file for release 2

Field name	Field format	Meaning of field
Path length	Integer	Path length of a tour
Number of tors with the given path length	Integer	Number of ants which have made tours with the given path length
Total number of nodes in the simulation	Integer	The total number of nodes in the simulation

continued on next page

Table A.9: Definition of output file for path length distribution

Field name	Field format	Meaning of field
Range	Integer	The range of awareness of the nodes in the simulation
Pheromone persistence rate	Float	The pheromone persistence rate used in the calculations in the simulation
Percentage of nodes that can send ants	Integer	This is self-explanatory
Number of healthy nodes	Integer	The number of nodes in the peer-to-peer system that are in the operational state

Table A.9: Definition of output file for path length distribution

A.5.3 Key statistics file for release 2

Field name	Field format	Meaning of field
Total ants in the system	Integer	Self-explanatory
Total number of ants dropped	Integer	Number of ants which have been dropped by the system either because the nodes they were passing through became non-operational or the nodes were congested and so could not accept further ants
Total number ants home	Integer	The total number of ants that have made it home to the source node
Number of healthy nodes	Integer	The total number of nodes in the operational state
Total number of nodes	Integer	The total number of nodes in the simulation
Range	Integer	The range of awareness of the nodes in the simulation

continued on next page

Table A.10: Definition of format of key statistics file

Pheromone persistence rate	Float	The pheromone persistence rate used in the calculations in the simulation
Percentage of nodes that can send ants	Integer	This is self-explanatory
Number of healthy nodes	Integer	The number of nodes in the peer-to-peer system that are in the operational state
Average degree	Float	The average degree of all the operational nodes in the system to date
Average path length	Float	The average path length of the tours that the ants have made during the course of the simulation

Table A.10: Definition of format of key statistics filc

Appendix B

Appendix

B.1 Purpose

In this appendix we detail the results of the validation tests we have performed between release 1 and release 2 of the simulator, by documenting the results of the simulators under a range of starting conditions. All validation simulations ran for 3000 iterations, with a standard test configuration of 200 simulated nodes, on a grid which is 100 by 100.

B.2 Validation results

In this section we detail the validation results of the simulator.

Number of nodes	Pheromone persistence rate	Initial range of awareness	Percentage of nodes able to send ants	Average path length	Average degree distribution	Simulator release
200	0.96	60	0.6	2.02	2.43	1
200	0.98	60	0.6	1.97	2.21	1
200	0.92	60	0.6	2.23	2.17	1
200	0.96	90	0.6	2.31	3.21	1
200	0.98	90	0.6	2.05	3.17	1
200	0.92	90	0.6	1.87	3.1	1
200	0.96	60	0.6	2.12	2.47	2
200	0.98	60	0.6	1.89	2.17	2
200	0.92	60	0.6	2.27	2.12	2

continued on next page

Table B.1: Simulation validation results

Number of nodes	Pheromone persistence rate	Initial range of awareness	Percentage of nodes able to send ants	Average path length	Average degree distribution	Simulator release
200	0.96	90	0.6	2.41	3.26	2
200	0.98	90	0.6	2.09	3.21	2
200	0.92	90	0.6	2.03	3.27	2

Table B.1: Simulation validation results

Bibliography

[1] R. Albert, A. Barabasi, and H. Jeong. Mean-field theory for scale-free random networks. *Physica A*, 272:173–187, 1999.

[2] A. Barabasi and R.Albert. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74, January 2002.

[3] A.-L. Barabasi, R. Albert, and H. Jeong. Scale-free characteristics of random networks: The topology of the world wide web. *Physica A*, 281:69–77, 2000.

[4] B Baran. Improved antnet routing. *ACM SIGCOMM*, 31(2), april 2001.

[5] Ranjita Bhagwan, Kiran Tati, Yu-Chung Cheng, Stefan Savage, and Geoff M. Voelker. Total recall: System support for automated availability management. In *Proceedings of Symposium on Networked Systems Design and Implementation (NSDI)*, march 2004.

