Agent-Based Resource Management in IP Networks

Karen Victoria Shoop

Submitted for the degree of Doctor of Philosophy

Department of Electronic Engineering
Queen Mary, University of London

Abstract

The growth in traffic across IP networks has been mirrored by a demand for higher quality service provision. As the generic IP best-effort paradigm is no longer suitable given the diversity of customer and application requirements, there is a need to provide Quality of Service (QoS) across multi-class networks. Such treatment must not only satisfy the requirements demanded of high-grade traffic but also ensure that best-effort traffic receives an appropriate level of service.

This thesis investigates the applicability of agent technology in multi-class connectionless networks. An analysis of agents in telecommunication networks is undertaken, questioning whether all work that claims to employ agents is indeed doing so. Likewise the thesis explores whether a body of network research could be described as agent-based despite not declaring such entities. The ramifications of such inconsistencies are discussed to highlight whether indeed intelligent software agents are well placed to provide the sophistication necessary for QoS provision in a distributed and dynamic environment. Furthermore, in a tightly coupled environment the autonomy associated with agents is constrained. Connectionless networks rely on a set of related next hops to route traffic along least cost paths; employing agent intelligence at each node may lead to inconsistencies. This research argues that while deploying agent technology may be inappropriate at the IP level, nevertheless techniques associated with an agent approach provide important enhancements to routing.

This thesis introduces a novel “sub-optimal” adaptation to the OSPF routing protocol, based on masqueraded cost metrics and allowing for proactive routing, in anticipation of congestion. Fuzzy reinforcement learning is then introduced to add further responsiveness to the system. Finally this is located within the development of agents in networks.

Acknowledgements

This project was largely supported with funding from Nokia Corporation. Many thanks are especially due to Andreas (Andrepeter) Heiner, for his helpful advice and endless patience.

Additionally I would like to acknowledge the EPSRC for funding the initial work on this project.

It is perhaps impossible to know where to begin with thanking my supervisors, John Bigham and Chris Phillips. They have encouraged, supported and cajoled me through this work: through various dead ends, simulation disasters and self-doubt. I would like to salute them as great mentors and very firm friends.

Also many thanks are due to Ho, Michele, Lynda and Laurie in the department for their support. Laurissa is due admiration for her expertise in reinforcement learning, shared over many lunches. Additionally all those who have formed room 356: Costas, Janny, Jim, Landong, Leonid, and most of all Damian for believing I could do this. Plus those who have very patiently put up with a friend trying to finish a PhD: Livy, Aimee, Jules, Nick, Fiona, Rob, Clare, Joe, Freddie, Theo and Albert.

Many thanks to my family for their encouragement: Sandra, Stanley (alias Mum and Dad), Tanya, Vik and Fiona

Most of all, special thanks to Mark – without whom all this would be pointless – for all his love, confidence and support.

And lastly to Zed, whose arrival midway through the PhD has led to some interesting challenges. Finally I shall be able to answer his call to “stop doing your PhD Mummy”.

Table of Contents

List of Figures

List of Tables

Glossary

ACK	Acknowledgement (TCP)
AF	Assured Forwarding
AI	Artificial Intelligence
AQM	Active Queue Management
AS	Autonomous System
ASP	Application Service Provider
ATM	Asynchronous Transfer Mode
BA	Behaviour Aggregate
BB	Bandwidth Broker
BE	Best-Effort
BFD	Bidirectional Forwarding Detection Protocol
BGP	Border Gateway Protocol
COA	Care Of Address
COPS	Common Open Policy Service Protocol
CPU	Central Processing Unit
CR-LDP	Constraint-Based Routed Label Distribution Protocol
DAI	Distributed Artificial Intelligence
DPS	Dynamic Packet State
DiffServ	Differentiated Services
EF	Expedited Forwarding
FEC	Forward Equivalence Class
FIPA	Foundation for Intelligent Physical Agents
FQ	Fuzzy Q strength
FTP	File Transfer Protocol
IETF	Internet Engineering Task Force
IGP	Interior Gateway Protocol
IN	Intelligent Network
IntServ	Integrated Services
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6

IS-IS	Intermediate System to Intermediate System
ISDN	Integrated Services Digital Network
ISP	Internet Service Provider
ITU	International Telecommunications Union
ITU-T	International Telecommunications Union – Telecommunications Standardisation Sector
LDP	Label Distribution Protocol
LP	Logical Path
LSA	Link State Advertisement
LSP	Label Switched Path
LSR	Label Switched Router
LSU	Link State Update
MAN	Metropolitan Area Network
MIB	Management Information Base
MPLS	MultiProtocol Label Switching
NSP	Network Service Provider
OO	Object-Oriented
OSI	Open Systems Interconnection
OSPF	Open Shortest Path First
OSPF-TE	Open Shortest Path First Protocol with Traffic Engineering Extensions
PEP	Performance Enhancing Proxy
PNNI	Private Network-Network Interface
PS	Policy Server
PSTN	Public Switched Telephone Network
QoS	Quality of Service
QOSPF	Quality of Service-based Open Shortest Path First
RED	Random Early Detection
RFC	Request for Comment
RIP	Routing Information Protocol
RNG	Random Number Generator
RSVP	Resource ReSerVation Protocol
RSVP-TE	Resource ReSerVation Protocol with Traffic Engineering Extensions
Sarsa	State, Action, Reward, State, Action

SCP	Service Control Point
SCORE	Stateless Core
SLA	Service Level Agreement
SLS	Service Level Specification
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
spf	shortest path first
TCP	Transmission Control Protocol
TE	Traffic Engineering
TED	Traffic Engineering Database
TD	Temporal Difference
TLV	Type/Length/Value Structure
UA	User Agent
UDP	User Datagram Protocol
VoIP	Voice over IP
WDM	Wavelength Division Multiplexing

1 Introduction

This section outlines the initial stimulus and the objectives for this research.

1.1 Motivation

The motivation underlying this research derives from the changing profile of IP[i] network users and applications. Data and voice network convergence – IP networks with both the public switched telephone networks (PSTNs) and integrated services digital networks (ISDN), leading to the growth of IP telephony[ii] – is the new paradigm. This convergence, together with the growth in exacting applications, such a video conferencing and distance learning, and an increasingly demanding user profile has led to a focus on how to manage network resources more efficiently. At the same time, alongside these novel profiles, the traditional best-effort traffic associated with IP networks has grown, for example due both to increased use of web applications and I/O heavy scientific applications [1]. Solutions must be sought to service both those users and applications that require higher quality treatment while also preserving the needs of best-effort customers and applications. It is notable that much work addressing these network challenges neglects the performance of best-effort traffic [2] despite no evidence that such traffic will cease to form the dominant traffic in such networks for the near future.

First it has to be established whether offering quality of service (QoS) is indeed a resource management issue. Although many papers refer to a dichotomy – those who advocate network dimensioning versus those who propose a managed QoS solution – little evaluation is provided to support the proponents of excess bandwidth. Since service differentiation – offering premium as well as best-effort and other traffic classes – results in higher overheads, an analysis must consider why this is considered an attractive or necessary option. Costs include increased network complexity, processing overhead, storage of reservation state; benefits include potentially increasing both network throughput and revenues.

[i] Networks that employ TCP/IP protocols. The role of the IP protocol is considered fundamental so TCP/IP networks/internets/internetworks are commonly called IP networks

[ii] referred to hereafter as Voice over IP (VoIP)

Since QoS appears to be a resource management issue, then intelligent agents would seem to afford increased functionality. Although their applicability to resource management has been demonstrated in connection-oriented networks, IP networks are characterised as connectionless. Furthermore, due to concerns about network security, a challenge was to investigate the use of static rather than mobile agents. However, the growth in agent telecommunications research has somewhat stalled. A prevailing explanation is that agents are a 'fad' – that the concept not just the abstraction is overused. There is a need for a thorough review of agent literature, to examine whether there has been and still is a role for agent technology or whether it is simply a label for a design metaphor and could simply be replaced by a more specific label in the context of the application e.g. Web Service in the context of business to business communication. This in turn requires an examination of a conflict between the notion of the agent, as a software engineering abstraction, and the concept of agent as embodied in network protocol literature.

1.2 Contribution

Much has been promised about the benefits of agents in telecommunications networks. It is perhaps surprising in light of such claims that there is comparatively little ongoing research and deployment in this area. As far as the author is aware, although there have been a few overviews of agents, for example [3], there has been no systematic analysis of the role of agents in networks, especially IP networks. A major contribution of this thesis is a review of agents in networks, culminating in a proposition explaining why the development may have been hindered. Additionally this analysis attempts to glean a possible role for agents in connectionless networks. While more has been written about agents in connection-oriented networks, the role of agents – specifically those that do not display mobility – in a connectionless network is rarely investigated. This thesis presents a role for agent – or agent-like – behaviour in such systems.

This thesis presents novel enhancements to the OSPF routing protocol that are sensitive both to the shifts in link costs as well as the trend in such costs. The initial

work presents a heuristic that spreads traffic away from optimal links. While appearing to contradict the goal of network optimisation, the proposal is that allowing low-class traffic to follow sub-optimal links increases network utilisation, thus increasing network optimisation. Agent intelligence is then employed to add further sensitivity. Recognising that adding intelligence to routers increases state, fuzzy logic is used as a means of inhibiting the dimensional growth associated with learning techniques.

1.3 Thesis Outline

Section 2 provides an overview of IP networks in order to establish why the provision of QoS across such networks is such an important research area. The subsequent section qualifies what is understood by QoS in networks. Having considered the varying definitions the two main 'schools' are addressed: section 3.1 examines the argument that no resource management is necessary – instead network over provisioning alone, by avoiding resource-contention, provides QoS; section 3.2 introduces resource management solutions.

Following this an introduction to the agent paradigm is presented, commencing with a presentation of the elusive nature of what constitutes an agent. From this section 4.3 questions whether the lability of this term in regard to its deployment in network protocols has undermined wider usage of this abstraction. Examples of agent applications in networks are provided. Additionally, similar practice that is not explicitly labelled agent-based is reviewed.

Section 5 presents an enhancement to IP routing, spreading non-premium traffic away from optimal paths. The purpose for this is to establish whether this is forms a beneficial strategy, before adding intelligence to the system. Section 6 introduces fuzzy reinforcement learning, providing an overview of both fuzzy control and reinforcement learning before delineating the novel application in IP networks. After evaluating results, the final sections establish the contribution to agent research.

[vii] In this thesis such computers are called routers, hosts or nodes

2 IP Networks

The growth of IP networks has been driven by the advantage garnered by decoupling services from the underlying hardware. Interconnection is via simple, connectionless protocols. This results in increased robustness due to reduced dependency between requester and receiver. In IP networks host computers connected to form subnets[vii]. Subnets in turn join other subnets to form the Internet. But, critically, compared to other networks QoS is explicitly omitted from network design.

The service provided across IP networks is characterised as a connectionless and unreliable system that offers best-effort packet delivery. The notion 'best-effort' implies that: admission is not denied to any traffic entering the network; all traffic is treated equally; traffic will be transmitted in the best possible way given available resources at any given time – artificial delays are neither generated nor unnecessary losses caused. A consequence of this is that there is no assurance of in-sequence delivery or indeed of packet arrival. Conceptually the reference model (ie the TCP/IP model) has five layers, as shown in Figure 1:

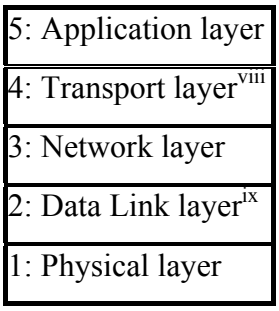


Figure 1: TCP/IP reference model

However, in practice the focus is placed on the three uppermost layers: the network layer responsible for connectionless packet routing and forwarding (defined by the Internet Protocol, IP); the transport layer responsible for effective transport service (either the reliable Transmission Control Protocol, TCP or the unreliable User

[viii] This is shortened from 'Host-to-Host Transport' layer
[ix] Some interpretations of the TCP/IP protocol suite have four layers and merge the data link and physical layers into one 'network interface/subnetwork/network access' layer

Datagram Protocol, UDP); the application layer responsible for application services (such as TELNET, FTP, SNMP). The IP protocol defines the basic transfer unit (packet), the datagram, across IP networks. Additionally it is responsible for packet routing, discussed in section 2.2.

2.1 The IP Datagram

If a more reliable service than best-effort is to be offered to some customers or to certain application traffic the routers have to be capable of distinguishing between the datagrams they receive. Enhancements to network protocols are necessarily conservative. Thus the means of differentiating packets should ideally be found in the IP datagram header, shown in Figure 2.

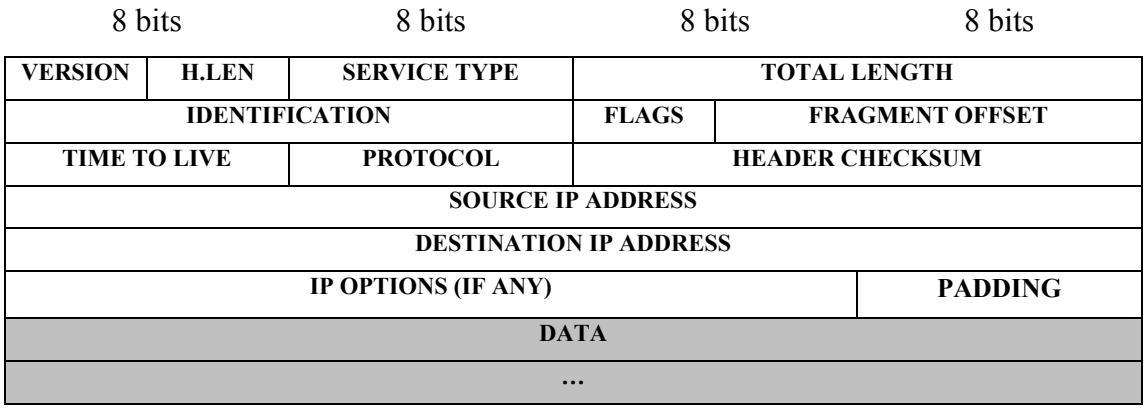


Figure 2: IPv4 Datagram

The preferred choice of field is the eight-bit SERVICE TYPE field, redefined by the IETF to provide for the Differentiated Services[x] codepoint (DSCP) [4], shown in Figure 3:

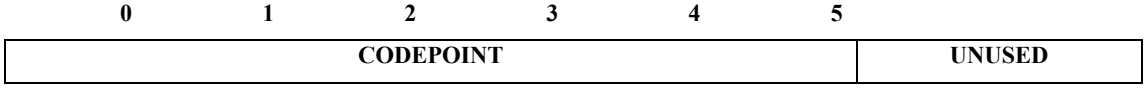


Figure 3: Service TypeField showing DSCP

[x] see section 0

This could theoretically identify 64 different levels of service, although in practice fewer classes would be utilised. Additionally for backward compatibility with previous subfield definition, the first three bits of the field (previously the precedence subfield) provide for eight classes of service. An alternative choice could be to use the IP OPTION field.

In the IPv6 protocol packet header there are two components that can support QoS via demarcating / differentiating service [5] [6]. The 8-bit TRAFFIC CLASS field corresponds to the differentiated services interpretation of the SERVICE TYPE in IPv4. Additionally the FLOW LABEL field was established for labelling packets belonging to certain traffic flows which require specific handling. Figure 4 shows the first line of the IPv6 datagram header:

VERSION	TRAFFIC CLASS	FLOW LABEL

Figure 4: First Line of IPv6 Header

2.2 OSPF

An Interior Gateway Protocol (IGP)[xi] is employed in IP networks to select the routers or paths through which traffic traverses. These routing protocols fall into two classes: those that employ a distance vector algorithm and those that employ a link state one. With the former neighbouring routers periodically share routing information; with the latter each router advertises to the network link state information (ie the state of each of its links) through a process called flooding (described below). This research uses OSPF, a well-tested, robust and widely deployed link state routing protocol [7], as the IGP. Other research work investigating QoS in the internet uses RIP – which employs the Bellman-Ford distance vector algorithm – as the IGP for example in order to use more than one QoS metric [8]. Another more formalised link state routing protocol, IS-IS [9], is also employed in some networks. However, OSPF is increasingly becoming the IGP of choice and, furthermore, it is the IETF recommended IGP. Enhancements that incorporate QoS into OSPF are discussed in section 3.2.4.

[xi] also known as intra-domain internet routing protocol

In each OSPF enabled router a topological database, known as the link state database, contains link details for the entire network or Autonomous System (AS). The topology is established through a neighbour discovery process at system setup[xii]. Each router runs the shortest path first (spf), also know as Dijkstra's, algorithm to calculate the shortest path from that router to every known destination in the AS [10]. This produces a shortest path tree, with that router as tree root. A routing table is then constructed to state the next hop (ie next router) for all destinations. If all the link state databases are not identical / synchronised the routing tables will be inconsistent and looping may arise. This is shown in Figure 5, where packets for destination C will be trapped in a loop between A to D and D to A till timeout.

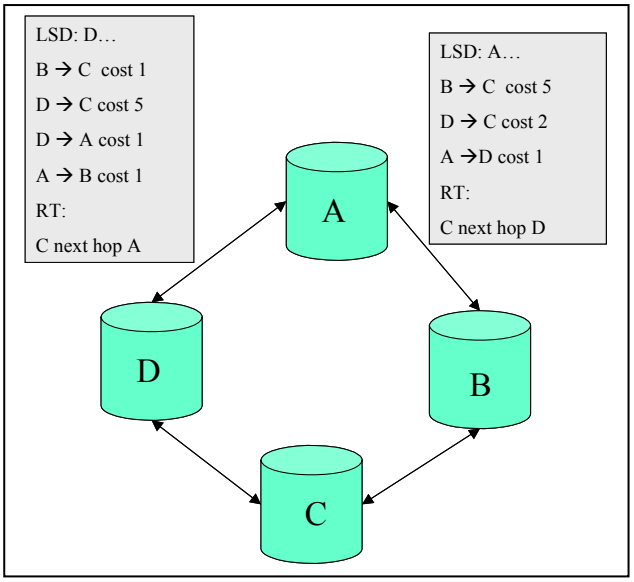


Figure 5: Looping due to inconsistent link state databases

OSPF specifies 'Hello' messages that are sent out regularly (the default setting of the HelloInterval is 10 seconds) between neighbours that act as keepalives. If a hello message is not received from a neighbour after a designated time, known as the RouterDeadInterval[xiii] the router sends out a Link State Advertisement (LSA) containing information about that link, encapsulated in a Link State Update (LSU)

[xii] The protocol specifies Database Description and Link State Request OSPF packets for database / topology discovery
[xiii] Cisco uses a default of 4 times the HelloInterval (ie 4x10 seconds) for the RouterDeadInterval

message. The neighbours, on receiving the LSU, extract the LSA, find the new link cost (in the 'metric' field), and update their link state database. The LSU is then retransmitted to all their neighbours. This forwarding process constitutes flooding. Routers discard LSAs/LSUs they have previously forwarded. This both limits the flooding mechanism and provides an implicit acknowledgement service (although OSPF also specifies an explicit Link State Acknowledgement). Once databases are updated, Dijkstra's algorithm is run again and an updated routing table constructed. Periodically (by default every 30 minutes, although Cisco now implements an OSPF LSA group pacing feature to stagger the refreshing [11]) every router floods an LSU packet containing details of all their connecting links. This flushing mechanism (the link state refresh) guards against, for example, corrupted link state databases and also acts as a keepalive. If there are no topological changes, OSPF is a quiet protocol, apart from the Hello messages and periodic updates.

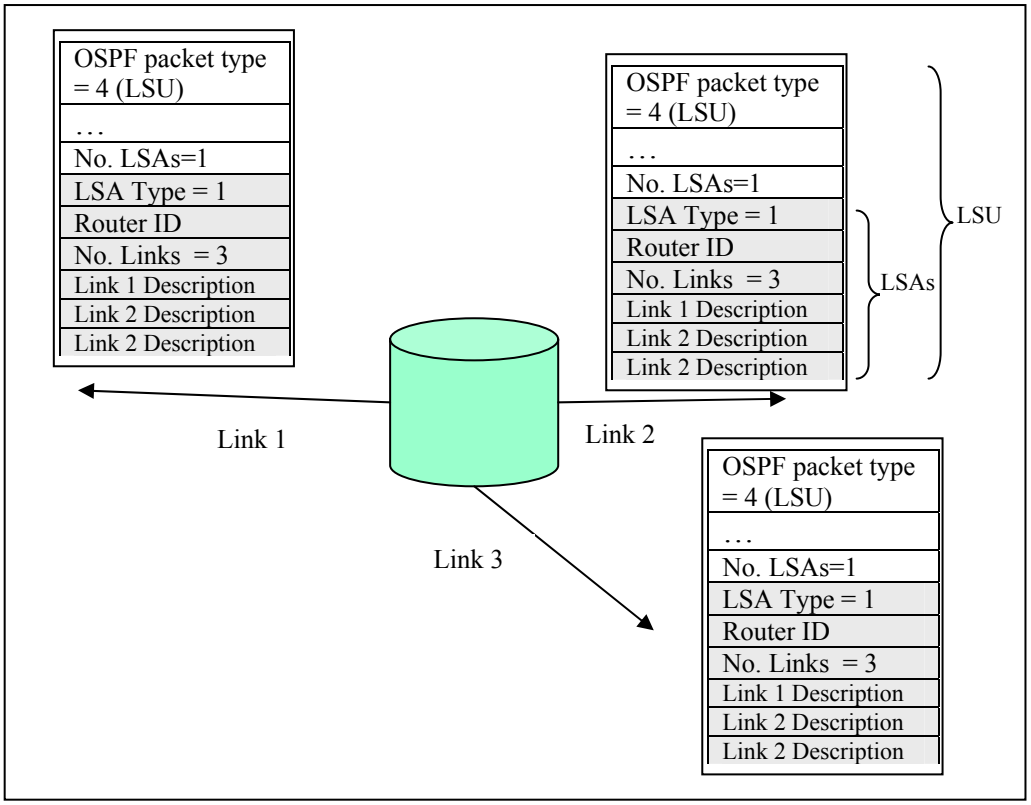


Figure 6: Flooding LSUs encapsulating Router LSAs

OSPF has been designed to swiftly respond to topology, rather than traffic, change, with the route cost largely based on traffic-insensitive metrics. Indeed the

implementation is optimised for a single metric, either the hop count or an administrative weight[xiv]. Examples of policies include Cisco (up to release 10.3) employing inversely proportional to link capacity [12][xv], later replaced by 10^8/BW[xvi] (line speed bps), ie reference bandwidth / configured bandwidth [13], while vendors such as Bay typically use the hop count configuration [8].

Such a protocol is opportunistic, selecting exclusively the current shortest/least cost path (and other equal-cost paths) to a given destination, ie the optimal route. Alternative, ie feasible, paths that offer acceptable costs, ie second-least cost, third-least cost, cannot be selected by the spf algorithm, even if the cost differential is negligible. Another consequence of this is, after new costs are flooded across the network, if a new cheaper cost path is found traffic will be rerouted across this. Although the original path may have been able to meet service requirements the opportunistic approach will automatically reroute. If a rapidly changing metric such as available bandwidth is selected this may result in frequent traffic oscillations. In turn users may experience variable delay and jitter, compromising their quality of service. Imbalance can also result in the network due to the shortest path calculations, with least cost paths potentially converging over the same links. This potentially leads to congestion over the optimal routes, with relative sparseness of traffic across other sections of the network, including other feasible routes to the given destinations.

[xiv] coded as a 16-bit integer
[xv] This parameter is still employed by some researchers investigating QoS routing
[xvi] This gives a cost of 1 for FDDI/fast Ethernet, 6 for token ring and 10 for Ethernet. The default reference bandwidth of 10^8 can be changed for media with higher bandwidths (such as Gigabit Ethernet)

3 Quality of Service

Servicing the demands new applications or users presents a challenge to the best-effort paradigm of IP networks. While the prevailing model in the telephony networks is characterised by offering QoS guarantees this is not intrinsic to the IP service model. To provide resource allocation across such networks thus requires investigating whether the current paradigm is sufficient – indeed that QoS can be achieved across an unmanaged best-effort network – or whether efficient management will be required.

A related concern is that of traffic engineering (TE) – the aim to optimise both network resource utilisation and traffic performance [14]. The traffic oriented objectives of traffic engineering overlap with those discussed below when addressing the notion of QoS for traffic streams. It should be noted that in best-effort networks minimizing packet loss is the key objective; with multi-class networks characterized by demanding applications/users other objectives such as delay become more critical. The resource oriented performance objective of traffic engineering focuses on ensuring that some links in a network are not congested while others are lightly utilised. Congestion may occur due to insufficient network resources – this problem can be ameliorated by enhanced provisioning (see section 3.1) or congestion control techniques such as queue management. However, the focus in this research is where inefficient resource allocation results in over- and under-utilised links/areas in the network. Traffic engineering, notably load balancing, can obviate the congestion resulting in both improved traffic profiles and network optimisation. The research presented here presents a novel means of spreading traffic over less-utilised links, thus can be considered traffic engineering for a resource allocation problem.

This chapter addresses the issue of QoS – what it is, whether it is indeed presents a challenge to IP networks and looking at research that addresses its provision. QoS, however, remains a loosely defined term: some characterise it by explicit measurable parameters; others focus on less precise notions of user perceptions. The International Telecommunications Union (ITU) definition of QoS emphasises “perceived QoS”, ie

reflecting the user's experience of a particular service: "*the collective effect of service performance which determines the degree of satisfaction of a user of the service*" [15]. By contrast an IETF definition focuses on 'intrinsic QoS', ie technical parameters that can be measured and compared against promised service: "*a set of service requirements to be met by the network while transporting a flow*" [16]. Various network or technology level QoS parameters are listed in Table 1, from [17].

Category	**Parameters**
Timeliness	Delay (latency)
	Response Time
	Jitter (variation in delay)
Bandwidth	Systems-level Data Rate
	Application-level Data Rate
	Transaction Rate
Reliability	Mean Time to Failure (MTTF)
	Mean Time to Repair (MTTR)
	Mean Time Between Failure (MTBF)
	Percentage of Time Available
	Packet Loss Rate
	Bit Error Rate

Table 1: Network QoS Characteristics

Such guarantees, however, can vary in precision, as outlined in [18]. For example, quantitative (or hard QoS) specifies hard guarantees for the QoS parameter. In such cases a contract could guarantee, for example, that delay is less than 150 milliseconds. The statistical guarantee allows for some deviation from the quantitative measure, using a probabilistic measure such as 95% of the time delay to be less than 150 milliseconds. The qualitative approach is more imprecise, allowing for more flexibility with implementation but more uncertainty over fulfilment. Finally the relative guarantee, probably the weakest of the categories, considers performance relative to another guarantee in the same system, for example better than a lower priority QoS class.

Moreover the user demands can be mapped explicitly (to specific requested throughput, latency etc) or implicitly (ie corresponding to the requested service class).

Different services have different demands: VoIP is sensitive to packet delay and its variation (ie jitter) but less so to packet (ie information) loss; jukebox services are less demanding with respect to delay [19]; for telemedicine delivery accuracy is more important than either jitter or overall delay [20]. Additionally, QoS can also be defined in terms of transparency and accessibility [21], or high availability and provision of an even traffic load distribution [22]. provides a mapping of QoS delay requirements for various applications, based on a user-centric (ie ITU-T) model [23].

Error Tolerant Application	**Error Intolerant Application**	**QoS specification**
conversational voice and video	command/control (eg Telnet, interactive games)	Interactive: delay << 1s
voice/video messaging	transactions (e-commerce, email, web browsing)	Responsive: delay ~ 2s
streaming audio and video	messaging, downloading (FTP, still images)	Timely: delay ~ 2s
fax	background (eg usenet)	Noncritical: delay >>10s

Table 2: ITU-T Model of User-Centric QoS

More precise definitions set out in [24], specify classes of service, ranging from class 0 (real-time highly interactive traffic that is sensitive to jitter) through class 3 (interactive transaction data) to class 6 (for default IP applications, with unspecified upper bound for mean delay, loss ratio etc).

In technologies such as Asynchronous Transfer Mode (ATM) QoS refers to set metrics, such as delay or jitter, that apply to a connection once it has been accepted [25]. Connections are only accepted when there are sufficient resources both to set up the call at the required QoS throughout the network and to maintain that of any existing calls. However, to integrate existing heterogeneous systems in order to provide for this is highly complex. By contrast the IP model considers network hardware as a transmission platform, with functionality residing in the software located in host servers or routers. It can be debated whether QoS can be achieved across such networks by allowing for an abundance of bandwidth, or whether it can only be achieved through a combination of management and novel technologies and protocols. This is addressed in the following sections.

A final point to note is that while the literature generally discusses 'optimal' routing, more accurately the selection of low-delay routes (the 'optimal' choice for each user) results in a Nash Equilibrium [26]. In general such equilibriums rarely coincide with social optimisation, and indeed total network latency is not-minimised. Thus routing along least-cost paths can be termed 'selfish' rather than optimal. Although this research concentrates on connection-oriented networks it is nevertheless valuable for indicating that optimal routing (from the perspective of the user) is inherently selfish, resulting a degraded network performance.

3.1 QoS Unmanaged Solution: Over Provisioning

Network congestion can be considered as symptomatic of insufficient network resources. A solution to this would be enhanced bandwidth provision rather than seeking to manage / control network traffic. In networks characterised by bandwidth abundance, bottlenecks would never arise, hence a best-effort service (whereby traffic is transmitted according to the best possible way given network resources) would be entirely sufficient. Consequently there is no need to differentiate between user flows, either on the basic of customer or application demands, and the network architecture can remain straightforward. With the advent of technologies such as wavelength division multiplexing (WDM) over provisioning of bandwidth has become feasible. Research has indicated that for links with capacity greater than 1 Gb/s, even at utilization levels around 80-90%, adding network management would decrease delay across the network by merely 4ms [27]. The QoS improvement to even stringent applications such as VoIP would be so marginal as to be unnoticeable. Confidence of a bandwidth glut has even led to concern over bandwidth outpacing processing [28]. Furthermore, an analysis of data networks claims that estimates based on the average size of data networks has greatly exaggerate the volume of data traffic and that IP networks are utilized at a low fraction of their capacity [29].

Although the following section details a range of techniques designed to explicitly implement QoS in practice the penetration of such approaches has been limited [30]. Despite the existence and availability of alternative technologies, over provisioning is

often the chosen approach. Furthermore, although much research in QoS provision focuses on prioritizing classes of traffic (from premium to best-effort), there may be organisational impediments that again prevent this happening in practice. Thus, business as well as technical issues appear to support the over provisioning claim.

However, it can be debated that over provisioning is not feasible beyond the network core [31]. The work in this thesis considers an access rather than such a carrier network. With an unpredictable demand model for data traffic [32] it is argued that improving network dimensioning in itself is unlikely to cope with future Internet usage, and may aggravate the problem [33]. Indeed, it has been demonstrated that techniques designed to reduce network load, such as proxy caching – where an intermediate server caches documents for a set of clients – are conversely responsible for an increase in bandwidth consumption [34]. Incomplete HTTP transfers, ie those aborted by user request, could consume 18% more bandwidth than in a system not operating with proxy servers[xvii], due mainly to bandwidth mismatch. As a consequence, the authors of [35] suggest the importance of modelling user behaviour when considering network provisioning. The behaviour profile of impatient users – who interrupt a transfer when frustrated by poor network performance, eg delay, low throughput – should be included when analysing network capacity. Another impediment to caching is the increased use of cookies – ie personalisation of web browsing. It would appear that an approach that is designed to lower the traffic burden has been undermined by lags in network upgrade, advances in application provision and user behaviour. This suggests that over-provisioning alone may not be efficient or sophisticated enough to provide for future services. Furthermore, an analysis of network-wide traffic flow has revealed that a small proportion of demands is responsible for the bulk of traffic [36]. It is argued that should such sources alter their behaviour, large-scale network variability will result, thus traffic engineering is critical for controlling such demand.

Finally, it may not be in the interest of the Internet service providers (ISPs) to treat all customers equally, ie to not differentiate. Without explicit resource management there

[xvii] If the proxy continues to download upon aborts

can be no varying tariff levels, thus the ISP misses out on potential profit margins [37]. In an arena characterised by commercial competition and high equipment costs, differentiating provides a means of increasing network revenue without equivalent investment in network infrastructure.

3.2 QoS: Resource Management

Having rejected the unmanaged approach, this section briefly outlines various solutions to the perceived need to manage service across networks. The approaches include enhancements to the IP protocol suite, technology shifts as well as augmentations to established routing protocols. It has been argued that many of the proposed schemes ignore the interaction between TCP and the lower layers [38]. Such an analysis is beyond the scope of this research. QoS solutions can be broadly subdivided into three blocks, or planes: management, control and data. The management plane is responsible for issues such as network policy, provision of service level agreements (SLAs), ie contracts, and metering. The control plane covers admission control, QoS routing and resource reservation, ie mechanisms for affecting the traffic paths. Techniques in the data plane include queuing and scheduling, packet marking and traffic classification, policing and shaping, ie those directly involved with the data traffic. The focus of this thesis is on the control plane, specifically QoS routing, although this necessitates employing an appropriate scheduling policy.

3.2.1 Integrated Services (IntServ)

The aim of the Integrated Services model (IntServ) was to offer precise per-flow service provisioning in the Internet [39]. The IntServ architecture offered two new service classes – guaranteed service (GS) and controlled load service (CL) – in addition to the traditional IP best-effort service. GS resembles the ITU Telecommunications Standardization Sector (ITU-T) dedicated bandwidth (DBW) transfer capability, and was developed for real-time applications. CL service resembles the ITU-T statistical bandwidth (SBW) capacity and was planned for

elastic applications with an expected QoS level. The Resource ReSerVation Protocol (RSVP) was used as the end-to-end signalling protocol [40]. This protocol is responsible for carrying reservation requests – the traffic specifications, network resource availability etc – through the network As RSVP uses a soft-state mechanism, a refresh of a path used by a session is necessary after a regular interval (typically 30 seconds).

Scalability limitations have served to hamper the commercial implementation of the IntServ/RSVP architecture. The precise granularity offered by IntServ, specifically the per microflow service guarantees which demand ever router maintains per-flow state, undermines its operability in large-scale networks [41], although deployment in smaller networks may be manageable. In response to these concerns the IETF developed the Differentiated Services model.

3.2.2 Differentiated Services (DiffServ)

Faced with the scalability concerns evident in IntServ, a model that provides for coarser granularity was proposed [42]. The Differentiated Services model (DiffServ) addresses the scalability concerns inherent in the stateful approach of IntServ by providing coarser granularity. This 'stateless' approach, by contrast, keeps complexity to the network edge, as traffic enters the network, whereas the network core remains simple. At the edge routers packets are aggregated into service classes, which are given differentiated treatment inside the network. All the classification, marking and policing takes place at the edge of the DiffServ domain. Packets belonging to a particular flow, or Behaviour Aggregate (BA) are marked with the DSCP in the SERVICE TYPE field of the IP header (see section 2.1) based on agreed policy at the domain boundary. Subsequent core routers apply specified queuing or scheduling behaviour – per hop behaviour (PHB) – based on the DSCP. All packets with the same DSCP are treated equally. Expedited Forwarding (EF) and Assured Forwarding (AF) form the known PHBs. The premium service, EF PHB, has been designed to support applications that demand low jitter, loss and delay. This service seeks to emulate a virtual leased line, providing a guaranteed peak bandwidth service with negligible queuing delay. The AF PHB offer similar delay characteristics as

(undropped) best-effort packets. The strength of its guarantee is dependent on how each link is currently provisioned for bursts of assured packets.

While DiffServ is more scalable its critics point both to lower flexibility and coarser assurance level compared to per flow mechanisms. Solutions such as dynamic core provisioning [43] have, however, provided means of providing fairer provisioning within traffic aggregates, although the centralised nature of the algorithm may raise scalability concerns. While a standard DiffServ guarantee may be, for example that premium traffic receives better handling than low-priority traffic, enhancements such as proportional difference [44] further refine the class differentiation. Also highlighted is the problem of scalable and robust admission control. Additionally, solutions such as DiffServ that keep per-flow state only at edge-routers are potentially less robust – one mis-configured edge router can affect the entire domain [45]. Indeed another major concern raised about DiffServ is the complex management required: routers must be precisely configured (using a complex configuration command script, with reconfiguration only possible through rebooting) and the QoS promised by the system must be closely monitored [46]. Another issue, raised in [47], is the limitations of the DiffServ "boundary-centric operational model". Signalling both from the network core to the DiffServ boundary, and from the boundary to the client/end application needs to be defined. Despite these qualifications, DiffServ is being adopted both within the MPLS world and by many investigating QoS routing. Section 4.5.2 presents policy-based management approaches that have been designed to ameliorate the management of DiffServ networks.

3.2.2.1 SCORE / DPS

The requirement for routers to maintain per-flow state in the IntServ model limited its scalability, and hence deployment. Another approach that seeks to preserve the per-flow granularity without burdening the routers is Dynamic Packet State (DPS), also known as the SCORE (stateless core) architecture [48][xviii]. Instead of locating the information necessary for providing the precision of IntServ service gurantees inside the routers, the per-flow rate information is now stored in the IP packets themselves.

[xviii] also referred to as Core-Stateless Fair Queuing (CSFQ)

As in DiffServ, edge-routers differentiate between end-to-end flows, ie provide per-flow management. This enables the support of per-flow DiffServ delay guarantees. Unlike IntServ the core routers no longer perform this task, turning a stateful network into a stateless one, ie the 'stateless core'.

As packets arrive at the edge routers the flow state is computed and inserted into the IP header. A major, if not the critical, problem with this approach is this use of the header. As discussed in section 2.1 there are a limited number of bits in the header for QoS differentiation. Additionally migrating adaptations / enhancements may in practice be problematic. Solutions suggested for DPS include the link layer and network layer headers, as an IP option or somewhere (ie finding some spare room) in the IP header. The second option may be the most feasible, though in practice this could still be challenging. The other two suggestions, however, are unlikely to be taken up as they require a major adaptation to the IP packet format. This radical alteration to pre-existing packet format undermines the chances of deployment of this approach [49].

3.2.3 Multiprotocol Label Switching (MPLS)

Multiprotocol Label Switching (MPLS) [50] provides a flexible means of establishing reserved paths across networks, thus guaranteeing the appropriate level of service requested. By aggregating traffic into simultaneous flows, known as forward equivalence classes (FEC), the aim is to enable scalability as well as reliability. Complexity is confined to the edge of the network, leaving the core simple, again to ensure scalability. Edge Label Switched Routers (LSRs) apply labels to packets entering an MPLS area. Other LSRs then use this label to forward the packet until it reaches its egress edge LSR, which removes the label. The path through the network is termed a Label Switched Path (LSP). At each hop along the LSP the MPLS label is used to ascertain the next hop in that LSP. The Label Distribution Protocol (LDP) sets the procedures by which the LSRs establish an LSP through the network, ie the means by which MPLS can support QoS. No single protocol is established in the MPLS architecture; protocols such as Constraint-Based LSP Set-up using LDP (CR-LDP) [51] or RSVP-TE (RSVP with traffic engineering extensions) [52] can be employed.

Additionally, to further support QoS, DiffServ BAs can be mapped onto MPLS, as set out in [53]. Aggregate flow can be mapped onto the LSPs that most closely offer the required DiffServ objectives. This necessitates fitting DSCP settings in the 3-bit experimental (EXP) field in the MPLS header.

An advantage of the connection-oriented path scheme of MPLS is that traffic can be shared between two paths, even when link costs are unequal. Using the shortest path paradigm, traffic can be split only over equal (lowest) cost paths. Thus, as shown in the small network in Figure 7, traffic from router R1 to router R4 will be sent via R3 if routing with Dijkstra's algorithm. The links from R1 to R4 via R2 will be underutilised. As congestion builds up over links R1-R3 and R3-R4 the costs may increase, making the route via R2 cheaper. This results in route flapping. Using MPLS, however, signalling protocols set up paths for each flow, reserving resources along these paths. This may result in fewer network oscillations.

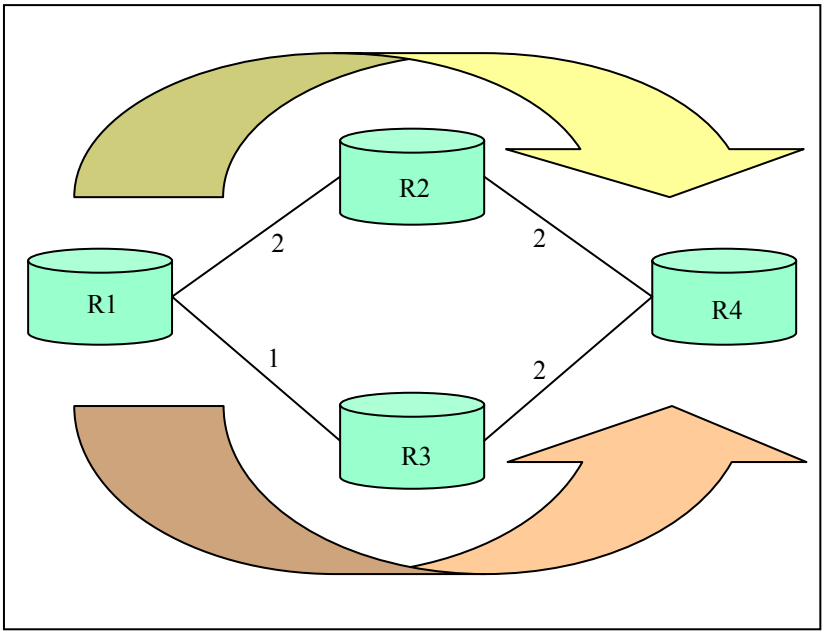


Figure 7: Unequal Cost Paths

MPLS is effectively a shim-layer between level 2 and level 3 (in the TCP/IP protocol model), ie between the data link layer and the network (IP) layer and is not as such a protocol. When used in IP networks it can be considered as a means to provide connection-oriented service in a connectionless network. As such a thorough analysis

of QoS and MPLS is beyond the scope of this research, which investigates connectionless ("cloud") rather than connection-oriented ("string-oriented") networks [54]. Despite some doubts – both technical (eg scalability) and economic – being raised about the widespread deployment of MPLS [30], initiatives such as BT 21C [55] suggest such predictions may be unduly pessimistic.

3.2.4 QoS Routing

Under QoS routing[xix] packets are forwarded based not only on the resource availability in the network but also according to the requirements of the traffic flows, for example guarantees offered by service providers. As outlined in section 2.2, routing using native OSPF is optimised for hop count or an administrative weight. The main objectives of QoS-based routing, as stated in [16], are considered to be dynamically determining feasible paths and optimising resource usage. In OSPF non-optimal costs cannot be used to route traffic, even if network resource optimisation would be improved by doing so. Although resource consumption can be limited by minimising hop count (where this is the prevailing metric), so aiding network resource efficiency, these hops may be heavily loaded. Network resource efficiency may also be optimised by spreading network load, ie seeking to utilise least loaded paths. This optimisation trade-off cannot be effectively addressed with the standard OSPF implementation. This section investigates the body of research that has investigated QoS enhancements to the OSPF routing protocol, sometimes termed QOSPF. An overview of routing strategies is presented in [56]. The performance of these enhancements is often comparable to that obtainable through technology shifts such as MPLS. This is considered advantageous as deployment across networks would be more straightforward. While conceding that optimisation my not be an attainable goal, manipulating the OSPF cost metric can prove an impressive resource allocation strategy.

An additional concern is that even when apparently indicating network availability, the routing tables generated in OSPF are based on imprecise state routing information [57] due to network dynamics, approximate calculations, routing aggregation and

[xix] Also known as constraint-based routing

hidden information (eg for security reasons). Indeed [58] argues that 99% of routing information was inaccurate at that time in the Internet. Approaches that attempt to infer resource availability probability information, sometimes termed 'probability based routing', have been introduced to compensate for the shortcomings of availability based QoS routing.