[6] Ginestra Bianconi and Albert Barabasi. Bose einstien condensation in complex networks. *Physical Review Letters*, pages 5632–5635, June 2001.

[7] M.L Boas. *Mathematical Methods in the Physical Sciences.* John Wiley, 2004.

[8] E. Bonabeau, F. Henaux, S. Guérin, D. Snyers, P. Kuntz, and G. Theraulaz. Routing in telecommunications networks with "smart" ant-like agents. In *Intelligent Agents for Telecommunications Applications '98 (IATA'98)*, 1998.

[9] Geoffrey Canright. Ants and loops. In *ANTS '02: Proceedings of the Third International Workshop on Ant Algorithms*, pages 235–242, London, UK, 2002. Springer-Verlag.

[10] G. Di Caro and M. Dorigo. AntNet: a mobile agents approach to adaptive routing. Technical Report IRIDIA/97-12, Université Libre de Bruxelles, Belgium, 1997.

[11] Gianni Di Caro and Marco Dorigo. Ant colonies for adaptive routing in packet-switched communications networks. In *PPSN V: Proceedings of the 5th International Conference on parallel Problem Solving from Nature*, pages 673–682, London, UK, 1998. Springer-Verlag.

[12] J. Chu, K. Labonte, and B.N. Levine. Availability and locality measurements of peer-to-peer file systems. In Victor Firoiu; Zhi-Li Zhang; Eds, editor, *Proc. SPIE, Scalability and Traffic Control in IP Networks II*, volume 4868, pages 310–321, jul 2002.

[13] Samir R. Das, Robert Casta, and Jiangtao Yan. Simulation-based performance evaluation of routing protocols for mobile ad hoc networks. *Mob. Netw. Appl.*, 5(3):179–189, 2000.

[14] A Datta and K.Aberer. Internet-scale storage systems under churn - A steady state analysis. Technical report, Ecole Polytechnique Federale de Lausanne, 2005.

[15] Gianni Di Caro, Frederick Ducatelle, and Luca Maria Gambardella. Anthocnet: an ant-based hybrid routing algorithm for mobile ad hoc networks. In *In Proceedings of PPSN VIII - Eight International Conference on Parallel Problem Solving from Nature*, number 3242 in Lecture Notes in Computer Science, pages 461–470, Birmingham, UK, sep 2004. Springer-Verlag. Best paper award.

[16] Shigeo Doi and Masayuki Yamamura. An experimental analysis of loop-free algorithms for scale-free networks. In *Ant Colony Optimization and swarm intelligence,4th international workshop*, volume 3172, pages 278–285. Springer, january 2004.

[17] Yixin Dong. Supernode based reverse labeling algorithm:QoS support on mobile ad hoc networks. In *Proceedings of the 2002 IEEE Canadian Conference on Electrical and Computer Engineering*, 2002.

[18] M. Dorigo and T. Stützle. *Ant Colony Optimization.* MIT Press, Cambridge, MA, 2004.

[19] S.N Dorogovtsev and J.F.F.Mendes. Evolution of networks. *Journal of Condensed Matter Physics*, March 2001.

[20] S.N Dorogovtsev and J.F.F.Mendes. *Evolution of Networks.* Oxford university Press, 2003.

[21] E.Gelenbe, R.Lent, and J.Nunez. Self-aware networks and QoS. In *Proceedings of the IEEE*, volume 92, pages 1478–1489, september 2004.

[22] M. Feng. A self-healing routing scheme based on AODV in ad hoc networks. In *CIT '04 Proceedings of the fourth international conference on Computer and Information Technology*, 2004.

[23] E. Gelenbe and P. Liu. QoS and routing in the cognitive packet network. In *Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, June 2005.

[24] E. Gelenbe, P. Liu, and J. Laine. Genetic algorithms for autonomic route discovery. In *Proceedings of IEEE 2006 workshop on distributed intelligent systems*, June 2006.