QOSPF [59] presents a refined version of OSPF that incorporates both link bandwidth and propagation delay. A "widest-shortest" (ie minimum hop with maximum bandwidth) path is pre-computed. See also [60, 61]. Source routing[xx], ie where a path to the destination rather than the next hop is computed, is employed in some models [62], contrasting to the exclusively hop-by-hop approach presented in this work. Similarly, the Cost-based QoS Routing techniques employed in [63] and the QoS system in [64] are explicitly designed solely for MPLS networks, not connectionless ones. Although the work in [65] also runs over an MPLS network, its employment of sub optimal paths is pertinent to the research presented here.

Another point to note is that much of the cited research primarily focuses on the traffic of one service class in the network, rather than addressing sharing resources between traffic requiring differential handling. Conversely, the work in [66] examined the ramifications of QoS routing on best-effort traffic in both lightly and heavily loaded networks. Selecting shortest-widest paths, for example, even in lightly loaded networks were shown to adversely affect the throughput of the best-effort traffic; QoS routing has, perhaps surprisingly, been demonstrated as desirable even when networks are lightly loaded [1]. That work furthermore determines that relying on data plane techniques alone by statically partitioning link resources [67] is inadequate to the challenge of multi-class routing.

The research outlined in [68] examines how routing protocols, including OSPF, can emulate "optimal routing", ie following an ideal set of paths and loads identified from using information about traffic entering and leaving networks. An optimal distribution of traffic is impossible due to the inherent constraints of shortest path routing (with

[xx] either loose or full source-routing

destination based forwarding) and splitting traffic solely over equal cost shortest paths. Furthermore the OSPF weight setting problem has been demonstrated to be NP-hard [69]. However near optimal results were obtained by approximating optimal link loads and applying novel traffic splitting heuristics. Performance levels from these experiments were comparable with those obtained using MPLS. These results are significant as they indicate that it is not necessary to anchor investigations into Internet QoS to novel technologies. This work reinforces the finding of earlier research that investigated optimising OSPF weights in order to enhance traffic engineering [70]. That earlier research had demonstrated that with appropriate weight settings 50-100% more demand could be supported than using Cisco's default[xxi] and approach within a few percent of the best possible routing including MPLS. Later work by the same authors developed their local search heuristic to accommodate link failures by focussing on critical links [71].

A consequence of QOSPF is a raised level of LSA flooding, due to shifts in link costs [59]. Experimental results [72] have demonstrated that flooding small packets such as LSAs consumes a small percentage of bandwidth, so should not represent a burden on an already congested network. Additionally the overhead caused by updating the link state databases and generating routing tables should not be problematic for modern router CPUs. Additionally research on reducing routing table computation overhead, such as the "divide-and-conquer" scheme [73] or router clustering [74], mitigates the router load.

However, the convergence issue is of greater concern. The work in [75] investigates routing around link failure by allowing weight changes. It may take a few seconds for all routers in the network to return to a steady state – ie for each router to update its link state database and recalculate the corresponding routing table[xxii]. During this time routers will have inconsistent link state databases. This may lead to looping (see earlier) if, for example node "A" routes all packets for destination "G" to next hop "B" and this node "B" routes all packets for destination "G" to next hop "A". Packets

[xxi] albeit, the research used the outdated Cisco inverse-capacity-weight metric

[xxii] Although beyond the scope of this research, convergence time takes even longer in a connection-oriented network as traffic engineered paths have to be rerouted – old paths torn down and new ones set up – after network perturbations.

will bounce between “A” and “B” until time out or till convergence, resulting in network inefficiency. Suggestions that address the accuracy versus overhead trade-off have examined when to trigger link state updates [72], thus lessening the rate of convergence. Choosing a higher threshold, so generating a LSA only after a sufficient rise in cost metric can be an acceptable compromise. The loss of accuracy in the link state databases often does not greatly reduce network performance. An alternative approach includes the time to detect failure in the convergence time. Modifying the Hello interval so that they are in the sub second range has been demonstrated to significantly reduce convergence time [76], provided that the interval be sensitively set. The research found that reducing the Hello interval further – to the millisecond range – resulted, however, in route flapping due to increased Hello timeouts. Despite this, millisecond convergence is considered necessary for high-availability and forms an area of active research [77]. Since strictly following the OSPF protocol results in a relative high granularity of failure – minimum 2 second detection – another approach is to employ the bi-directional forwarding detection protocol (BFD) to track connectivity [78]. Another approach has been to reduce the interval between the periodic update floods: the default interval of 30 minutes is reduced to 2 seconds in [79]. This is unlikely to be feasible, as it would result in continuous database updating.

A criticism that can be levelled at much of the work in QoS routing is that it fails to address the performance of best-effort traffic. A ‘best-effort-friendly’ (‘BE-friendly’) method, presented in [80], selects QoS paths that minimize best-effort delay. However, the network under consideration implements MPLS: all QoS traffic follows LSPs; best-effort traffic is destination, hop-by-hop routed.

3.2.4.1 Opaque/Traffic Engineering LSA

This section looks at an example of QoS routing – OSPF-TE – in greater depth. The discussion of OSPF in section 2.2 considered the deployment of Router LSAs (OSPF LSA type 1). Enhancements to OSPF, notably OSPF-TE utilise the novel opaque

LSA[xxiii] (type 10, flooded within an area), defined in [81]. This allows supplementary information about link states to be inserted into an LSA.

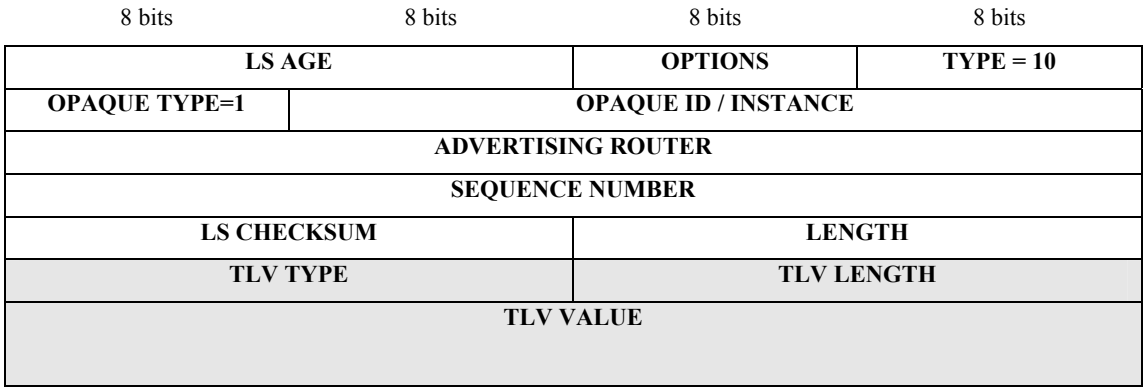


Figure 8: OSPF Opaque LSA

The link state ID – 32 bits in the router LSA – is now decomposed into the 8-bit opaque type field and the 24-bit opaque ID. The Traffic Engineering (TE) LSA [82] uses type 1 of the former field, and refers to the latter field, which has no topological significance, as the 'instance'. The purpose of this field is to allow the maintenance of multiple traffic engineering LSAs. The Type/Length/Value (TLV) type specifies the type of information carried; the length field specifies the length of the value field in bytes or octets; the value field contains the actual value. In the (TE) opaque LSA the TLV triplet, termed a link TLV, encodes link-specific information including maximum link bandwidth (ie true link capacity), maximum reservable bandwidth and unreserved bandwidth.

The novel LSA is flooded in the same manner as router LSAs, and the Link State Database now incorporates the extra traffic engineering data. Using this extended datastructure, now termed the TE database (TED), routers are able to compute end-to-end MPLS paths offering QoS guarantees. Unlike the native OSPF Link State Database, the TED can be revised by the node as the status of each of its links alters. If approaches are employed to reduce LSA flooding, router databases will no longer be synchronised and looping may result. This could be alleviated by more frequent flooding. However, contrary to the findings reported in [72], protocol overhead is

[xxiii] Opaque LSAs can only be flooded to opaque-capable neighbours, ie those who set the O-bit in the Options field as part of the neighbourhood discovery process

substantial. Unlike native and other enhanced versions of OSPF, where LSAs are sent with information regarding the router, with OSPF-TE advertisements are sent for each link. Where nodal degree is high, for example in a dense mesh network, protocol traffic can increase considerably [83]. This work demonstrates the alteration to the basic trade off – between the accuracy of routing information and the overhead due to flooding protocol traffic – by manipulating the OSPF MinLSInterval and MinLSAArrival settings, responsible for controlling the rate of LSAs. This suggests that careful selection of network triggers may enhance the viability of OSPF-TE, albeit in connection-oriented networks. As such this is beyond the scope of this research, but is included to demonstrate both that incorporating extra information into LSAs and adding to protocol traffic are viable management strategies.

3.2.4.2 Alternative Routing

The shift here is from local optimisation, ie the least-cost path, to acknowledging that network-wide optimisation may be obtained through more efficient resource utilization. However, although the ability to select acceptable paths may be desirable, uncontrolled alternate routing [84] is rejected due to adverse performance impact in times of network stress [16]. The attractions of this approach are founded on both feasibility (ie that traffic can follow an alternative rather than being dropped) and fairness (ie sharing resources). Alternate routing is derived from telephony, to support flows that could not follow their primary paths, so reducing network blocking. As network load increases, to avoid being blocked, some traffic is routed to the alternate path. However, this utilises more resources than if all traffic is routed along its primary path. As load increases further the primary traffic on the alternate paths may suffer and in turn become rerouted to a corresponding alternate path. The net result in times of heavy load is inefficient resource utilisation. To ameliorate the impact of rerouting away from the optimal path mechanisms such as using state protection to prioritise primary over alternative traffic can be employed. Under this scheme alternate routing is blocked once utilisation on that path is above a certain threshold.

The obvious objection to the above approach is that OSPF selects purely the shortest cost (or equal shortest cost) paths. To allow for selection of alternate paths would

require an overhaul of the protocol, or use of connection-oriented techniques beyond the scope of this research. However, it is included here as background towards the enhancements developed later in this thesis.

4 Agents

An agent is a software engineering abstraction that has proved elusive to precise definition. The major characteristic that probably all definitions agree on is autonomy, such that the designer delegates to rather than instructs the agent. An elementary agent definition considers it as an entity that perceives its environment through sensors and acts upon that environment through effectors / actuators [85]. More developed, although still simplistic, definitions describe a software entity responsible for automating tasks [86]. Various alternative definitions adapt this to incorporate the properties that are considered essential to distinguish an agent from a program or object or other software-engineering abstraction – some stress goal-directedness, others mobility, others learning, others communication skills or sociability and others focus on response in a timely fashion, or location in some 'real world'. More specifically the authors of [87] identify the following dimensions that characterise agents: autonomy, reactivity, proactivity, responsibility, continuity, interactivity, adaptability, rationality, cooperation and robustness.

4.1 Parent Disciplines

While not aiming to provide an in-depth analysis of the history of intelligent agents, an overview of the parent disciplines provides clues to why there is some confusion about what constitutes an agent. Agents can be seen to have emerged from concurrent actors (themselves a product of Distributed Artificial Intelligence, DAI), where an actor: "*is a computation agent which has a mail address and a behaviour. Actors communicate by message-passing and carry out their actions concurrently*". However, more recent understandings would expect behaviour beyond simple message passing and concurrent action.

The approach delineated in [85] stresses the artificial intelligence (AI) origins of agents. According to this analysis, software agents fall under the 'acting rationally'

quadrant of AI[xxiv]. This is in contrast to thinking humanly (the cognitive science approach), acting humanly (as investigated in Turing's imitation game [88]) and thinking rationally (the 'Laws of Though', ie a purely logic-based approach). By contrast Műller in [89] promotes the importance of cognitive psychology alongside classical AI planning systems to the development of the agent paradigm. To these he adds control theory, with a footnote acknowledging object-oriented (OO) programming and distributed systems (this latter is further stressed in [90]).

In common with the above texts the analysis here bypasses the cognitive science / psychology links. While most work on agents in telecommunications has stressed the AI nature of agents, control theory will be reconsidered later in this section. This may prove fruitful for examining why 'agents' have not been as widely deployed as predicted. Significantly, if agents initially grew out of control theory and AI planning, but then diverged from the former, one would not expect to see the term agents deployed in control theory research. This implies that structural, or institutional, issues have hampered agent progress. Or, to be more precise, that agents may have developed outside 'agent'-friendly departments and as a result not been ascribed as such[xxv]. Thus if the limitations identified by Műller that inhibit the agent side of control theory have been lifted, then it can be argued that intelligent control theory is another element of agent development. Indeed, the description of the reinforcement learning problem in [91] states:

> "*We use the terms agent, environment, and action instead of the engineers' terms controller, controlled system (or plant) and control signal because they are meaningful to a wider audience*".

In section 4.5.1 aspects of intelligent control will be proposed as agent-based, according to most acceptable definitions of 'agent'. This will be contrasted to some work that has come from agent-friendly departments that fails to adequately demonstrate the application of agents, despite their claims.

[xxiv] Rational action is considered to be where an agent selects the most appropriate action to achieve its goals given what it senses and what it may have been informed about the environment.
[xxv] The control theory parallel development is also noted in [91]

4.1.1 Why Agents in Networks

As stated earlier, networks are increasingly characterised by complexity: an expansion in technologies; the convergence of voice and data networks and infrastructures, enhanced by market deregulation. This can also be viewed as increased network depth – set of services – as well as breadth – the number of users [92]. Due to this growth both in network complexity and traffic volumes there is an increased need for systems / networks / services that are reactive (ie responsive and adaptive in a timely fashion), proactive and decentralised [93]. Distributed, dynamic and open systems demand some autonomy; delegation is necessary in order to manage more effectively compared to human-centred management [94]. Distributed management, instead of a monolithic / centralised structure, would appear to offer advantages such as scalability, flexibility and robustness. However, it is acknowledged that careful consideration should be given to the granularity of agent architecture to avoid unnecessary complication and communication overhead.

An advantage of the agent approach is its capacity to incorporate legacy software. 'Agentification' essentially encapsulates such software inside an agent shell, thus enabling non-agent enabled systems or nodes to work alongside agent-based ones. However, this in turn raises the prospect of the hollow agent – one that appears like an agent but lacks any agent-properties other than those provided by the agent wrapper.

4.2 Agent Properties

Since this field has proved so contentious it is advantageous to attempt to identify more thoroughly the composition of an agent. Furthermore, it is useful to delineate some boundaries that establish how an agent could usefully operate in network environments. There is a considerable focus in the literature on mobile agents, see for example the survey in [95]. However, network managers may prove reluctant to surrender control to unpredictable entities that can be difficult to control. The research developed here exclusively focuses on static agents, ie those confined to a node. The overview of agents in networks, in section 4.4.4, will nevertheless provide an illustration of some mobile agents, notably ants, but the purpose of this is to indicate

the breadth of agent-network research and to illustrate the use of reinforcement learning techniques.

The first ‘simple’ agent definition provided earlier would enable a simple control system, such as a thermostat, or software daemons to be considered as agents. Such a classification is usually refined to incorporate intelligence. A prominent definition of such agents interprets intelligent behaviour as flexible behaviour, ie characterised by reactivity, proactiveness and social ability [96]. Indeed, the stress placed by the authors, Wooldridge and Jennings, is on agent sociability, ie communication and cooperation/negotiation skills.

However, ascribing intelligence to agents is in itself difficult as some architectures afford little behaviour to an individual agent that could be considered intelligent from an AI perspective, as proposed by the cognitive or deliberative school (represented in the DAI domain). Brooks explicitly rejected decision-making based on manipulation of symbolic representations of knowledge (as displayed for example by deductive-reasoning agents, see [97]) and argued that intelligence is not disembodied but is a product of the interaction that an agent maintains with its environment [98]. Intelligent behaviour could be seen to emerge under his ‘subsumption’ architecture from the interaction of various simpler behaviours. Although critics of his work point to the limited applicability of the architecture, emergent intelligence, as championed by the ‘reactive’ school, has also been displayed in multi-agent systems modelled on (social) insect behaviour. In the multi-agent world ants, for example, [see section 4.4.4] are intentionally created as simple, disposable agents – intelligence emerges from the behaviour of the colony rather than through individual deliberation or deduction. Thus the notion of a smart or intelligent agent is not in itself simple: the agent could be Wooldridge / Jennings intelligent (ie collaborative), it could be intelligent from an AI perspective (eg able to learn or to manipulate a knowledge base) or the intelligence could emerge either through interaction with the environment and/or other agents.

As objects become more sophisticated it may be useful to distinguish them from agents. Although agents share many characteristics, objects are structurally simpler

and inherently more passive [99]. For example, an object has to be activated (or invoked) by sending a message. Objects can access all publicly accessible methods of other objects (ie objects have no control over their behaviour); agents can only request other agents to perform actions. Active objects, encompassing their own thread of control, reach closer to the notion of an agent. However, it can be argued that their patterns of interaction are still rigid and pre-designed, and that they lack the fluidity of agent organisational structures.

4.3 'Agents' in Network Protocols

It has been acknowledged that the agent paradigm is challenging. This is not merely due to the above difficulties in agreeing on a consistent working definition of what makes an agent but also due to the pragmatics of engineering such systems, as outlined in [100][101]. There is no doubt to those authors that agents, as they argue, have been oversold – the benefits from such an abstraction tool may also in some situations be achieved using non-agent techniques. This will be further investigated in section 4.5. Their analysis concentrates on novel applications that often fallaciously (or optimistically) claim to employ agents. Additionally, the term 'agent' is embedded in the architectures of various network schemes. This section introduces the proposition that the history of 'agents' in networks has operated orthogonally to the development of the agent paradigm (derived from AI, control theory and cognitive science). This argument is more extensive than the statement that 'agents' in network literature / architectures differ from 'software agents' or 'intelligent agents'. The proposition here is that the limited capabilities that constitute 'agents' in some network protocols have dampened the expectations for agent technology. In turn this permits enhanced 'agents' to be developed without the sophistication or flexibility promised for either 'true' software agents or their framework. The following forms an introduction to a thorough, and much needed, analysis of the deployment of 'agents' in IP networks, examining how these are deployed in, for example, mobile IP, Simple Network Management Protocol (SNMP) and DiffServ.

Mobile IP is designed to enable transparent routing of IP datagrams to mobile devices (such as laptop computers) in the Internet [102]. In mobile IP each mobile device (termed 'node') has a home address that corresponds to its home network. When the node roams outside its home network any packets addressed to this home address have to be forwarded. The router or node responsible for both tunnelling datagrams to be delivered to the mobile node and for maintaining the location information regarding this node is termed the 'home agent' (HA). For delivery to be successful the mobile node must register with another entity, also termed 'agent' on the new, or 'foreign', network. This foreign agent (FA) allocates a new IP address, termed the care of address (COA), to the mobile node. This COA is then registered by the mobile node with its HA through the exchange of a Registration Request and Registration Reply message. The HA encapsulates any packets destined for the mobile node and tunnels it to this registered address. The FA in turn de-encapsulates the packet and forwards it to the mobile node. Without these delivery agents, as a node changed its point of attachment it would lose its ability to communicate.

Yet it is arguable whether these agents are indeed agents, in a form distinguishable from a 'router' or an 'entity' or just a program. While they 'communicate' with messages, such as Agent Advertisements, this lacks the sophistication of a speech action protocol, as outlined for example by FIPA [103]. Although this would fail the Wooldridge / Jennings agent definition, it should be conceded that communication skills are not stressed in all agent definitions or practice. However, the agents presented fail to accord with either maximal (eg Wooldridge / Jennings) or minimal (eg 'simple') definitions of agents: decentralised management and elementary communication is not sufficient.

In SNMP [104], the TCP/IP standard for network management, 'agents' are again employed: there is a manager-entity ("traditionally called an agent"[xxvi]) relationship, as originally devised for OSI systems management [105]. 'Agents' in each device – such as a bridge, router, hub and switch – are responsible for data collection regarding the managed object. This information is stored inside a Management Information

[xxvi] Case et al, section 2.1, p.2

Base (MIB). The agents are polled by the SNMP management station with requests for information on that device's operational status. The management station then displays the retrieved information for analysis by a network manager.

The RFC for SNMP acknowledges that calling the SNMP entity in each node an agent is a consequence of the established naming (ie established in the earlier RFCs); the terminology is not due to inherent agent-like properties. A sample glossary [106] provides the following definition of agent: "*In network management an agent is the server software that runs on a host or router being managed*", which again fails to accord with even a generous definition of an agent.

Furthermore, the transformation of agent as simple component into agent as complex software engineering abstraction (the agent paradigm) is a point of confusion in more general[xxvii] agent literature. In [107] the concept of agent-manager via SNMP is introduced as evidence of 'agents' as 'indispensable tools' for network managers. Such agents are then contrasted to the superior performance of 'intelligent' agents. However, it is stated these smarter entities that can perform the dual roles of manger and agent have this additional capacity due to code that "*tells them exactly what to do, how to do it, and when to do it*". Autonomy has been identified as perhaps the one characteristic (albeit problematic) that those seeking for agent definitions can agree on. Since autonomy implies delegation rather than instruction then these intelligent agents, albeit smarter than SNMP agents, are again also not really agents. By constructing such a low unfocussed baseline for agents the result is that other entities become included under such a nebulous heading. The redundancy of the term merely serves to limit the practical application of the paradigm.

This confusion can also be found in DiffServ, where bandwidth brokers are explicitly called agents in the RFC [108]:

> *Thus this architecture is designed with agents called bandwidth brokers (BB) [2], that can be configured with organizational policies, keep track of the current allocation of marked traffic, and interpret new requests to mark traffic in light of the policies and current allocation.*

[xxvii] ie software agent, not protocol

This has resulted in inconsistencies in research papers in this field: research that appears to present agents instead describes enhancements to the bandwidth broker concept. Thus in the abstract of [109] the two have merged: "*For each link-state routing domain in the network there is a topology aware QoS agent (also known as a bandwidth broker)*". This paper confirms that the agents in the authors' earlier works, compiled in [110], are synonymous with bandwidth brokers. That bandwidth brokers are entities that are delegated the responsibility of traffic marking appears to conform to the agent paradigm. Yet their role lacks the flexibility associated with that abstraction – the aim of delegation goes beyond mere distributed control. The flexibility, above all the sociability, of the Wooldridge / Jennings model, is lacking. While conceding that this is only one of many definitions of an agent, the bandwidth broker fails to incorporate other properties associated with agents, for example omitting any AI.

Again, the Snoop protocol [111], developed to improve TCP efficiency in wireless networks, also deploys 'agents'. These entities are 'TCP modules', responsible for monitoring and caching all packets passing through the agent's base station. When packets are lost[xxviii] the agents retransmit them locally without forwarding the ACKs to the sender. Since the TCP layer remains unaware of packet loss, the congestion control algorithm is not triggered. The Snoop protocol is an example of Performance Enhancing Proxy (PEP), ie a method aimed at reducing performance degradation due to the characteristics of wireless links. The 'agent' in the protocol would appear to be the 'entity' – TCP-aware module – that enables the PEPs. Again, it could be seen that action is performed – Snoop is enabled – rather than an agent deliberates / decides / negotiates. The 'agents' are merely distributed entities – possibly actors.

Finally, the Sequence Agent (SA) – developed in the packet sequencing architecture – is responsible for coordinating itinerary creation [30]. The tasks of such an agent include validating requests, providing itinerary leases, lease renewal and teardown. In small networks there is one agent; as network size increases multiple agent peers

[xxviii] indicated by duplicate TCP acknowledgements (ACKs)

communicate across domains. As argued in the previous paragraphs, however, there is little that distinguishes these entities as agent.

It could just be accepted there are legacy reasons why the term 'agent' is employed in the literature. This innocuous usage encompasses entities for example with manager-managed/slave relationships, entities that communicate according to protocols, entities that enact organizational policies. It can even be reduced to the most basic definition – something that does something, ie enacts agency – as stated in the following 'characteristic' of the TCP/IP suite: "TCP/IP protocol, and other protocols like it, is a *result* of the action of autonomous agents (computers)" [112]. Alternatively, as highlighted here, we can try to establish that there are extreme contradictions in the usage of this term. Focussing on this is not mere pedantry. Where a term is familiar in one domain, here networks, reintroducing it as a paradigm created from outside the domain (whether AI planning, control theory or cognitive science) results in inconsistencies, potentially undermining the deployment of agent-like agents. As Wooldridge and Jennings noted about the pragmatics of engineering agents [101]:

> "*Ignoring them*[xxix] *will result in a backlash against agents similar to that experienced against expert systems, logic programming, and all the other good ideas that have promised to fundamentally change computing*"

While much of the paper that contains this quote warns about the over-abundance of software claiming to be agents, here the stress is on the relative paucity of deployment. The significant role that it was hoped intelligent agents may play could have been destabilized at a much earlier point by the overuse of the simplistic 'agents' detailed above.

4.4 Agents in Networks

While acknowledging the concerns outlined in section 4.3, nevertheless an agent-based approach has been identified as an apposite mechanism for modelling interaction across networks. Where networks are complex, characterised by a

[xxix] ie the pragmatic aspects of agent technology

distributed and sizeable volume of information, agents offer the necessary flexibility to manage resources. Research has demonstrated the advantages of employing software agents specifically across telecommunications networks, where agents can use intelligence, for example, to negotiate contracts or to exploit resources such as bandwidth in times of congestion. Other research outlined here, it will be argued, utilises structures that are identical to the agent software engineering abstraction in all but name. Yet, also included is some work that claims to be agent-based yet fails to adequately demonstrate the role of agents.

Additionally, a body of more theoretical work has demonstrated the advantage of agents for applying co-ordination and/or negotiation mechanisms [113,114], including trade-offs in telecommunications networks [115]. A more thorough analysis of this is beyond the scope of this thesis but such work compliments the applied agent work.

4.4.1 Agent Architectures for Resource Allocation

The focus in this sub-section is on agent architectures decomposed into hierarchical layers. Higher-level agents are responsible for deliberation, monitoring or collaboration and can disseminate their knowledge down to the lower level, increasingly reactive agents. Likewise, these agents can dispatch their discoveries or problems, to the upper levels.

Deploying agents for resource allocation in telecommunications networks was proved to be an advantageous strategy in [116]. This work utilised agents to provide flexibility in allocating channels in cellular networks, such that cell blocking was minimised and channel usage maximised. Modelled on the INTERRAP architecture [89], the reactive agent layer was responsible for the rapid accommodation of traffic demand, the planning control layer aimed to optimise the local channel load distribution while the top most cooperative control layer focussed on load balancing across a wider area. By decomposing functions into layers, and through coordination the agent approach achieved better flexibility, despite some scalability and robustness concerns. Additionally, all calls were treated equally in this approach – no preference was given for service type.

The IMPACT[xxx] project implemented control strategies on an ATM test bed as a society of interacting agents [117]. The research employed an hierarchical agent architecture, implementing resource management strategies in reactive and planning layers. Two of the resource (management) agents were located in the higher (slower) planning layer – where for example network monitoring occurs – while the remaining resource management agent was located in the rapid reactive layer. The latter agent had to make immediate decisions over network admittance based on limited state so needed to function without the potential delay associated with planning competence. However, the reactive agent was located within the framework of the more strategic competence so, when necessary, higher-level decisions made by the planning layer – such as the bandwidth allocation for pipes managed by that agent – could be relayed down. Various other agents were deployed, for example, to operate as brokers, manage auction bids and to represent service providers.

Successful implementation of the IMPACT society of agents was demonstrated across several test beds, albeit noting overheads due to choice of coding language and implementing SNMP [118]. One of the key concerns about the IMPACT project was scalability: with one reactive agent for every source-destination pair the network suffered severe growth constraints [119]. The agent devised to address this problem, by establishing connections traversing several IMPACT domains, was never implemented due to time constraints. Additionally the directory facilitator agent – responsible for white-pages services – represented a vulnerable single-point-of-failure in the IMPACT structure. Should this agent fail all other agents would become incapable of finding each other.

In the SHUFFLE agent telecommunications project, agents were implemented in a system that dynamically allocated radio and associated fixed network resources in 3G mobile systems [120]. The aim was to provide end-users with an improved and more cost-effective service, and operators with increased opportunities for contingency management where allocation policies need to be dynamically changed. The system

[xxx] Implementation of Agents for CAC on an ATM testbed

evaluated how the resulting resource allocation system improved the overall performance of the network and the scheme was compared with more centralised approaches. The agent implementation allowed the project to explore various resource management strategies. Some of these strategies merely required minimal planning applied at the reactive level, while some required intelligent negotiation between components of the system in the planning layer. The results demonstrate a clear advantage to decentralised control. Additionally the intelligent, reputation-based selection of networks yielded over 25% improvement in blocking and dropping rates compared with conventional network selection (where the network that carries connection request is always asked to handle the call) in dynamic demand scenarios (intermittent hotspots or cell failures, for example). The project also demonstrated that SLA constrained QoS relaxation (by reduction of requested bit rates) yielded an improvement in blocking and dropping rates. Results show clearly that even the sophisticated intelligence of the negotiation of shapes could be performed in real time, as well as the relaxation and referral mechanisms, but the performance of the middleware is critical to any application. The mapping to the agent communication language, the network latency, the processing by conversation managers and the allocation to tasks lead to significant delays.

The hierarchical architecture for MPLS-enabled networks in [121] was designed in response to the scalability concerns associated with the previous agent systems. By making the system complement the conventional management apparatus, robustness to agent system failure was ensured. Two agents were distributed to each node: deliberative P-agents (one per node) for maximising network performance and subordinate reactive M-agents (one per link) for monitoring. Should the M-agent be unable to respond to congestion over its logical path (LP) it alerts the node's P-agent, which then communicates as necessary with the corresponding agent in other nodes to alleviate any hotspot. Additionally, P-agents are intended to incorporate learning.

The work in [122] presents an agent approach to responding to adverse conditions – for example reacting to natural disasters or the added stress of large-scale public events – in (PSTN / ISDN / SDH) telecommunications networks. Traffic Management Networks (TMN) Operational Services (OS) collect traffic information from the

Network Elements (NEs, ie the digital exchanges) and pass commands down as necessary. OS can issue routing controls or traffic volume controls to network level, but traffic management is performed at network management level due to possible network heterogeneity (such as NEs from more than one vendor). Agents are located at each node (ie NE) – they can be particular to vendor or NE type. As in the IMPACT project, a hierarchical approach is employed: the control agent is reactive, running in a multi-agent host system. This system in turn notifies the reactive agent of changes in network status.

Routing in [123] is calculated on-line based on network state. A controller agent is responsible for a region within a network. Such regions are then clustered into meta-regions (in a similar fashion to PNNI [124]), controlled by a parent controller agent, which in turn are grouped into a higher region creating a hierarchical clustering structure. To make this adaptive, these regions are categorised into equivalence classes of nodes reachable at a certain bandwidth, such that a decreasing level of bandwidth mutually connects all nodes in regions higher up the hierarchy. Problems are ideally served locally and then passed up the hierarchy until the controller agent knows the two endpoints. This agent then coordinates the agents below it in the hierarchy to solve the routing problem. As demand rate increases the relative performance of the adaptive routing hierarchy suffered, although the authors argue that in non-uniform traffic scenarios the adaptive techniques should prove advantageous.

As multi-agents systems (here used synonymously with DAI) become larger and the environment unreliable, adaptability – both of the agents and of the interaction structure among the agents – becomes imperative. If an agent's problem space is suitable for machine learning or other AI techniques this ensures adaptability when scaling up. Additionally, including the actions and aims of other agents into an agent's input space, so ensuring the propagation of an agent's policy adaptations to the other agents in the space, can result in more interesting strategic behaviour, as demonstrated by Vidal and Durfee [125].

4.4.1.1 Agent Framework

To complement the work on agent architectures more formal work has been undertaken to improve agent frameworks. The aim of the Agentcities [126] initiative is to create a '*global, open, heterogeneous network of agent platforms and services*'. The focus lies on supporting consensual standards, open source, open access and shared resources. Agents run on different platforms, owned by separate organisations, with differing implementations and diverse service provision. Customers select a network service – essentially a standardised Service Level Specification[xxxi] (SLS) – and then choose further modifications to the SLS, including schedule, extra QoS requirements and traffic description. The initial domain to both test and demonstrate the project was a travel agent platform (ie provider of location-based services). An interest group on wireless applications has sought to dynamically respond to user needs based on location through interaction between agents in both wireless and wired networks. The project still requires further work in developing ontologies, using semantic frameworks and content languages to encourage and enable agent communication. Although such developments are beyond the scope of this thesis it is included to demonstrate that work is still ongoing on agents in networks.

4.4.2 Agent Intelligence: Routing

The purpose of some of the earlier sections has been to examine the claims made for the role of agents in networks. This has necessitated not only establishing what is meant by an agent but also to expose the role of 'agents' in both network protocols and applications. This section investigates the work in agent-based network routing that is related to the research outlined in this thesis. These fit more closely the AI model of agents, primarily using reinforcement learning to update and refine routing tables. An advantage of reinforcement learning is that no prior knowledge (or model) is imposed on the agents – all knowledge and behaviour is learned from the environment. For a fuller analysis of reinforcement learning see section 6.1.1.

In the reinforcement learning model presented in [127] – termed the proportional routing model – the action space of each agent is a proportion vector, consisting of the

[xxxi] The SLS is defined as the technical component of an SLA

percentage of traffic for each destination sent along each outbound link. In the training stage the input for the agent is the action taken by that agent plus any network observations from that time interval, such as the proportion vectors of other agents. The corresponding output is the system-wide throughput for that interval. The advantages of using adaptability in a routing strategy were clearly demonstrated. Unlike some previous work on adaptive agents, based round a Stackelberg game where the 'leader' agent imposes its actions into the other 'follower' agents' action space [128], the research 'interleaves' their decisions so that any agent is both a leader to a certain extent and a follower. Thus each agent includes the actions of other agents in their action space. While there is concern about the extra state that may accrue for each agent the development of agent adaptability is encouraging. However, it is unlikely that this could be extended to an OSPF-enabled network – not only does it employ the Bellman-Ford metric but OSPF does not permit proportional routing.

In another project employing reinforcement learning [129], each router in the network is represented as a partially observable Markov decision process (POMDP). The node decides where to route a packet according to a stochastic policy. This policy computes the shortest path and then sets controllers to route most of the subsequent traffic down the chosen path. Sporadically, traffic is also sent to explore any alternative links. Once a packet has arrived at its destination it sends an acknowledgement signal. This allows routers to calculate packet delivery time, which provides a reward value, which in turn is used to update the policy parameters. The policy algorithm's performance is compared to a static routing scheme and two other deterministic routing algorithms, one based on shortest path and the other on value search reinforcement learning. The results demonstrate a clear advantage of the stochastic approach over the deterministic algorithms.

The work using Q-learning in [130] generates extra control packets by sending link cost information from the next hop (rather than the destination node) to the sender. Oscillatory behaviour was exhibited, and although results proved better than using static routing algorithms, testing against dynamic algorithms was neglected. In [131] agents at every node also employ reinforcement learning – here Q-learning (see section 6.1.1.1) – with results tested against a network solely routing using a distance

vector algorithm. The agents aimed to optimally map state (spare capacities on connections and internal queues) to actions. After an initial period of learning results were considered to be 'promising' for improving both network reliability and efficiency, although the authors concede it is difficult to extrapolate the results to a larger network. A weakness with all the studies is a failure to report on the increased state space that is generated by using reinforcement learning.

In [132] Application Service Providers[xxxii] (ASPs) – assign a user agent (UA) to each customer registering for a service. The UAs negotiate the customers Service Level Specification (SLSs) with the Network Service Providers (NSP)[xxxiii], represented by a Policy Server (PS). Customers are offered either the desired QoS class (corresponding to a scheduling priority or dropping ration) or merely best-effort service depending on a utility measurement after the SLS-compliant charge is factored in. In common with the earlier analysis of the bandwidth broker, the interaction between entities (UA and PS) lacks the sophistication and flexibility promised by the agent paradigm. Another point to note is that this operates in an MPLS-enabled (ie connection-oriented) network.

Unlike the above work with one agent per user (ie the UA), the work presented in [110] – which offers both immediate and advance reservations – has one reservation agent per network domain. Again, as mentioned earlier this 'agent' is synonymous with the bandwidth broker. Due to the slippage of usage of the term agent it is useful to relocate such example with other projects that also appear or claim to be using agents. The agent/BB queries routers about the status of their links, and is responsible for admission control. Later work evaluated the cost of the reservation system [109]. A punitive overhead identified was the cost of request-reply transactions when using a reliable communication protocol, such as TCP. The network core, ie where providers negotiate QoS contracts with each other, is presented as the most suitable location for advance reservation, unlike the access networks under consideration in this thesis.

[xxxii] third party organisations that provide outsourced services such as VoIP and video conferencing
[xxxiii] usually termed ISPs in related research

4.4.3 Market Based Approach

Many agent models cast their agents as co-operatively working to serve a common framework, for example improving network utilisation. This assumes that the network is a single common resource. In the deregulated telecommunications marketplace such assumptions may prove unrealistic. To reconcile this, market-based approaches have instead modelled self-interested agents, representing competing network owners in a market-based economy. Several market-based paradigms exist that employ an auction protocol/mechanism for allocating calls, for example [133], and in Intelligent Networks (IN) the computational economy model proposed in [134] and [135]. The dependence by the former on a centralised controller or by the latter on a distributor agent or auctioneer in the market models undermines system robustness. Partially to avoid this centralised entity and the resultant vulnerability should this fail a quote-driven market approach has been proposed [136]. A limitation for applicability to connectionless networks is that service providers trade bandwidth associated with a fixed set of source-destination pairs.

In [137] three sets of agents operate: those that sell the network resource (the link agents), those that buy those link resources and sell on these bundled as paths (the paths agents) and lastly those that represent a user, buying the paths (the call agents). Negotiation between agents is mediated via the double auction protocol, conducted at link markets (link agents selling to path agents) and path markets (path agents selling to call agents). As network utilisation rises the marginal utility for resources (ie links) also rises, so the pricing functions are structured accordingly. In the small sample network – consisting of 7 nodes connected with 24 directed links – 150 agents were established: 24 link agents and 126 path agents. As witnessed with the IMPACT project this represents a severe limitation to the scalability of the solution. Furthermore, it is obviously difficult to extrapolate from this to a connectionless system.

In [138] Service Control Points (SCP) form the nexus of service execution in the Intelligent Network (IN) – an overlay network responsible for service provision to the corresponding transport network. If demand (ie service requests to that SCP) exceeds the capacity of the IN the SCP becomes overloaded. To manage this, a load control

mechanism depresses the call acceptance rate at the Service Switching Point (SSP) – through which the telecommunication users access services offered by/in the IN – so that the SCP overload diminishes. A market-oriented programming paradigm [139] is employed to allocate Service Logic (SL) (ie access rights for the incoming load to the SCP) according to SSP demand rates. This creates an economy in which agents trade commodities – ie access to SL – through an auctioneer. When an agent sells an allocation of SL it receives some network money. The agent at the SSP is not endowed with any commodities (but has network money) while the agent representing the SCP has the capacity of the SCP to trade. In the agent architecture the co-ordinator, while enabling the smooth running of auctions, does not function as a centralising point for the auctions. In this respect this agent is neither a bottleneck nor potential vulnerability in the system. Functionality is similar to the Agent Management System (AMS) and the Directory Facilitator (DF) in a FIPA compliant agent platform [140].

The benefits of this approach were tested against Automatic Call Gapping (ACG) algorithm in a network consisting of 8 SSPs, 4 SCPs. The three SL types offered are VPN, a ring-back service and restricted-call forwarding service. Beyond an overloaded level (around 90%) the performance of the ACP diverges from the agent approach and degrades due to oscillations. A high level of revenue is maintained with the novel approach. Yet the flexibility and benefit of agent approach carries increased overhead due to the communication. This possibility could be reduced if a customised implementation rather than a general-purpose platform were employed. The work demonstrated a clear improvement over previous approaches in IN load control that have only one SCP or centralised controller.

4.4.4 Ants

Modelling the foraging behaviour of ants has proved a fruitful area of network routing research, notably [141,142]. This behaviour – termed stigmergy – is characterised by indirect communication through environmental modification, here by depositing pheromones. As ants forage they deposit pheromones, to guide them back to the nest. After finding food the ant returns home, reinforcing the pheromone trail. Food sources

located closer to the nest are reinforced sooner, are stronger (as less pheromone has evaporated) and hence are more likely to be chosen by the other ants. In turn the pheromone is further reinforced and this least-cost path established.

In simulated ant networks a probabilistic (routing) table, representative of modification on the environment, mimics the strength of the pheromone trail. Ant packets investigate and report network topology and performance, altering the routing tables. Two distinct strategies are employed: updating the tables en route (online step-by-step), or once the destination has been reached. Ants will probabilistically select routes with the highest stigmergic reinforcement. Additionally there is a mechanism that simulates the evaporation of the probability-pheromones, and noise is introduced to encourage exploration instead of mere exploitation of the paths.

Shortcomings of this approach include slow convergence in response to network stress, scalability problems and possible sub-optimality due to the localised perspectives of the ants. Moreover whether ants could in practice be implemented in physical networks due to security considerations is questionable. However, this is a very active area of ongoing refinement, for example using genetic algorithms [143] or reinforcement learning with neural nets to dynamically modify ant response speed [144]. The purpose of including this approach is to highlight the issue of agent definition. In the basic AntNet model [145] ants are very simple agents, although they can store internal state, notably past history. Their basic abilities can be augmented, for example to incorporate a simple recovery procedure. Additionally they are disposable – in some models they die on arrival at their destination. Their autonomy is questionable, due to their simplicity. They lack the more sophisticated communication protocols that often are ascribed to agents. Yet they co-operate, via indirect communication. Yet the net result – shortest path routing – is achieved through the ***colony*** of mobile, distributed, active packets, a point reinforced in Dorigo's writing. Furthermore, more recent work in this area [143] has removed *a priori* knowledge (so both routing table structure as well as content is evolved), requiring greater autonomy of the ant-packets.

4.5 Parallel Research

The previous sections have attempted to address how agents operate in networks. Although it may appear that there is a body of work employing agents it was questionable whether some work can justifiably claim to be using agents rather than mere components or entitities. This section further develops this investigation by looking at non-agent-based research and trying to qualify whether this could be termed agent-based. The consequences of this can be interpreted in two ways. One interpretation is that if agent-based and non-agent-based research is indistinguishable in methodology, then the term becomes redundant. Agent, actors, managers, components and entities all blur into the same. However, as has been argued in the previous sections, entities that are NOT agents can be identified. The terminology – agents – may be the same while the praxis has differed. If some projects have been deemed to not be agent-based this is due to methodology/application differences. Clear distinctions can then be drawn between some agent-based and non-agent-based research. Thus the argument of the blurring can be partially refuted as identifiable distinctions operate.

The contrary argument would seek to reinforce common ground between some agent-based and non-agent-based research. Here the focus is on the application and not the title. As quoted earlier the term agent may be comprehendible to a wide audience, but it is not necessary the prevailing term for all disciplines. Flexible, intelligent, distributed management or control is not unique to agent-based research. Where such research shares the same characteristics as other agent-based work it is fruitless to preserve a rigid boundary between agent and non-agent work. Instead the notion of agent-like becomes valuable.

4.5.1 Control Theory

In [89] it was argued that control theory lacked the sophistication associated with agent research. This section provides one example of how recent developments in control theory have overcome such limitations. The intention behind providing an example is that it suggests that there may be a body of work that is agent-like, without being credited as such.