[25] Erol Gelenbe, Ricardo Lent, Alfonso Montuori, and Zhiguang Xu. Cognitive packet networks: QoS and performance. In *MASCOTS*, pages 3–, 2002.

[26] Erol Gelenbe, Ricardo Lent, and Zhiguang Xu. Networks with cognitive packets. In *MASCOTS*, pages 3–, 2000.

[27] Erol Gelenbe, Ricardo Lent, and Zhiguang Xu. Measurement and performance of a cognitive packet network. *Comput. Networks*, 37(6):691–701, 2001.

[28] A Haar. Zur theorie der orthogonalen funktionensysteme. *Math Ann*, 69:331–371, 1910.

[29] Poul E. Heegaard, Otto Wittner, and Bjarne Helvik. Self-managed virtual path management in dynamic networks. In Ozalp Babaoglu, Márk Jelasity, Alberto Montresor, Christof Fetzer, Stefano Leonardi, Aad van Moorsel, and Maarten van Steen, editors, *Self-Star Properties in Complex Information Systems*, volume 3460 of *Lecture Notes in Computer Science*. Springer-Verlag, 2005.

[30] H.Jeong, Z.Neda, and A.Barabasi. Measuring preferential attachment in evolving networks. *Europhys Letters*, 6:567–572, 2003.

[31] Ying-Tung Hsiao, Cheng-Long Chuang, and Cheng-Chih Chien. Computer network load balancing and routing by ant colony optimization. In *Proceedings of 12 IEEE international conference Networks*, volume 1, pages 313–318, November 2004.

[32] H. Hsieh and R. Sivakumar. On using the ad-hoc network model in cellular packet data networks. In *Proceedings of the third ACM international symposium on mobile ad hoc networking & computing*, pages 36–47. ACM Press, 2002.

[33] I.Farkas, I.Derenyi, H.Jeong, Z.Neda, Z.N.Oltvai, E.Ravasz, A.Schubert, A.Barabasi, and T.Vicsek. Networks in life:scaling properties and eigenvalue spectra. *Physica A*, 314:25–34, 2002.

[34] I.Kassabaldis, M.El-Sharkawi, R.Marks, P.Arabshahi, and A.Gray. Swarm intelligence for routing in communication networks. *IEEE Globecom*, November 2001.

[35] Raj Jain. *The art of computer systems performance analysis*. Wiley, first edition, 1991.

[36] W.E Leland, M.Taqqu, W.Willinger, and D.V.Wilson. On the self-similar nature of ethernet traffic. *IEEE/ACM transactions*, 2:1–15, February 1994.

[37] Derek Leonard, Vivek Rai, and Dmitri Loguinov. On lifetime-based node failure and stochastic resilience of decentralized peer-to-peer networks. *SIGMETRICS Perform. Eval. Rev.*, 33(1):26–37, 2005.

[38] Xuefei Li and Laurie Cuthbert. Multipath QoS routing of supporting diffserv in mobile ad hoc networks. In *SNPD-SAWN '05: Proceedings of the Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks (SNPD/SAWN'05)*, pages 308–313, Washington, DC, USA, 2005. IEEE Computer Society.

[39] Hsien-Chou Liao, Yi-Wei Ting, Shih-Hsuan Yen, and Chou-Chen Yang. Ant mobility model platform for network simulator. In *ITCC '04: Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'04) Volume 2*, page 380, Washington, DC, USA, 2004. IEEE Computer Society.

[40] Prakash Linga, Indranil Gupta, and Ken Birman. A churn-resistant peer-to-peer web caching system. In *SSRS '03: Proceedings of the 2003 ACM workshop on Survivable and self-regenerative systems*, pages 1–10, New York, NY, USA, 2003. ACM Press.

[41] Jason Liu, Yougu Yuan, David M. Nicol, Robert S. Gray, Calvin C. Newport, David Kotz, and Luiz Felipe Perrone. Simulation validation using direct execution of wireless ad-hoc routing protocols. In *PADS '04: Proceedings of the eighteenth workshop on Parallel and distributed simulation*, pages 7–16, New York, NY, USA, 2004. ACM Press.