The work in [146] argues that a highly nonlinear system with large uncertainty such as the Internet is unsuited to the mathematical modelling associated with conventional congestion control. Also classic control theory is considered ineffective outside single switch node systems due to the complexity of large-scale networks with multiple parameters. An Active Queue Management (AQM) algorithm with intelligent control, ie knowledge structure, is presented. This Adaptive Optimized Marking (AOM) scheme achieves shorter queue length and drop rate than random early detection (RED) through tuning the trade off between buffer occupancy and link utilisation[xxxiv]. In the model Organisation and Coordination levels are responsible for higher level functions such as planning and intelligent decision making. The expert system forms the machine intelligence in the organisation level. The coordination level translates this to a control pattern for the lower layers. Both levels make qualitative decisions, whereas the execution level makes quantitative decisions, as it has to construct precision control signals. Or, to quote the authors: "*Organization decides what the system is…Coordination decides where to control…Execution decides how to control the system*".

However, it could be argued that intelligent control is equivalent to agent-based control. Certainly it accords with definitions that concentrate on knowledge representation and reasoning. Additionally this AQM system is located within the system, unlike the classic knowledge or expert systems that are disembodied. Müller's analysis of the parent disciplines of intelligent agent design perhaps is the most pertinent for this analysis [86]. The controller process is considered analogous to an agent. Where the analogy breaks down, Müller argues, is in the complexity of most environments, which are not amenable to traditional solutions by differential equations associated with control theory. Likewise control theory is associated with an inability to manipulate incomplete and inconsistent information. However, the aim of the researchers here is to explicitly move away from the classical approach and hence the major obstacle to an agent definition is removed. As has been state earlier, with no authoritative definition of an agent, the presence of knowledge structures will

[xxxiv] subject to the assumption that IP networks exhibit stationary or slow changing traffic distributions

not satisfy all agent researchers – Brook's emergent intelligence model for example would reject such constructs. However, this control theory model would accord with many other agent examples, including some delineated earlier.

4.5.2 Policy Based Management

This final section provides an introduction to projects investigating policy based management, a growing area of means of automating network management through high-level directives [147]. Here policy is taken to mean "*the unified regulation of access to network resources and services based on administrative criteria"* [148]. Section 0 stated that the work in this project focused on the control, as opposed to data and management, plane. However, in order to fully qualify the role of agents in connectionless networks it is valuable to investigate developments in the management plane. These enhancements extend the bandwidth broker concept, and as already stated it would be overly generous to term that entity an agent. Additionally, policy based management usually does not profess to identify the components in the architecture as 'agents'[xxxv]. However, there are many features underlying policy based management – distributed sophisticated management and monitoring, communication protocol – which would appear to demonstrate the flexibility associated with agents.

The IETF states that a Policy Based Management System (PBM) should enforce differing levels of QoS guarantees for both users and applications, via policy rules [149]. These rules govern admission control, scheduling, traffic shaping for various users under varying traffic conditions. Parameters for the rules include a range of QoS metrics such a requested bandwidth, jitter or starting times. These systems are set up as two-tiered applications: for final policy decision, the policy manager (or policy server) at the top; the edge or boundary routers[xxxvi], for policy enforcement, at the lower layer.

[xxxv] Although see AQUILA project
[xxxvi] Additionally there is an LDAP server which stores the policy rules

4.5.2.1 Policy Projects

The AQUILA[xxxvii] project implemented an architecture for end-to-end QoS provisioning in the Internet. The core network is DiffServ enabled and over this lies an overlay network – the Resource Control Layer (RCL). This layer performs resource control (monitoring, controlling and distributing resources) via the Resource Control Agent (RCA). Significantly it has been stated that: "*An RCA is a generalisation of the concept of the Bandwidth Broker in the DiffServ architecture"*. Additionally another 'agent' – the Admission Control Agent - in this layer, linked to each ingress/egress router, is responsible for both policy and admission control. Finally, this layer acts as in interface to the QoS for the End-user Application Toolkit (EAT). The EAT middleware operates at the control plane and is responsible for QoS reservations. Inter-domain there is a Border Gateway Routing Protocol (BGRP) Agent[xxxviii] at each border router that aggregates reservations for the same destination. Discussion about the agent-like qualities will wait till all three projects are introduced.

The Cadenus[xxxix] project investigates automated service delivery by providers, through dynamically negotiated SLAs [150]. The aim is to translate (ie automate) an SLA, as specified by an end user, into an SLS, which describes the technical details of network specification. It is argued that the use of an SLS automates service activation in IP networks (whereas a user's QoS request would be carried as a signal under the telecommunications model). The project operates with a longer-term dynamic QoS perspective than AQUILA (there is nevertheless acknowledged overlap with all three projects) and additionally does not investigate inter-domain QoS. The Cadenus architecture partitions the system into 'Mediators', which map user's QoS requests to the corresponding service/network resources. This clearly demarcates service both from resource control and management and from the service creation machinery.

The Access Mediator (AM) interacts with the user – establishing best-fit services – and the service providers – negotiating dynamic SLA features. The Service Mediator (SM) both incorporates new services to the Service Directory as well as managing the

[xxxvii] **A**daptive Resource Control for **Q**oS **U**sing an **I**P-based **L**ayered **A**rchitecture
[xxxviii] This is still only in framework
[xxxix] **C**reation **a**nd **D**eployment of **En**d-**U**ser **S**ervices in Premium IP Networks

physical access to the requested services (employing the Resource Mediator, RM). Additionally the SM is responsible for preparing the user's SLA and translating this into the SLS. The above mediators advertise their existence to each other via the Service Directory: whereby the SM is the seller and the AM is the buyer of the advertised services. There is only one Resource Mediator within an AS, and additionally one Network Controller for each network technology[xl] within that domain. Communication between the RM and the network is based on COPs-like policy rules. The mediators employ the 'Active Object Model'.

The demarcation of service treatment (carried out by the SM) and the resource treatment (carried out by the RM, whose role is to translate service demands into specific network resources demands) differs from a standard SLS definition, since this usually defines scope (ie ingress and egress node). Thus a new type of SLS is identified so that the separation is not violated. The traditional offline SLA is identified as suitable for subscription and provisioning but not for the usage (call-by-call) process. So CADENUS considers an invocation or i-SLA/i-SLS. The i-SLA just contains the service class to distinguish QoS levels, since all the other parts of the contract have been negotiated previously in the SLA subscription / provisioning process.

The TEQUILA project [151] is concerned with longer-term traffic engineering than the other two projects. It investigates QoS provision in IP networks through SLS negotiation, monitoring and enforcement, intra-domain traffic engineering and inter-domain SLS negotiation. The focus is on service management, ie defining services and service classes (service creation), the negotiation and subscription to services and service assurance. The framework consists of two time frames or epochs: the longer term service subscription – where customers subscribe for future services – and the more immediate service invocation for per-call requests – ie where customers invoke the services to which they have subscribed. This echoes the resource management timing: off-line network dimensioning and dynamic route management. Route

[xl] ie one for ADSL 'technology domain', one for DiffServ td…one for MPLS

selection is made in a distributed fashion, but the cost metric used to calculate the paths are manipulated by the network dimensioning component.

The TEQUILA architecture is hybrid: the network-dimensioning element – responsible for mapping traffic requirements to the physical network – is centralised, while other network management elements are distributed to the nodes (either just the edge routers or to all routers) and are reactive. Additionally the high-level Policy Management Tool is centralised while the Policy Repository can be distributed. After storing policies in the repository activation information is passed to relevant Policy Consumer for retrieval and enforcement. The centralised Network Dimensioning (ND) maps traffic requirements to physical network resources and provides Network Dimensioning Directives – such as definitions of label switched paths (LSPs), anticipated loading of per-hop behaviours (PHBs) – to accommodate predicted traffic demands. The lower traffic engineering elements – Dynamic Route Management (DRtM, edge routers only, manages parameters for selecting LSPs) and Dynamic Resource Management (DRsM, all routers, manages buffer & scheduling parameters) – manage resources allocated by ND. For example, the DRsM would translate anticipated PHB loading into scheduling parameters. Provisioning thus incorporates long-term SLS and dynamic network state. In addition to producing the guidelines for sharing network resources, the ND is also policy-influenced from above. Example policies include: how often to trigger dimensioning; importance of a particular PHB; maximum number of alternative paths; parameter specifying the relative merit of low overall cost against network overload avoidance.

TEQUILA's system objectives are both traffic (ie obligations to customers via SLS) and resource-oriented (network optimality). The design requirements also incorporate avoiding overloading parts of the network and providing overall low network load. To avoid network hot-spots, instead of employing standard routing algorithms the ND employs a version of a k-shortest path algorithm. This finds paths subject to the cost and utilisation constraints. These two constraints lead to conflicting optimisation objectives and a non-linear optimisation problem [152].

The EURESCOM project P1008 [153] identified a need for both the traditional long-term service contract as well as a novel, dynamic, short-term contract. TEQUILA (as well as CADENUS) provides both features – SLS subscription (SLS-S), concerned with long-term policy-based admission and SLS invocation (SLS-I), a more dynamic component, which dynamically deals with each flow.

In the TEQUILA architecture the principal reasoning – policy management and network dimensioning – is centralised. This not only makes the architecture potentially more vulnerable, in the example of network failure, but also introduces higher signalling overhead. Additionally the architecture is committed to a Diffserv/MPLS-based network.

4.5.2.2 Common Open Policy Service Protocol (COPS)

Common Open Policy Service Protocol is a client server protocol that defines communication messages between two operating entities, the policy decision point (PDP) and the policy enforcement point (PEP) [154]. In the policy based management architecture the PDP is located in the policy server, while the PEP is located at the edge/boundary routers. COPS can operate in an outsourcing mode, whereby a PEP receives a request for a connection servicing. The PEP then passes this up to its allocated PDP, which has to obtain the relevant policy rules from the LDAP server. Using these, an assessment is made by the PDP whether to accept or reject the connection. This decision is then passed back down to the requesting PEP, which in turn enforces the policy. By contrast, in the provisioning model the updates are found at the LDAP server, without the prompt caused by a connection request. Any policy changes are then enforced. COPS is considered to be a flexible protocol that is adaptable to other protocols. However, there are scalability questions due to both the limited number of PEPs supported by one PDP and the constraint on a PEP only connecting to one PDP. Additional concerns include inter-vendor COPS operation and support of legacy routers.

Unified Policy-Based Management (UPBM) has been proposed to ameliorate some of the problems with COPS in policy based management architectures [149]. This three-

tiered architecture adds network routers[xli] alongside the edge routers at the bottom of the hierarchy and also includes a middle tier: the policy enforcement agent (PEA). This translates different policy rules due to the relaxation of the tight coupling between PEPs and PDPs. Additionally PEAs act as intermediaries, providing COPS and content translation. This means the PEPs can now be non-COPS compliant, for example with legacy routers. When a PEA interacts with a new router it can use inter-PEA communication where repository is non-sharable.

4.5.2.3 Challenging the Demarcation

It could be contended that the distinctions made in the earlier sections were somewhat arbitrary, reflecting the prejudice of personal research. Thus, it would be expected, researchers schooled in the Wooldridge / Jennings approach would focus on multi-agent co-operation and reject less-collaborative AI-heavy agent models. Likewise, those favouring hierarchical decomposition may favour models with reactive agents and higher-level monitoring agents. Equally, although multi-agent systems usually stress decentralisation, in practice this is not always followed: for example, centralisation – of reservation state and SLAs – is found in the agent-based work of [155]. This could reverse much of the analysis of previous sections by arguing that there exists a growing area of what could be called agent-like applications. Certainly two of the projects mentioned used the term 'agent' (although as mentioned for one this was related to the bandwidth broker).

The purpose of the section on policy based management has been to challenge assumptions about agents – perhaps both their flexibility and architecture can be found elsewhere, so the agent / not-agent distinction is increasingly redundant. It can conversely be argued that by building up from the bandwidth broker concept, the interaction between the entities or 'agents' in policy based management has been constrained. As discussed in the section on protocol agents the manager-managed relationship is too tightly circumscribed. Autonomy, while difficult (see the following section), underlines agent systems. Where this may be difficult then the AI model offers a partial solution, at least of agent-like behaviour.

[xli] core routers are not controlled as part of standard PBM

If it is conceded that these are not agent architectures, due to absence of AI and limited decentralisation, then the question 'why use agents?' is raised. In the SHUFFLE project cited earlier, the choice/overhead of the agent middleware was identified as a critical limitation. COPS, the middleware here would seem to be less problematic, with ongoing research addressing its weaknesses. Agents are not the only solution and it is important to locate them next to similar research.

4.6 Summary: the role for agents

While it is not novel to address the difference between network management agents and intelligent agents deployed in networks – see [156] for example – highlighting the tensions between the two models has aided an analysis on how successfully (or not) intelligent agents have penetrated telecommunications networks. If protocol agents provide such a weak model of agency, and if policy based management provides a flexible notion of control, the role of the agent may appear weak. Perhaps an analysis that stresses common ground with non-agent research and at the same more clearly demarcates agent roles will allow agent enhancements to flourish.

Although the multi-agent system has been identified as pertinent to distributed management of complex networks due to distribution and flexibility this has proved more problematic in connectionless networks. While MPLS has provided a connection-oriented bridge to a connectionless world, evidence has been presented [70] for not requiring this technological upgrade. This then leaves the question of the suitability (as well as practicality) of agents in IP networks. Autonomy is the critical characteristic of an agent. OSPF relies on tight coupling between nodes that undermines this vital trait. This would appear to inhibit the successful deployment of agents. The analysis of agent deployment largely concurs with this. However, perhaps extracting elements from agent projects – without reducing them to the hollow model characterised by protocol agents or the non-agent agents described by Wooldridge / Jennings – can point to a successful use of agents. Instead, a more fragile agent-like model is proposed. This currently lacks the full complexity of agent communication, although certainly this could be added, alternatively taking advantage of the range of

protocol traffic available. Agents are fully distributed to nodes, unlike bandwidth brokers or the manager-agent model. However, unlike the Wooldridge / Jennings model the focus is on agent learning, notably reinforcement learning.

5 Sub-Optimal Routing

Chapter 3 examined various understandings of QoS. The research presented in this, and the subsequent sections, focuses on routing enhancements that spread traffic away from the optimal paths offered by OSPF. This in turn increases network utilisation, with the aim of evening the distribution of traffic load within the network, satisfying both QoS and resource management goals.

In all the novel enhancements presented in this thesis, end-to-end delay is used as the QoS metric. While acknowledging that more recent work has attempted to model multi-dimensional QoS [157], most comparative research avoids introducing such complexity and focuses on a single QoS dimension. Alternative deployments [79] have used the Bellman-Ford algorithm, which supports two metrics – since OSPF uses Dijkstra's algorithm, which supports only one metric, this is beyond the scope of this research.

Much QoS routing research has employed bandwidth (also termed load or utilisation) as the critical metric, for example [158,159,71,1]. With this bottleneck metric the path weight is that of its worst link (ie with lowest bandwidth). However, end-to-end delay is of significant concern for time-sensitive traffic and is receiving increasing focus [160]. Delay is seen by the ITU as one of the key parameters that affect the user. Indeed to the user, delay incorporates the effect of other parameters such as throughput [23]. For such an additive metric the path weight is represented by the sum of the weights of its links.

5.1 Pseudo Delay Mechanism

At times of network stress it is imperative to the network operator that premium traffic is not impeded by less valuable traffic. In a multi-class environment the routing system should spread this less valuable traffic away from the optimum routes so that it no longer affects the performance of gold traffic. The pseudo-delay mechanism

introduced here is designed to engineer this by masquerading longer or otherwise 'costlier' routes as optimal. By providing pseudo-delay cost metrics, generated from observed network delay, the router can still employ the standard Dijkstra shortest path algorithm, to produce new 'shortest' paths.

The concept of selecting a sub-optimal path is not in itself novel [161]. However, such work has often sought to address the issue of inaccurate link state information that arises due to the impracticality of continuously flooding latest costs. Here, by contrast inaccuracy is deliberately injected into the costs in order to encourage traffic down sub-optimal routes. This may appear to conflict with a standard objective of traffic engineering to optimise IP network performance [162]. However, it can be argued that the resource oriented performance objectives of traffic engineering set out in [14] will be addressed by allowing traffic to follow sub-optimal routes. Additionally much of the existing work has been carried out in connection-oriented networks for the CR-LDP and RSVP-TE resource reservation schemes.

Traffic is generated in the network using an ON/OFF model. Routing is implemented using OSPF with delay as the cost metric. At system initialisation the cost of each hop is set to the Cisco default. These initial values are modified by delay figures as the simulation develops. However, only gold traffic is routed according to observed delay, using exponential weighted moving averages:

$$Ad^t = \alpha Od + (1-\alpha)Ad^{t-1} \qquad \textbf{(Equation 1)}$$

where Ad^t is weighted moving average delay at time t, Od is the observed delay and α is a constant. The cost metrics for silver and bronze (ie best-effort) traffic are modified by two factors – termed theta_1 and theta_2. These are generated such that when the OSPF cost metric is modified by the thetas they display a more severe delay figure for such traffic. The routing table for lower preference traffic is constructed consequently from data based on these pseudo-delay figures. A hop that is on the optimal path for the gold traffic would now be more costly so would not necessarily be included in the apparently 'least cost' path for the lower grade traffic. The new least cost path is now a 'sub-optimal' path presented as optimal. No alteration to the

routing algorithm is required. As network congestion increases this mechanism thus moves lower class traffic away from the optimal paths, where it may affect the performance of gold traffic.

As stated above, the justification for employing sub-optimal routing is by observing that were lower class traffic routed along optimal paths it may receive less preferential treatment. One means to differentiate between traffic types would be to employ strict priority scheduling at each router. At times of network stress low-grade traffic could be starved at a router while preferential, delay-sensitive traffic is serviced. Alternatively, by routing away from these paths, the low-grade traffic is no longer delayed or starved at "hot spots".

If the packet delay between the two nodes is above a critical threshold, this observed delay is used to modify the thetas for that link. Critical thresholds are set for each service class to reflect delay-tolerance. As well as behaving reactively the system also displays proactive behaviour. For example, if the critical figure has not been reached but the traffic has been monotonically increasing (or decreasing) and crossed a lower 'trigger' threshold (again set separately for each traffic type) the system registers this trend. Unlike the threshold trigger approach investigated in [72], more precise information is obtained by setting the trend trigger higher, ie confirming a trend after a longer period. A modifier is then calculated to depress the new theta value/s. This approach allows the system to monitor eg network congestion and anticipate such problems. By depressing the theta calculation with a modifier, the response will be lesser than if the critical threshold is passed. Equally, the system responds to downward shifts in delay – ie as congestion decreases a downward delay trend is identified and the thetas accordingly altered.

The thetas are calculated using both the exponential weighted moving average delay and the exponential weighted moving average of the differences in observed delay. The exponential weighted moving average for the differences (ΔAdt) is calculated as:

$$\Delta Ad^{t}=\beta(Od^{t}-Od^{t-1}) + (1-\beta)(\Delta Ad^{t-1}) \qquad \textbf{(Equation 2)}$$

to reflect both performance shift and the rate of this shift in the calculation. If responding to an observed trend that crosses the trigger threshold, but is below the critical threshold theta_1 is calculated using a modifier to produce the following equation:

$$\theta_1 = \theta_1^{t-x}(Ad^t * \gamma(1 + \Delta Ad^t)), \quad \textbf{(Equation 3)}$$

where t-x is the time of the last theta revision. The value of γ is adjusted to reflect the scaling down of the simulation discussed in section 5.1.1 (ie affecting the default link cost).

Modifications to the link cost metric in the link state database ensure that this metric is never lower than the original figure (ie the Cisco default). Theta_1 is used to manipulate the delay metric for silver traffic and a combination of the two thetas is used to affect the delay metric for bronze traffic. These figures are then flooded as an LSA to all other network nodes to advise them of the shift in network state.

5.1.1 Results

Early experiments demonstrated that the pseudo-delay mechanism rerouted lower status traffic away from optimal paths. This ensured that preferential traffic was not impeded at times of network stress. Rerouting onto the sub-optimal paths ensured that the lower grade traffic received better treatment; had such traffic shared the same paths as gold traffic it would have suffered at times of congestion. However, oscillatory behaviour, especially of the bronze, best-effort traffic was observed. This traffic was the first to be rerouted away from the optimal paths, but as delay increased on the sub-optimal paths again it would be rerouted. To avoid this flux, together with the frequent LSA generation and processing overhead, further dampeners were introduced into the system. Additionally, to make the simulation more malleable, variables such as transmission speed and packet generation rate (and consequently the critical and trigger threshold figures that activate updates) were scaled down, as mentioned in the simulation section. As a result the findings from the simulations should be read as relative, as absolute delay figures have to reflect this scaling down.

These earlier experiments, additionally, employed the default OPNET RNG (ie that used by Microsoft Visual Studio for simulations run in the Windows environment).

Figure 9 shows network conditions prior to employing the pseudo-delay mechanism. Each graph shows averaged end-to-end delay over 11 runs, with 95% confidence intervals. Although the gold traffic experiences relatively high delay (ie above 150 milliseconds) this can be explained by the scaling down of network parameters. Additionally all traffic classes experience an initial traffic surge in the empty network. As buffer build-out rises, however, performance stabilises for gold and silver traffic. By contrast the end-to-end delay increases for the best-effort traffic. The performance of this traffic is both markedly lower and volatile due to the combination of bursty nature of the traffic combined with the less generous treatment by the scheduler. The poor performance in the buffers is perhaps exacerbated by the relatively high volume of gold traffic (50%) in these simulations[xlii].

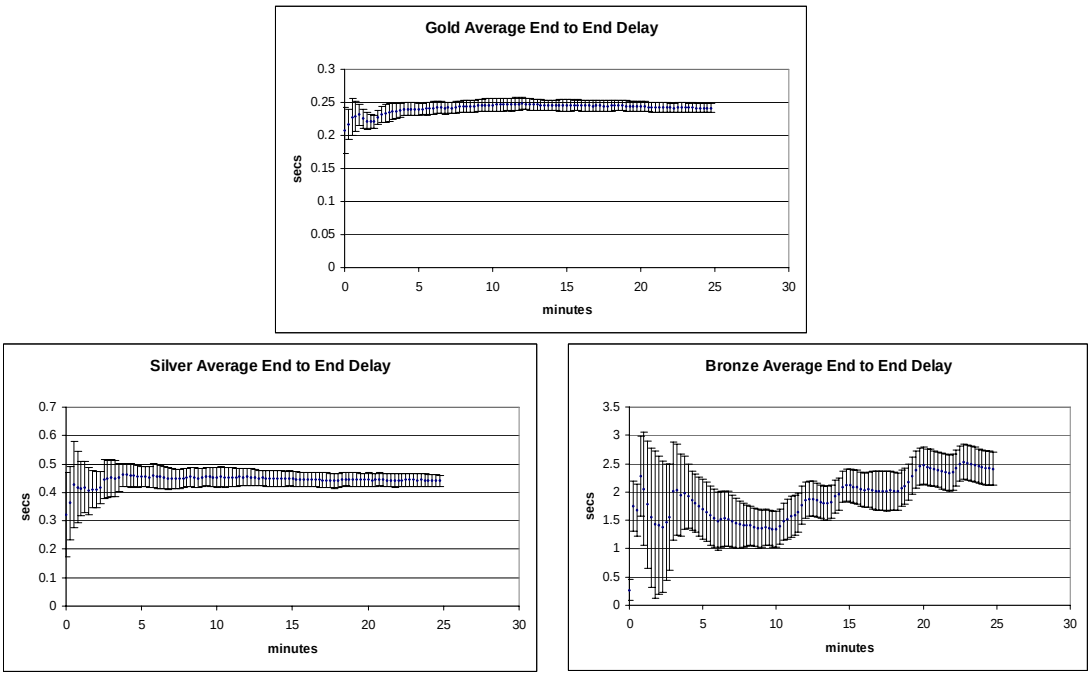


Figure 9: Routing without the Pseudo-Delay Mechanism

[xlii] The simulations results in section 8 are across networks with a lower volume of premium traffic

Figure 10 shows sample results of routing applying the pseudo-delay mechanism. For clarity confidence intervals have been omitted. Here the value of gamma (γ) in the modifier (equation 3) is set to 1000 and the value of the theta factors in critical conditions is based on both observed and average delay.

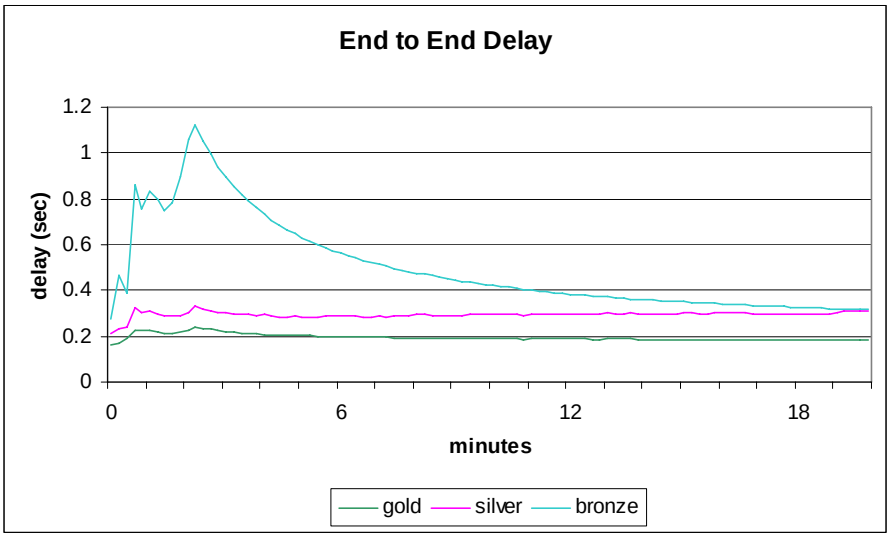


Figure 10: Routing with the Pseudo-Delay Mechanism

As the pseudo-delay mechanism is introduced, perturbations are still observed initially for all traffic, most significantly bronze, but this disappears and the bronze end-to-end figure rapidly decreases to around a quarter of the peak value. Various adaptations have been explored for calculating modifiers, thetas and thresholds. An example is shown in Figure 11 a, b & c (with 95% confidence intervals). By changing the critical update theta calculation to $\varepsilon(1+Ad^t)$, where ε is a constant, and shifting the ratio of theta_1 and theta_2 that generates the link cost for bronze traffic there are reductions in degree of perturbation and both peak and mean end-to-end delay for all classes. Obviously it is hard to quantify the benefit in a real network because of the scaling used, but there is reason to believe the force of the result still applies. The gold end-to-end delay mean is increasingly decreasing towards the critical 150 milliseconds figure in a slow network. However, the results demonstrate that by manipulating the delay figures for the lower class traffic the performance of all traffic is improved.

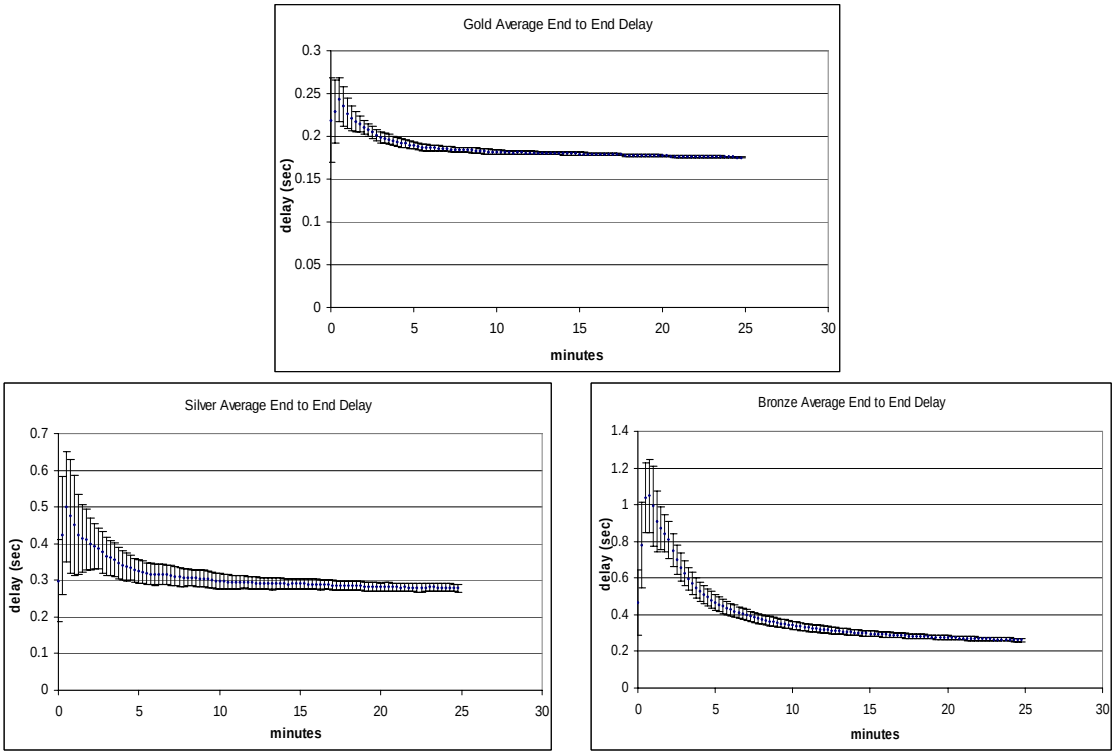


Figure 11: Routing With the Enhanced Pseudo-Delay Mechanism

Although sub-optimal routing is employed, an optimisation aim of traffic engineering is to increase throughput, and by extension network utilisation. In the early experiments while throughput for gold traffic remained constant, that for silver and bronze increased by 31% and 36%, respectively. In the modified experiments, with a higher percentage of gold traffic in the system to place extra stress on all classes, bronze traffic is the main beneficiary with a 40% increase in throughput. Throughput for the other classes increased slightly, but not significantly.

6 Learning

The previous section introduced a heuristic for spreading traffic away from the 'optimal'[xliii], congested links. Despite incorporating a mechanism for incorporating trend, the setting of threshold figures appeared arbitrary and rigid. A shortcoming of the algorithm is that delay only registers as critical once it reaches a precise, preset figure, yet is not treated as critical if it is eg a millisecond below this figure. A more responsive approach to trigger system responsiveness is outlined below – using principles of learning incorporated with fuzzy logic to ameliorate the inflexibility of the flooding triggers.

A characteristic of several agent systems is the capacity of the agent to learn from its interaction with the environment. Although not all agent definitions incorporate learning, this property can aid an agent in responding more appropriately to a dynamic environment. The model developed here seeks both to discover when to flood and how high to set the theta factor. Additionally, instead of devising a communication strategy to propagate the learning, the model piggybacks on the existing, albeit limited protocol-based, communication between nodes provided by the routing protocol (ie OSPF). Although there is a loss of the sophistication usually demanded of an agent communication language (ACL), this removes the requirement of deploying agent middleware (with the attendant overhead, including devising ontologies etc, discussed in section 4.4.1).

A disadvantage of many machine learning techniques is that a complete model of the problem domain has to be predefined. In most machine learning a supervisor knowledge base provides examples that guide the learning. It may, however, be impractical or impossible to produce models of appropriate behaviour for all situations that an agent encounters. Instead it may be more fruitful for an agent to learn from interacting with the environment, ie by its own experience rather than guided by a supervisory knowledge base. Although dynamic programming can be

[xliii] Remembering that the 'optimal' route may be considered selfish and thus not optimise network performance

used to solve reinforcement learning techniques, the requirement of a thorough, precise environment model, rather than one gleaned through discovery.

By contrast reinforcement learning has proved attractive as the programmer does not have to define a vast set of conditions, instead learning entirely through the feedback resultant from acting on the environment. Since everything it learns has to derive from such interplay, a reinforcement learning agent is characterised by having to balance exploitation with exploration. As a rational agent is seeks to maximise its goal so it should select the most productive action. This will be one that has provided the highest reward in the past – thus the agent is exploiting its gained knowledge. However, in order to discover potentially more valuable actions the agent also needs to explore the environment, ie according to a set policy it should occasionally not follow what presents as the optimal action and instead choose an alternative action.

However, a disadvantage of such techniques is that the state space is prone to dimensional explosion. As the problem space explored by the agent grows there is a corresponding escalation in the agent's state space. Fuzzy logic or fuzzy set theory has been demonstrated as a solution to resolving the state space expansion in reinforcement learning [163]. Instead of having to store values for each state observed by the agent these can be graded by membership of fuzzy states, thus reducing the complexity of the state space. Thus fuzzy reinforcement learning has been chosen to add intelligence (agent behaviour) to each node, while according with the IP aims of limiting state.

6.1 Fuzzy Reinforcement Learning

Learning is used in this simulation to try to discover whether or not to flood an SLA, ie generate a more sensitive responsiveness to network delay. While flooding will result in routing tables that more accurately reflect the network state, a negative consequence is the time required for network convergence. Additionally, another parameter receptive to learning is the factor used to spread lower class traffic away from optimal routes, ie the theta value to add to the true cost (delay) for the lower class traffic. This section first investigates reinforcement learning and outlines its

suitability for solving these problems. An introduction to fuzzy logic is then presented, finally combining the two into the selected fuzzy reinforcement learning model to show how the flooding and theta decisions can be learned by the agent.

6.1.1 Reinforcement Learning

Reinforcement learning has been chosen as it can learn directly from the dynamics of the environment. No prior knowledge of the environment is required and there is no need for training and modelling decisions. In order for the agent to learn, evaluative feedback is employed to indicate the success of an action. This is in contrast to an instructive feedback model where a new action would be chosen independently of the previous one.

A reinforcement learning system is primarily composed of a policy, reward function and value function. The policy corresponds to the action choice in response to the perceived environment state. The policy is essentially equivalent to the definition of an agent's behaviour, ie a mapping from percept to action. This corresponds to the (reactive) agent (*Ag*) function [97]:

$$Ag : E \rightarrow Ac \qquad \textbf{(Equation 4)}$$

where the agent (*Ag*) is the function mapping the environment (*E*) state to an action (*Ac*). Although a rational agent is considered a goal-maximiser, reinforcement learning will not necessarily choose the greedy action. Instead the policy should balance exploitation (ie acting on what is already known) with exploration (ie randomly searching or choosing an action). An ε greedy approach is a common policy employed in reinforcement learning. In the exploitation phase the action selected has the highest strength, or returns the highest reward. However, this chosen action is not necessarily the one that is performed; the action selection mechanism is set to randomly explore, ie flood, with a small probability[xliv] of ε.

The process can be considered as a run, ie a sequence of episodes, where an episode consists of a state, action selection and the resultant state. Figure 12 shows a model of

[xliv] for example, the value of ε used in the simulations is 0.001

a run consisting of episodes, moving from one state to the next. At state s_t, at time t, action a_t is chosen. More formally, $a_t = \pi(s_t)$, where π is the policy (eg ε greedy) at s_t. This generates reward r_{t+1} and returns the new state s_{t+1}, ie a Markov chain with a reward process.

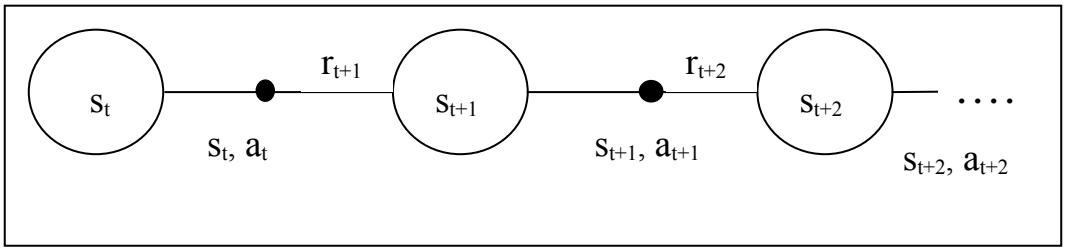


Figure 12: Episodes of states and state-action pairs[xlv]

The reward function returns the immediate desirability of the state (or state-action pair) for the agent. The routing protocol in IP networks is designed to be quiet – only responding by flooding LSAs when necessary – so it is generally more desirable not to flood. Thus the reward for not flooding is set to be higher on average than that returned from a flood. Although an agent cannot alter the reward function, this function can be used to affect the policy, ie future action selections in a given state.

The reinforcement value function by contrast corresponds to an estimation of the longer-term value of each state (or state-action pair). Unlike rewards, which are directly provided by the environment, the value of a state (or state-action pair) can only be estimated gradually by the agent as it interacts with the environment. The value corresponds to the totality of the rewards over the future from that state. Although some states (or state-action pairs) may offer a low reward (ie immediate feedback) the states that follow that choice may generate high rewards, so a greater long-term value is accrued. Thus a flooding decision may produce a lower immediate reward, but result in lowered congestion, yielding an elevated long-term value.

[xlv] From Sutton, Barto op cit page 145

The value functions or judgements, since they are estimates, must be reified throughout a run or simulation. Two prominent update approaches exist for reinforcement learning problems: Monte Carlo and temporal differences. In the former value estimates are only reinforced at the end of a run. A Monte Carlo method demands that a run terminates, so that feedback can be provided solely on completion of and not during that run.

A temporal difference (TD) approach has been chosen in this research reinforcing value estimates after the next step. Unlike Monte Carlo methods, temporal difference methods (in common with dynamic programming) are characterised by bootstrapping, ie updating estimated values with other estimates. The step-by-step temporal difference approach – where changes to the value estimate are based on a difference between estimates at two different times – can be generalised by the following update rule:

$$V(s_t) \leftarrow V(s_t) + \alpha[V(s_{t+1}) - V(s_t)] \quad \textbf{(Equation 5)}$$

where $V(s_t)$ is the estimated value of state s at time t and α is a learning parameter[xlvi]. Thus the estimate of a state's value at time t is updated based on that estimate plus difference between the estimates of that state at two distinct time steps (hence, temporal difference). A factor, λ, is used to indicate how many preceding temporal states are to be updated – in this research the simplest case is used where λ is set to 0, ie TD(0), so only the preceding state is updated (ie s_t is updated by estimates of s_{t+1}). This minimal TD method can be represented by:

$$V(s_t) \leftarrow V(s_t) + \alpha[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)] \quad \textbf{(Equation 6)}$$

where r_{t+1} is the reward obtained for moving to state s_{t+1} and γ is a discount factor.

Other techniques – including simulated annealing and genetic algorithms – termed 'evolutionary methods' and differentiated from reinforcement learning techniques in [91], can also be used to solve reinforcement learning problems. Unlike the approach investigated here, value functions are not employed in evolutionary methods, so

[xlvi] Also known as the step-size parameter. If this is set to zero no learning (ie revision of values) takes place; as it reaches one learning takes place at a faster rate

individual states, or state-action pairs, are not estimated. Actions chosen inside a run are not registered. Thus moves that may have contributed significantly to the success of the final outcome are weighted equally with those that may have had a negative or neutral impact.

A shortcoming of reinforcement learning is the expansion of state space required for an agent's reasoning. Techniques deployed to control the scalability limitation of the associated large look-up table – both the space required for storing and the speed of information access – have included neural networks and self-organizing maps. However, since both these techniques and reinforcement learning itself are characterised by slow learning rates these are not feasible solutions in busy network environments. As seen in the previous section, employing fuzzy states and actions reduces state space, as a vast range of crisp states corresponds to a greatly reduced range of fuzzy states, with faster convergence. Thus combining fuzzy tools with reinforcement learning, ie fuzzy reinforcement learning, may be a feasible given the constraints associated with the network environments under investigation.

6.1.1.1 On-Policy and Off-Policy Learning

An important way of differentiating between the various temporal difference methods is whether they are on-policy or off-policy learners. On-policy learners evaluate and improve the value of a policy while using it for behaviour control; off-policy learners separate the policy used to generate behaviour from that which is being evaluated, ie evaluate one policy while following another. A consequence of this latter approach is that the system can learn about policies that are never followed. Q-Learning [164] is an example of an off-policy temporal difference control algorithm. This updates a state action pair based on the maximum reward achievable from the next state-action pairing:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\left[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)\right] \quad \textbf{(Equation 7)}$$

where $Q(s_t, a_t)$ is the value of the state action pair taken at time t, α is the learning factor and γ a discount factor. This can be shown in a backup diagram[xlvii] as:

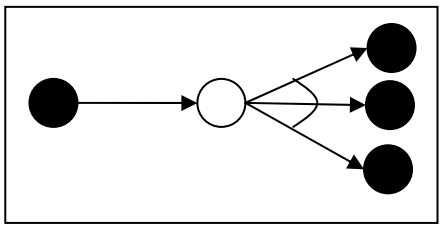


Figure 13: Q-Learning Backup Diagram

where the filled circles indicate action nodes, the white circle a state node and the arc that the maximum of the action nodes will be taken. Thus the first action (leftmost filled circle) – corresponding to a_t of state action pairing (s_t, a_t) – results in the new state (s_{t+1}), the white circle. From this the action that would return the maximum reward would be chosen to reinforce the value $Q(s_t, a_t)$. The arc indicates that the maximum of the next action nodes is taken. If the topmost action taken from s_{t+1} were predicted to return the highest reward this would be included in the update function. However, due to the need to maintain sufficient exploration this may not be the action selected by the policy at that next state.

By contrast the on-policy approach evaluates only the policy being followed – ie the policy is enhanced solely using estimated values for the current policy. Sarsa[xlviii], an on-policy approach, learns the value of state-action pairs from transitions from state-action pair to state-action pair. The notable difference from the Q-Learning equation is that the maximum operator is discarded and replaced with the value of the next (followed) state-action pair:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\left[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right] \quad \textbf{(Equation 8)}$$

With Sarsa, unlike the procedure in Q-Learning, if an ε-greedy policy is applied, this value of ε is included in the Q update. Thus the best policy given the systematic departures (exploration), is learned under Sarsa; the best policy learnt under Q-

[xlvii] Although shown horizontally for consistency with other diagrams, standard backup diagrams are drawn vertically

[xlviii] The name is derived from the state-action transition quintuple: **S**tate, **A**ction, **R**eward, **S**tate, **A**ction

Learning does not incorporate this exploration so explicitly. A result of this is that the cost of exploration is factored into the on-policy learning and the system can avoid more disadvantageous outcomes [165].

For this research a fuzzy Sarsa, ie on-policy, algorithm was chosen. This approach has been demonstrated to provide robust and accurate results with a significantly smaller state space than the corresponding non-fuzzy model [166]. The model used is explained fully in section 6.1.3.

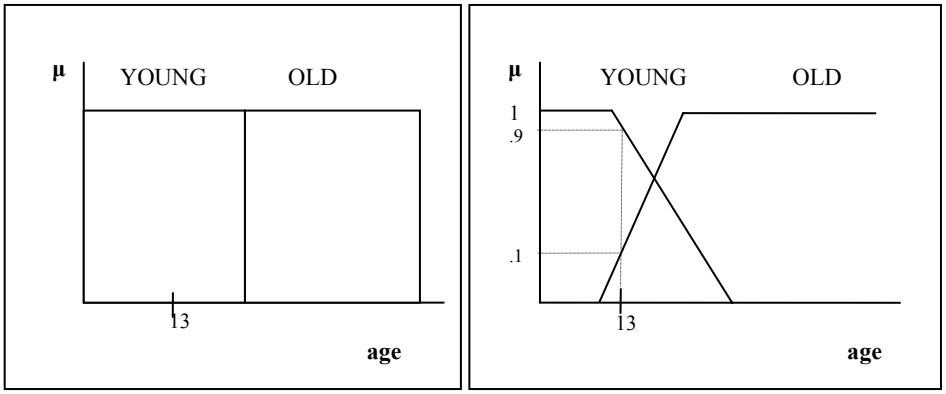
6.1.2 Fuzzy Logic Control

It is complex to construct a precise mathematical model for all the variables – whether triggering thresholds, or theta parameters – in the system. Where a formal analytical model cannot be used – ie rigorous theoretical approaches are inapplicable – fuzzy logic can prove a valuable tool [167, 168]. Fuzzy logic control has been employed to solve various network challenges, including active queue management schemes in IP networks for congestion control [169], call admission control [170] and routing in connection-oriented networks [171, 172]. An overview of its applicability to QoS management is provided in [173]; a comparison to other techniques used to handle uncertainty is provided in [174].