[42] Stephane Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, second edition, 1999.

[43] Vittorio Maniezzo, Marco Boschetti, and Márk Jelasity. An ant approach to membership overlay design. In *ANTS Workshop*, pages 37–48, 2004.

[44] Daniel Merkle and Martin Middendorf. Modeling the dynamics of ant colony optimisation. *Evolutionary Computation*, 10:235–262, 2002.

[45] Nicolas Meuleau and Marco Dorigo. Ant colony optimization and stochastic gradient descent. *Artif. Life*, 8(2):103–121, 2002.

[46] Alberto Montresor and Ozalp Babaoglu. The BISON project. *IEEE Computational Intelligence Bulletin*, 1(1):6–9, dec 2002.

[47] Alberto Montresor and Ozalp Baboglu. Biology-inspired approaches to peer-to-peer computing in BISON. In *Proceedings of the 3rd International Conference on Intelligent Sys-*

tem Design and Applications (ISDA'03), Advances in Soft Computing, pages 515–522. Springer-Verlag, aug 2003.

[48] L. Muscariello, M. Mellia, M. Meo, and M. Ajmone Marsan. An MMPP-based hierarchical model of internet traffic. In *ICC 2004 — IEEE International Conference on Communications Gate to the global information society*, june 2004.

[49] Anders Mykkeltveit, Poul Heegaard, and Otto Wittner. Realization of a distributed route management system on software routers. In *Proceedings of Norsk Informatikkonferanse*, Stavanger, Norway, nov 2004.

[50] Valeri Naoumov and Thomas Gross. Simulation of large ad hoc networks. In *MSWIM '03: Proceedings of the 6th ACM international workshop on modeling analysis and simulation of wireless and mobile systems*, pages 50–57, New York, NY, USA, 2003. ACM Press.

[51] M J Newman. Models of the small world. *Journal of Condensed Matter Physics*, 101:819–841, May 2000.

[52] T.H Noh. End-to-end self-healing SDH/ATM networks. In *GLOBECOM '96. 'Communications: The Key to Global Prosperity*, volume 3, pages 1877–1881, Nov 1996.

[53] Abry P., Richard Baraniuk, Patrick Flandrin, Rudolf Riedi, and Darryl Veitch. The multiscale nature of network traffic discovery, analysis, and modelling. *IEEE Signal processing magazine*, 19(3):28–46, May 2002.

[54] Kihong Park, Gitae Kim, and Mark Crovella. On the relationship between file sizes transport protocols and self similar network traffic. In *Proceedings of the international conference on Network Protocols*, pages 171–180, October 1996.

[55] Himabindu Pucha, Saumitra M. Das, and Y. Charlie Hu. The performance impact of traffic patterns on routing protocols in mobile ad hoc networks. In *MSWiM '04: Proceedings of the 7th ACM international symposium on modeling, analysis and simulation of wireless and mobile systems*, pages 211–219, New York, NY, USA, 2004. ACM Press.

[56] Sean Rhea, Dennis Geels, Timothy Roscoe, and John Kubiatowicz. Handling Churn in a DHT. In *Proceedings of the 2004 USENIX Technical Conference, Boston, MA, USA*, June 2004.

[57] Vinay J. Ribeiro, Rudolf H. Riedi, Matthew S. Crouse, and Richard G. Baraniuk. Multiscale queuing analysis of long-range-dependent network traffic. *INFOCOM (2)*, pages 1026–1035, 2000.

[58] Rudolf Riedi, Matthew S Crouse, Vinay Ribeiro, and Richard Baraniuk. A multifractal wavelet model with application to network traffic. In *IEEE Transactions on Information Theory*, volume 45, pages 992 – 1018, april 1999.

[59] S. Robert and Jean-Yves Le Boudec. A markov modulated process for self-similar traffic. *Internationales Begegnungs und Forschungszentrum fuer Informatki, Schloss Dagsthul, Saarbr ucken*, pages 1–14, September 1995. http://infoscience.epfl.ch/search.py?recid=9.