Fuzzy set theory/logic considers degrees of belonging to a set as opposed to classic (Boolean) set theory where an element is either a member or not a member of a given set. Instead of truth of membership being represented either by 1 (member) or 0 (not member), in fuzzy logic truth values lie in the range [0,1]. As an example, when considering variables such as age, temperature or bandwidth, a classic approach would create discrete sets (or intervals), for example young/old, cold/hot, empty/congested, where "young = ¬ old" etc. The truth of each state is an either-or membership statement: for example, in a world comprising people of all ages, for the element 'age 13 years' the truth of being young is 1 while the truth of being old is 0. By contrast, since fuzzy logic is based on truth values in the range [0,1] rather than just 0 and 1, the element 'age 13 years' may belong to fuzzy set YOUNG with membership degree 0.9 and belong to fuzzy set OLD with membership degree 0.1. As

a result the transition from membership of YOUNG to membership of OLD is more gradual than the abrupt jump from member to not-member in the traditional rigid (true/false) model.

Figure 14: Classic (interval-based) (a) and Fuzzy (b) Membership



(a) **(b)**

Fuzzy logic is used in fuzzy controllers to simulate human thinking. A fuzzy set is a mapping of real numbers (such as ages {24, 25, 35, 37, 82}) to a set of symbolic labels (YOUNG, MIDDLE-AGED, OLD), to reflect how a user classifies with natural language. The 'fuzzification' process involves taking crisp values from a 'universe of discourse'[xlix], such as age (or, in networks: delay or available bandwidth), and classifying it into a fuzzy set, such as OLD. The degree (or grade) of membership to which this value belongs to the set is calculating by using a membership function – $\mu_{OLD}()$. Membership functions can be formed from a range of shapes: they can be generated from cosine or exponential functions; they can be linear, trapezoidal, triangular or singleton. For computational simplicity triangular or shouldered patterns are often chosen [175].

[xlix] Also known as 'world domain' or 'reference super set'

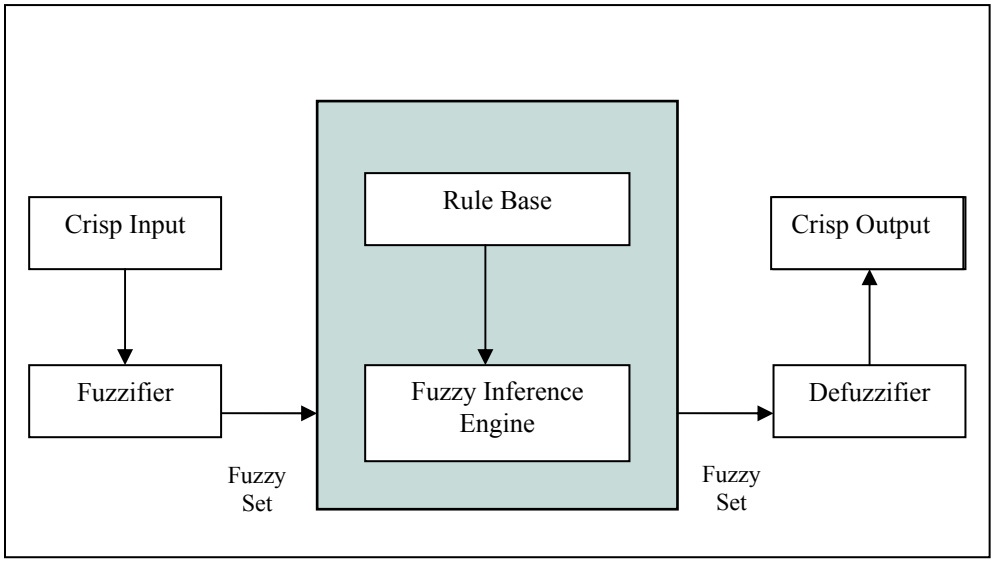


Figure 15: Fuzzy Controller

Figure 15 shows a Fuzzy Control System, including the fuzzifier/fuzzification unit discussed above. In the Mamdani-style inference approach used here [176], the next stage of the process involves the fuzzy inference engine taking the fuzzy input and applying relevant fuzzy rules. Relationships between fuzzy sets are represented by such fuzzy rules, stored in the rule-base, usually in the format '*if – then* -', ie implications. As an example, a rule could be

'IF (age IS OLD) THEN (compensation IS HIGH)'

'Age IS OLD' is true with any value within [0,1]. If, for a given crisp value s_i (eg 'age 2 years'), the membership $\mu_{OLD}(s_i)$ is zero, this rule is not active. Although the rule fires it cannot contribute to the final output value. Significantly, for any crisp value, multiple rules may both fire and be active: crisp input 'age 13' may fire rules with antecedents of 'IF(age IS OLD)' as well as 'IF(age IS YOUNG)' with membership of both greater than zero.

In fuzzy controllers the relationships between objects – either within the same set or between different sets – are of interest. An AND (ie logical $\wedge$)[1] operation is generally used in a rule to combine at least two objects in the antecedent, for example

'IF (age IS OLD) AND (injury IS HIGH) THEN (compensation IS HIGH)'

The AND operation corresponds to taking the minimum, ie the weakest, of the degree of membership values. Alternatively, usually where there is a parallel connection, the OR (ie logical $\vee$) operator can be used. This operator returns the maximum of the degree of membership values.

The implication process works to truncate the output fuzzy set ('compensation IS HIGH') to the height given by the antecedent. A graphical example is shown in Figure 16 (with arbitrary membership values).

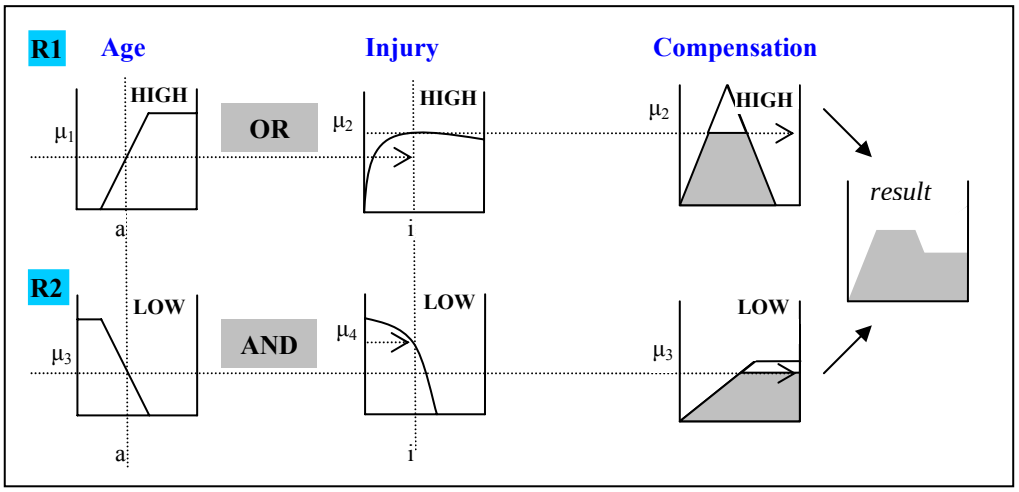


Figure 16: Fuzzy Inference

The inference engine fires the following two rules:

R1: IF age a is HIGH OR injury i is HIGH THEN compensation is HIGH

R2: IF age a is LOW AND injury i is LOW THEN compensation is LOW

[1] Although in Mamdani's paper (op cit) the symbols are reversed: $\wedge$ corresponds to min (ie or) and $\vee$ corresponds to max (ie and)

For rule R1, using OR, the maximum μ value (of the antecedent) is taken to represent how much the rule contributes (to the consequent). Here $\mu_{injury}^{HIGH}(i)$ (shown as μ_2 in the diagram) is higher than $\mu_{age}^{HIGH}(a)$ (ie μ_1). This degree of membership (μ_2) is used as the firing strength for that rule. This strength is in turn used to modify (crop) the output graph, ie the one corresponding to HIGH compensation. In rule R2 the operation AND is used, thus the lowest degree of membership (μ_3) is used for the firing strength. Finally the inference engine aggregates the two contributing output graphs, using the AND (max) operator, producing a new fuzzy set, represented by the rightmost graph.

To return a final crisp output (or decision) this fuzzy set must be defuzzified. Various techniques can be employed, notably mean of maxima (ie the point with the strongest possibility), last of maxima (LOM) or first of maxima (FOM). The most common deffuzification approach to obtain a crisp control signal is the centre of mass (also know as centre of gravity or centre of area) method. This is simplified from calculations over a continuum of points to that using a sample of points:

$$signal = \frac{\sum_i \mu(x_i)x_i}{\sum_i \mu(x_i)} \qquad \textbf{(Equation 9)}$$

The following section explains how this is incorporated into a reinforcement learning model to provide a means of reacting to and anticipating congestion.

6.1.3 Fuzzy Reinforcement Model

Membership functions (HIGH, MEDIUM and LOW) for the delay experienced by gold traffic[li] are shown in Figure 17. Earlier work used s-curves, but trapezoidal functions were substituted for their computational simplicity. In the model the membership functions for delay are additive, ie for any crisp value the sum of the

[li] ie the amount of time the traffic spends in the out-subqueue before it leaves that node

membership functions equals one ($\Sigma\mu = 1$) as this has been shown empirically to make the system more robust to noise [177].

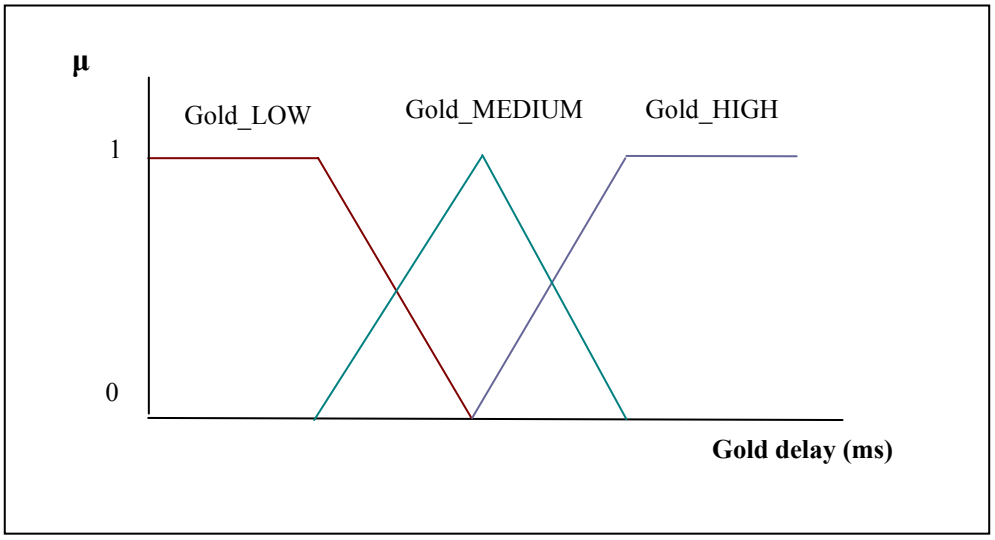


Figure 17: Delay Membership Function

Early work used just fuzzy membership with no learning. Encouraged by this, the first work employing learning restricted the number of sets to just two, ie HIGH and LOW, to reduce state space. To compensate for the loss of the middle fuzzy set the sets for LOW and HIGH were adjusted to allow for greater overlap. In addition, the observed delay values for silver traffic were not used, ie there were no silver_HIGH and silver_LOW fuzzy sets. Thus the decision model concentrated solely on the observed values for the two extreme classes: gold and bronze. The rationale behind this was to investigate whether learning, with artificially constrained state space, proved advantageous. Encouraged by these early results the state space was expanded further – from 4 to 27 – to incorporate all three classes and sets. Figure 18 provides a map of the fuzzy sets for delay:

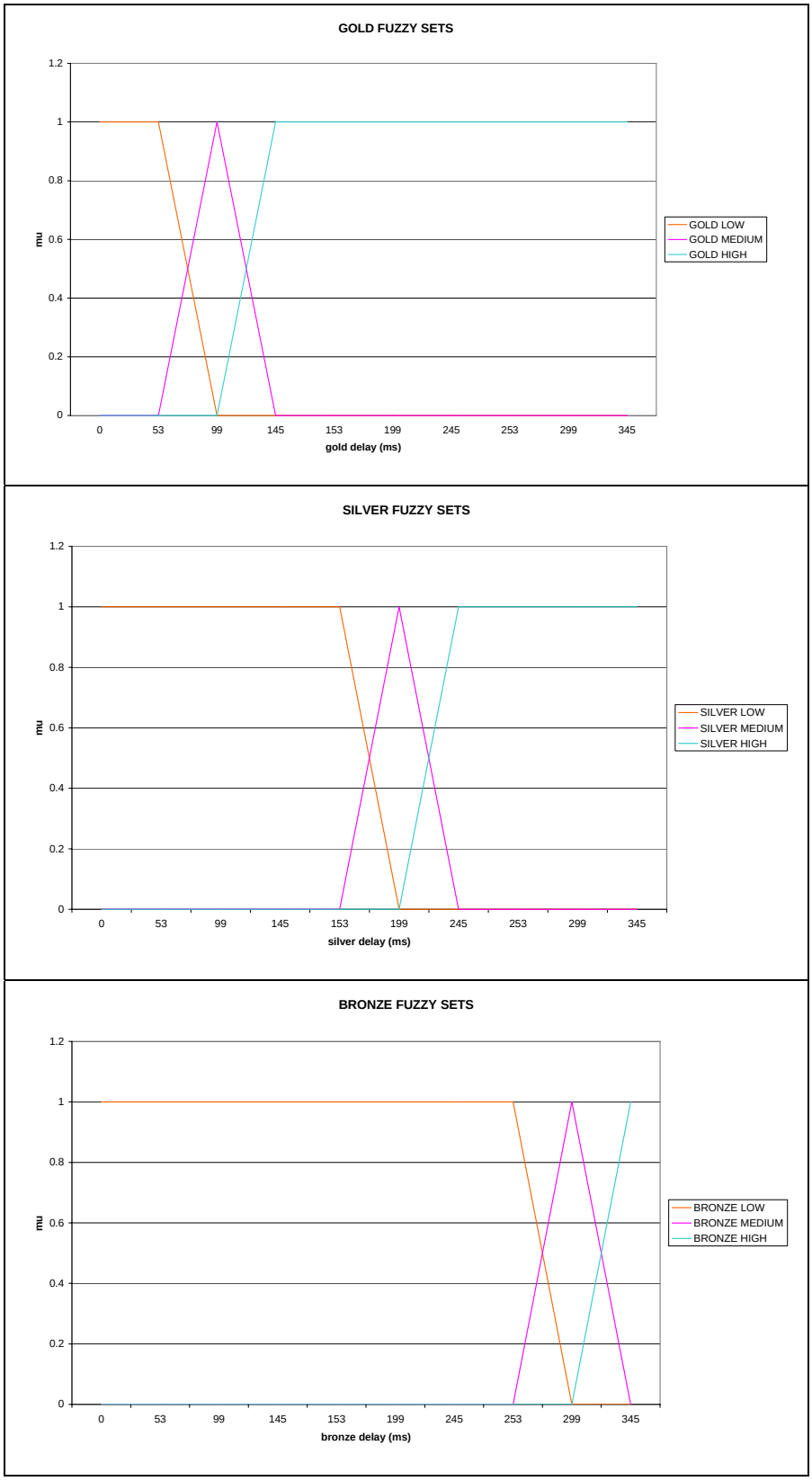


Figure 18: Fuzzy Sets for Delay

Additionally there is one membership function for delay trend, the exponential weighted moving average of the relative difference (δ):

$$\delta_t = \frac{\beta(OD_t - OD_{t-1})}{OD_{t-1}} + (1-\beta)\delta_{t-1} \qquad \textbf{(Equation 10)}$$

where OD is the observed delay. To prevent a sluggish delay trend figure the value of β was set to 0.8. Thus this metric responds more sensitively to recent shifts in delay. The same fuzzy set, shown in Figure 19, was used for all classes. This is to identify whether, despite apparent low absolute delay figures, delay is building up over that link. This potentially allows the system to behave proactively, moving traffic away from a link before delay becomes critical.

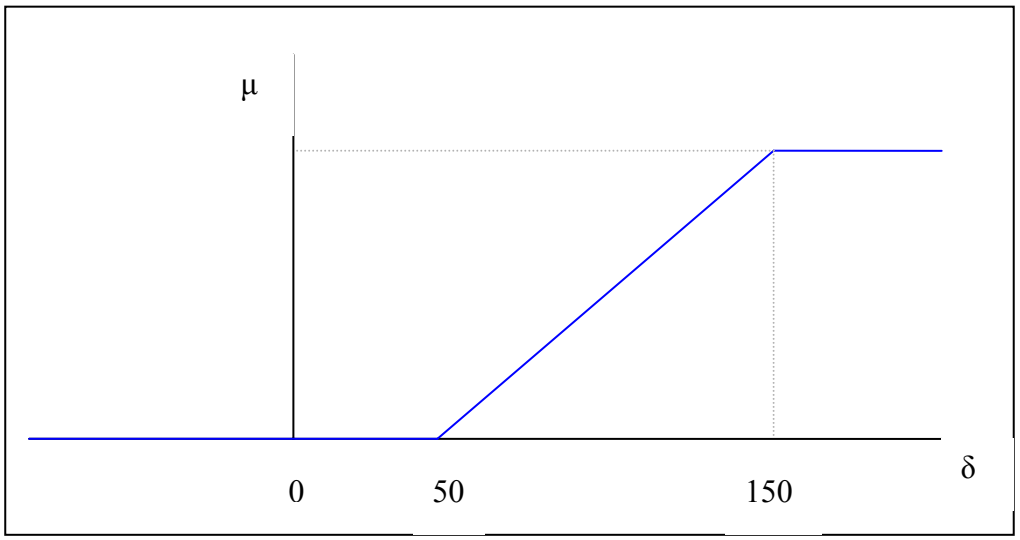


Figure 19: Delta Fuzzy Set

For each time period there is a triple of observed values: delay experienced by gold traffic, delay experienced by silver traffic and delay experienced by bronze traffic. This triplet forms a crisp state/input. There are twenty seven (fuzzy) states ($\hat{s}1_a$ to $\hat{s}1_{za}$) corresponding to one crisp input s1, used to fire rules:

Crisp state	Fuzzy states	Gold Membership	Silver Membership	Bronze Membership
s1	$\hat{s}1_a$	Gold_HIGH	Silver_HIGH	Bronze_HIGH
	$\hat{s}1_b$	Gold_HIGH	Silver_HIGH	Bronze_MEDIUM
	$\hat{s}1_c$	Gold_HIGH	Silver_HIGH	Bronze_LOW
	$\hat{s}1_d$	Gold_HIGH	Silver_MEDIUM	Bronze_HIGH
	…	…	…	…
	$\hat{s}1_x$	Gold_LOW	Silver_ MEDIUM	Bronze_LOW
	$\hat{s}1_y$	Gold_LOW	Silver_LOW	Bronze_HIGH
	$\hat{s}1_z$	Gold_LOW	Silver_LOW	Bronze_MEDIUM
	$\hat{s}1_{za}$	Gold_LOW	Silver_LOW	Bronze_LOW

Table 3: Fuzzy States

For each of the three traffic classes the OR (ie maximum) value of the delay (eg $\mu_{GOLD\_HIGH}(obs\_gold\_delay)$) and the delay trend (eg $\mu_{GOLD}(gold\delta)$) memberships is found. In Table 3, for fuzzy state $\hat{s}1_a$ the gold class membership corresponds to:

$$\mu_{GOLD\_HIGH}(obs\_gold\_delay) \vee \mu_{GOLD}(\delta) \quad \textbf{(Equation 11)}$$

The maximum was chosen so that a sudden shift in network conditions (the delay membership) would not be negated by a sluggish trend. The AND (ie minimum) value of the gold and bronze memberships for each traffic class is then found, to produce the membership for that state, eg $\mu_{\hat{s}1a}$ is composed of:

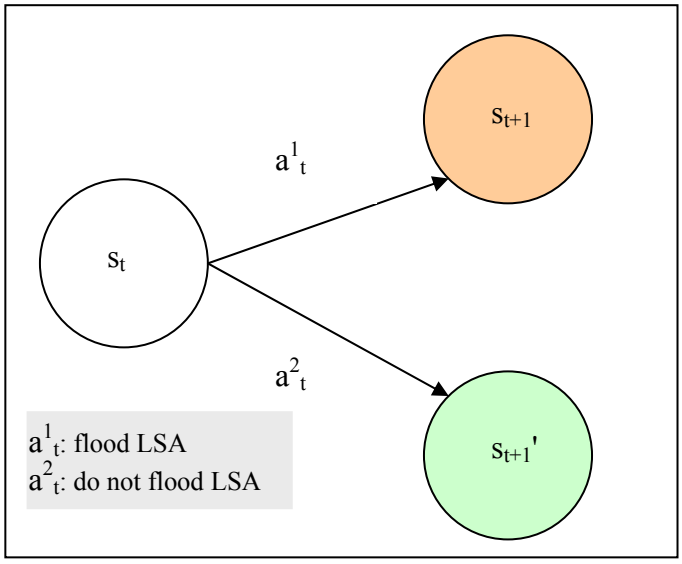
$$[\mu_{GOLD\_HIGH}(obs\_gold\_delay) \vee \mu_{GOLD}(\delta)] \wedge [\mu_{SILVER\_HIGH}(obs\_silver\_delay) \vee \mu_{SILVER}(\delta)] \wedge [\mu_{BRONZE\_HIGH}(obs\_bronze\_delay) \vee \mu_{BRONZE}(\delta)] \quad \textbf{(Equation 12)}$$

For each state there are two possible actions to choose from: θ_High and θ_Low[lii]. Here θ is the factor used to manipulate the true link cost for lower class traffic, as first presented in the pseudo-delay mechanism in Section 5.1. The relationship between the state and actions is presented in the backup diagram, Figure 20. This diagram shows the possible successor states[liii] after an action: taking action a^1_t at state s_t moves the simulation on to state s_{t+1}; taking action a^2_t at state s_t moves the simulation on to state s_{t+1}'. It should be noted that, unlike in a state transition diagram, here the consequent,

[lii] it will be seen later how these actions correspond into flood or not flood, together with the theta calculation

[liii] This is true for fuzzy and non-fuzzy states so the ŝ notation is not used

s_{t+1}, of the state action pair (s_t, a^1_t) may not necessarily be a different state to the consequent s_{t+1}', of the state action pair (s_t, a^2_t). Indeed s_{t+1} may be identical to its precedent s_t. The state outcome is dependent on the effectiveness of the action and, for non-deterministic processes, the environmental conditions. For example, in an auction scenario action a^1_t could be 'bid £40' and action a^2_t could be 'bid £10'. If the desired object were either removed from sale, or attracted a higher counter-bid neither actions would be successful (given that the aim of the action was to make a purchase). Thus the consequent states s_{t+1} and s_{t+1}' would be identical, ie with the same amount



of money and number of goods bought as state s_t.