[60] Ruud Schoonderwoerd, Owen E. Holland, Janet L. Bruten, and Leon J. M. Rothkrantz. Ant-based load balancing in telecommunications networks. *Adaptive Behavior*, pages 169–207, 1996.

[61] L. Singh and G. Dattatreya. A novel approach to parameter estimation in markov-modulated poisson processes. In *IEEE emerging technologies conference*, October 2004.

[62] R. Sterritt and D. Bustard. Autonomic computing—a means of achieving dependability. In *Proceedings of IEEE International Conference on the Engineering of Computer Based Systems (ECBS'03)*, pages 247–251, April 2003.

[63] Roy Sterritt. Autonomic networks: Engineering the self-healing property. *Engineering Applications of Artificial Intelligence*, 17:727–739, October 2004.

[64] Roy Sterritt. Autonomic networks: engineering the self-healing property. *Innovations in Systems and Software Engineering*, 1(1):79 – 88, April 2005.

[65] Roy Sterritt, David W. Bustard, Darren Gunning, and Phillip Henning. Autonomic communications and the reflex unified fault management architecture. *Advanced Engineering Informatics*, 19(3):189–198, 2005.

[66] Daniel Stutzbach and Reza Rejaie. Understanding churn in peer-to-peer networks. Technical report, University of Oregon, 2005.

[67] Thomas Stützle and Marco Dorigo. A short convergence proof for a class of ant colony optimization algorithms. *IEEE Trans. Evolutionary Computation*, 6(4):358–365, 2002.

[68] T. Tassier and F. Menczer. Emerging small-world referral networks in evolutionary labor markets. *IEEE-EC*, 5:482–492, Oct 2001.

[69] Chris Tofts. The autosynchronisation of lepothorax acervorum described in wsccs. Technical Report ECS-LFCS-90-128, The kings buildings Edinburgh EH9 3JZ, 1990.

[70] Chris Tofts. Task allocation in monomorphic ant species. Technical Report ECS-LFCS-91-144, Laboratory for foundations of computer science Department of Computer Science University of Edinburgh, The kings buildings Edinburgh EH9 3JZ, 1991.

[71] K. Trivedi. *Probability and Statistics with Reliability, Queueing, and Computer Science Applications*. Prentice-Hall, 1982.

[72] Xin Wang, Shiram Sarvotham, Rudolf Riedi, and Richard Baraniuk. Connection-level modeling of network traffic. In *Dimacs Workshop on internet and www measurement, mapping and modelling*, 2002.

[73] Horst F. Wedde, Muddassar Farooq, Thorsten Pannenbaecker, Bjoern Vogel, Christian Mueller, Johannes Meth, and Rene Jeruschkat. Beeadhoc: an energy efficient routing algorithm for mobile ad hoc networks inspired by bee behavior. In *GECCO '05: Proceedings of the 2005 conference on genetic and evolutionary computation*, pages 153–160, New York, NY, USA, 2005. ACM Press.

[74] W. Willinger, V. Paxson, and M. S. Taqqu. *Self-similarity and Heavy Tails: Structural Modeling of Network Traffic*. Boston press, 1998. citeseer.ist.psu.edu/86873.html.

[75] Walter Willinger, Murad Taqqu, and Ashok Erramilli. *Stochastic Networks:Theory and Applications*. OUP, 1996. A Bibliographical Guide to Self-Similar Traffic and Performance Modeling for Modern High Speed Networks.

[76] H. Zhang, A. Goel, and R. Govindan. Using the small-world model to improve freenet performance. *Proc IEEE Infocom*, 2002.

[77] Saida Ziane and Abdelhamid Melouk. A swarm intelligent multi-path routing for multimedia traffic over mobile ad hoc networks. In *Q2SWinet '05: Proceedings of the 1st ACM international workshop on quality of service & security in wireless and mobile networks*, pages 55–62, New York, NY, USA, 2005. ACM Press.