Figure 20: State → Actions → New States

At fuzzy state $\hat{s}1_a$, corresponding to Gold_HIGH AND Silver_HIGH AND Bronze_HIGH the system can choose between fuzzy actions θ_High or θ_Low. These state action pairings are shown in Table 4:

	AND			Fuzzy Action
$\hat{s}1_a$	Gold_HIGH	Silver_HIGH	Bronze_HIGH	θ _High
	Gold_HIGH	Silver_HIGH	Bronze_HIGH	θ _Low

Table 4: Fuzzy State Action Pairs

This state-action pairing can also be described as fuzzy rules, R_1 and R_2, where the fuzzy state forms the antecedent of the rule, and the fuzzy action the consequent:

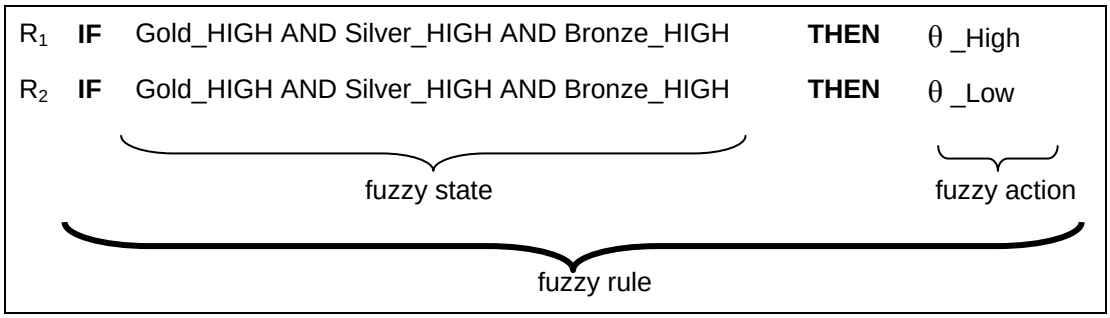


Table 5: Fuzzy Rules

Critically, this differs from standard fuzzy controllers that often show only one possible consequent from a state (rule antecedent), for example
R_1: **IF** Gold_HIGH AND Silver_HIGH AND Bronze_HIGH **THEN** θ _High
R_2: **IF** Gold_HIGH AND Silver_HIGH AND Bronze_LOW **THEN** θ _Low,

or, omitting the silver class and the medium level, in the more common tabular form:

	Bronze_HIGH	Bronze_LOW
Gold_HIGH	θ_High	θ_High
Gold_LOW	θ_Low	θ_Low

Table 6: Fuzzy State-Actions without Learning

In standard controllers with no learning there is only one possible action per fuzzy state. By contrast, when using reinforcement learning the system is trying to learn the appropriate actions for the prevailing conditions. The corresponding table for such a scenario is instead:

	Bronze_HIGH	Bronze_LOW
Gold_HIGH	θ_High θ_Low	θ_High θ_Low
Gold_LOW	θ_Low θ_High	θ_Low θ_High

Table 7: Fuzzy State-Actions with Learning

Rather than deciding at design time that the action of choice for the state Gold_HIGH AND Bronze_HIGH should be to flood an SLA[liv] this is instead resolved (ie learnt) at run time, for each set of observed (ie crisp) delay figures. The rationale for this is that although intuitively there are scenarios where a definite action can be defined, as shown in Table 8 (where * represents any level), it is not evident how the system ought to behave for all states. Thus all states (including the ones with intuitive action choices) are learned.

If any class of traffic is experiencing high levels of delay then the node should flood traffic. Related rules:					
IF	GOLD_HIGH	SILVER_*	BRONZE_*	**THEN**	θ_High/Flood
IF	GOLD_*	SILVER_HIGH	BRONZE_*	**THEN**	θ_High/Flood
IF	GOLD_*	SILVER_*	BRONZE_HIGH	**THEN**	θ_High/Flood
If gold or silver traffic are experiencing low levels of delay while bronze traffic is not experiencing a high level of delay then the node should not flood traffic. Related rules:					
IF	GOLD_LOW	SILVER_LOW	BRONZE_LOW	**THEN**	θ_Low/¬Flood
IF	GOLD_LOW	SILVER_LOW	BRONZE_MED	**THEN**	θ_Low/¬Flood

Table 8: Intuitive Statements and Corresponding Fuzzy Rules

Table 9 provides a partial list of the state action pairs (rules). Each crisp state has a membership in more than one fuzzy state, ie fires more than one rule, ie state-action pair. Where the membership value (μ) for the rule/state action pair is zero the rule will not contribute to the decision making process. In turn, for each fuzzy state in this model there are two state action pairs, ie possible rules to fire. Each state action pair has an associated strength, or FQ value, which indicates that pair's suitability to be in the optimal model[lv]. For example, for state $\hat{s}1_a$ and action θ_High there is one FQ value – FQ($\hat{s}1_a$, $\hat{a}_1$) – and for state $\hat{s}1_a$ and action θ_Low there is another FQ value - FQ(§1a, â2). An ε greedy policy is taken to choose the action for each fuzzy state. This guarantees that with (small) probability ε a random action is chosen; otherwise the action with the highest known reward, ie FQ value, is chosen for each fuzzy state ($\hat{s}1_a$ - $\hat{s}1_{za}$), corresponding to crisp state s1. This provides for exploration as well as

[liv] It will be shown later how θ_High corresponds to flood and θ_Low corresponds to NOT flood
[lv] The FQ element of fuzzy Sarsa corresponds to the Q-value of standard Sarsa, which in turn is equivalent to the V or value element in Q-learning.

exploitation of known values. Additionally, where the FQ values are equal for the two state action pairings the resultant action is chosen randomly.

$\hat{s}1_a$	Gold_HIGH	Silver_HIGH	Bronze_HIGH	$\mu_{\hat{s}1a}$	θ_High	$FQ(\hat{s}1_a, \hat{a}_1)$
	Gold_HIGH	Silver_HIGH	Bronze_HIGH		θ_Low	$FQ(\hat{s}1_a, \hat{a}_2)$
$\hat{s}1_b$	Gold_HIGH	Silver_HIGH	Bronze_MED	$\mu_{\hat{s}1b}$	θ_High	$FQ(\hat{s}1_b, \hat{a}_1)$
	Gold_HIGH	Silver_HIGH	Bronze_MED		θ_Low	$FQ(\hat{s}1_b, \hat{a}_2)$
$\hat{s}1_c$	Gold_HIGH	Silver_HIGH	Bronze_LOW	$\mu_{\hat{s}1c}$	θ_High	$FQ(\hat{s}1_c, \hat{a}_1)$
	Gold_HIGH	Silver_HIGH	Bronze_LOW		θ_Low	$FQ(\hat{s}1_c, \hat{a}_2)$
$\hat{s}1_d$	Gold_HIGH	Silver_MED	Bronze_HIGH	$\mu_{\hat{s}1d}$	θ_High	$FQ(\hat{s}1_d, \hat{a}_1)$
	Gold_HIGH	Silver_MED	Bronze_HIGH		θ_Low	$FQ(\hat{s}1_d, \hat{a}_2)$
	…	…	…		…	
$\hat{s}1_x$	Gold_LOW	Silver_MED	Bronze_LOW	$\mu_{\hat{s}1x}$	θ_High	$FQ(\hat{s}1_x, \hat{a}_1)$
	Gold_LOW	Silver_MED	Bronze_LOW		θ_Low	$FQ(\hat{s}1_x, \hat{a}_2)$
$\hat{s}1_y$	Gold_LOW	Silver_LOW	Bronze_HIGH	$\mu_{\hat{s}1y}$	θ_High	$FQ(\hat{s}1_y, \hat{a}_1)$
	Gold_LOW	Silver_LOW	Bronze_HIGH		θ_Low	$FQ(\hat{s}1_y, \hat{a}_2)$
$\hat{s}1_z$	Gold_LOW	Silver_LOW	Bronze_MED	$\mu_{\hat{s}1z}$	θ_High	$FQ(\hat{s}1_z, \hat{a}_1)$
	Gold_LOW	Silver_LOW	Bronze_MED		θ_Low	$FQ(\hat{s}1_z, \hat{a}_2)$
$\hat{s}1_{za}$	Gold_LOW	Silver_LOW	Bronze_LOW	$\mu_{\hat{s}1za}$	θ_High	$FQ(\hat{s}1_{za}, \hat{a}_1)$
	Gold_LOW	Silver_LOW	Bronze_LOW		θ_Low	$FQ(\hat{s}1_{za}, \hat{a}_2)$

Table 9: Fuzzy State Action Pairs for all States

The focus now shifts to the penultimate column of Table 9 – how to choose the action. Not only are crisp states fuzzified in the fuzzy reinforcement learning algorithm but so are the actions. Here the actions θ_Low and θ_High are represented by two fuzzy sets. The θ_Low fuzzy set is represented as a singleton, returning a zero value. The rationale for this is that this corresponds to a 'do not flood' action choice – thus the value of the smear factor (the θ) is irrelevant. However, the (fuzzy) action θ_High corresponds to a 'flood with theta' action choice. The membership function for θ_High is a steep curve[lvi], truncated to return a maximum value of two, given by the following equation:

$$\min\{2, 1 - e^{-\alpha x}\} \qquad \textbf{(Equation 13)}$$

An inverse transformation is employed – ie the membership of the fuzzy state is used to obtain the value of theta. This θ value is used for the action figure, $ac(\hat{s}_i)$, or ac_{High_θ}) towards the calculation of the theta used to manipulate the link cost metrics.

[lvi] Various other curves were investigated, including cosine and logarithmic functions

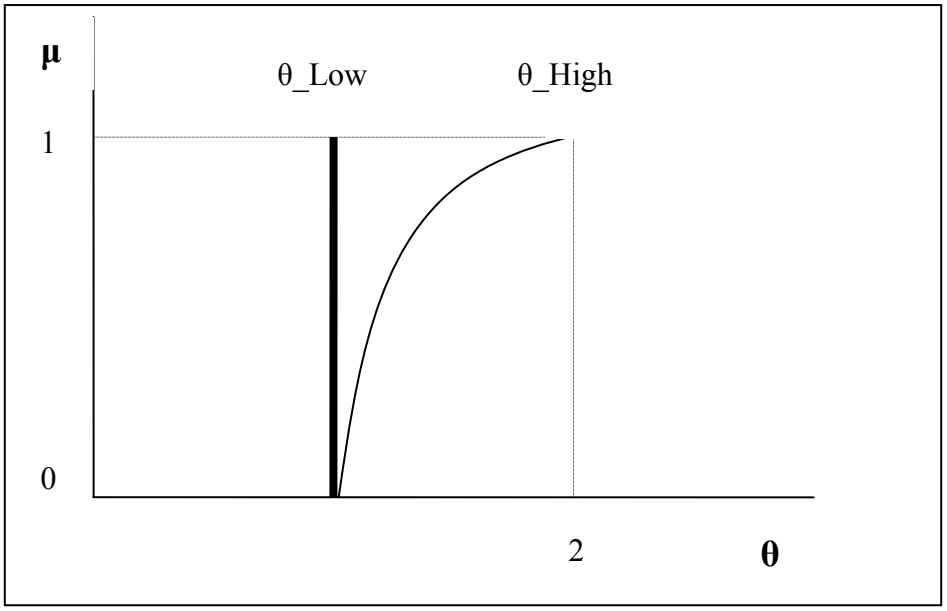


Figure 21: Fuzzy Action Membership Functions

In practice the critical action decision for IP networks is binary, ie whether to flood or not, as opposed to the more familiar continuous decision space discussed in section 6.1.2. A weighted probabilistic choice is used to determine (defuzzify) the flooding decision. This is consistent with similar approaches for reinforcement learning problems, eg [178]. For each fuzzy state the FQ values for a flood action are weighted by the state's membership value. The sum of these are then normalised against the totals for both flood and do not flood:

$$prob(FLOOD) = \frac{\sum_{i=1}^{27} \mu(\hat{s}_i) FQ_i\left(\hat{s}_i, \hat{a}_{High\_\theta}\right)}{\sum_{i=1}^{27} \mu(\hat{s}_i) FQ_i\left(\hat{s}_i, \hat{a}_{High\_\theta}\right) + \sum_{i=1}^{27} \mu(\hat{s}_i) FQ_i\left(\hat{s}_i, \hat{a}_{Low\_\theta}\right)} \quad \textbf{(Equation 14)}$$

A flood then occurs with the above given probability. This, again, allows for exploration (ie not following what presents as the optimal solution). The value of theta, to manipulate the cost metric, is generated by calculating the centre of mass of all the chosen actions for each fuzzy state:

$$theta = \frac{\sum_{i=1}^{27} \mu(\hat{s}_i) ac(\hat{s}_i)}{\sum_{i=1}^{27} \mu(\hat{s}_i)} \quad \textbf{(Equation 15)}$$

where $ac(\hat{s}_i)$ is the action value for the state action pair with the highest FQ value for each fuzzy state $(\hat{s}_i)$[lvii], for all fuzzy states with $\mu>0$. This action value provides the theta value, ie the factor added to the true cost of a link, as shown in Table 10. The cost of the link is then flooded in the LSA using gold observed delay for the gold traffic across that link, silver observed delay plus theta for the silver traffic and bronze observed delay plus theta for the bronze. Thus all link state databases will be updated with the fabricated link costs.

Gold Traffic	Silver Traffic	Bronze Traffic
$ObservedDelay_{GOLD}$	$ObservedDelay_{SILVER} + \theta$	$ObservedDelay_{BRONZE} + \theta$

Table 10: Theta Flooding

The purpose of the reinforcement learning model is to update the FQ values for each state-action pair (rule), ie to learn the appropriate FQ value for state-action pairs. The fuzzy Sarsa equation, from [166], is used to update the FQ values for each state-action pair:

$$FQ(s_{t-1}^i, a_{t-1}^i) = FQ(s_{t-1}^i, a_{t-1}^i) + \alpha \xi_{(s_{t-1}^i, a_{t-1}^i)} \left(r_t + \gamma \sum_{\forall j} FQ(s_t^j, a_t^j) \xi_{(s_t^j, a_t^j)} - FQ_{t-1}(s_{t-1}^i, a_{t-1}^i) \right) \quad \textbf{(Equation 16)}$$

where α is the learning factor, γ is a discount factor, r the reward and ξ is the 'fuzzification factor'. The discount factor was set to 0.9, seen as a typical value for discrete-time reinforcement learning [179]. The fuzzification factor, introduced in [180], is used to weight each rule contribution. This is represented by the relative contribution of the state action pair (rule) with respect to the contribution provided by all the state action pairs that correspond to the same crisp state:

[lvii] Hereafter for clarity s will be used in place of ŝ as all future states will be fuzzy so the crisp:fuzzy distinction does not need to be maintained

$$\xi_{(s^i,a^i)} = \frac{\mu(s_t)}{\sum_{i=1}^{27} \mu(s_i)} \qquad \textbf{(Equation 17)}$$

where $\mu(s_t)$ is the membership of that fuzzy state (of the state-action pairing) whose FQ values are being updated.

A reward, calculated when the new observed delay figures are viewed ten seconds later[lviii], forms the means to evaluate outcomes. The reward is a signal from the environment to the node (or, more formally, agent). The reward returned is weighted to (initially[lix]) return a value of one for a no flood decision. Otherwise the relative difference (Rel_D) of the delay, capped to return a minimum value of zero and a maximum of one, is returned as the reward:

$$\text{Rel\_D} = \frac{OD_{ij}^{t-1} - OD_{ij}^{t}}{OD_{ij}^{t-1}} \qquad \textbf{(Equation 18)}$$

where OD_{ij}^{t} is the observed delay over link i-j at time t. Research has highlighted the advantage of selecting a relative over a 'delta'/fixed threshold [181,182] when triggering updates, thus this appears a valid means of establishing a reward. As stated earlier, the essential difference between Sarsa and Q-learning was that the former is an on-policy learner and so employed only actions that are followed. Thus $\sum_{\forall j} FQ(s_t^j, a_t^j)$ corresponds to the FQ values selected by the ε greedy policy at the next time interval.

The algorithm for the reinforcement learning is as follows:

[lviii] Time intervals are discussed in Section 7.5
[lix] This value is varied in simulations

- Initialise all FQ(s,a) values to zero
- Initialise s_t (start fuzzy state)
- Choose at for s_t using COA, using all s_t that match the crisp state s and at using ε greedy selection policy
- For each step (ie every 10 second delay inspection):
    - Take action a_t – observe r and s_{t+1}
    - Choose a_{t+1} from s_{t+1} using e greedy selection policy for all s_{t+1} match s_{t+1}
    - $$FQ\left(s_{t-1}^{i}, a_{t-1}^{i}\right) = FQ\left(s_{t-1}^{i}, a_{t-1}^{i}\right) + \alpha\xi_{\left(s_{t-1}^{i}, a_{t-1}^{i}\right)}\left(r_t + \gamma\sum_{\forall j} FQ\left(s_t^{j}, a_t^{j}\right)\xi_{\left(s_t^{j}, a_t^{j}\right)} - FQ_{t-1}\left(s_{t-1}^{i}, a_{t-1}^{i}\right)\right)$$
- $s_t = s_{t+1}$, $a_t = a_{t+1}$

Figure 22: Tokarchuk's Fuzzy Sarsa Algorithm

7 Design and Verification

A simulation was constructed using OPNET modeller 8.1 [183]. This operates at the packet level.

7.1 Topology

In the simulation there are 13 active nodes[lx], of which 6 (the darker nodes in Figure 23) generate data traffic. All 20 links are bi-directional with identical transmission rates, giving the average node degree of 3.23. Such a topology is consistent with comparable research [184].

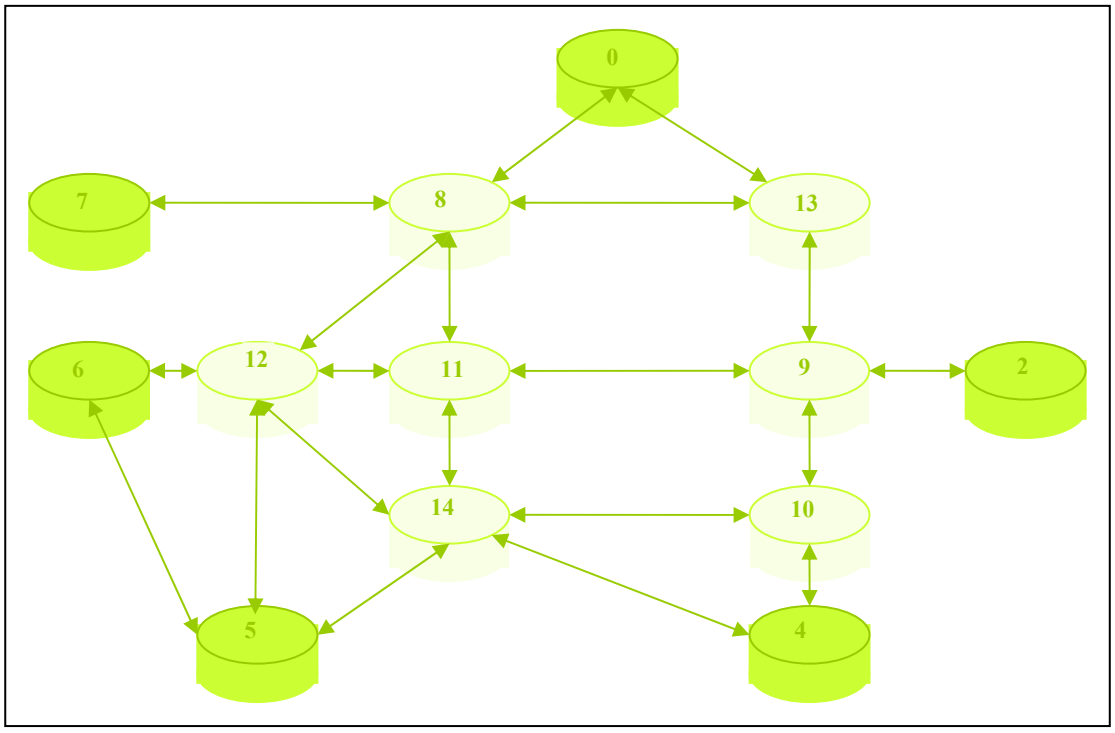


Figure 23: Network Topology

The destination for each data packet is allocated randomly, with the exception of traffic generated at nodes 0 and 7. All traffic from these two nodes has the destination field set to 11. The rationale behind this is to guarantee generating a traffic 'hot spot'

[lx] Irregularities in the node numbering are explained by inactive nodes which are not represented in the digram

where the congestion level causes the delay to increase over a link – here the link from node 8 to node 11.

7.2 Nodes

The nodes have been designed to be largely consistent with current node architectures, such as Juniper Networks M160 [185] or Cisco 7200.

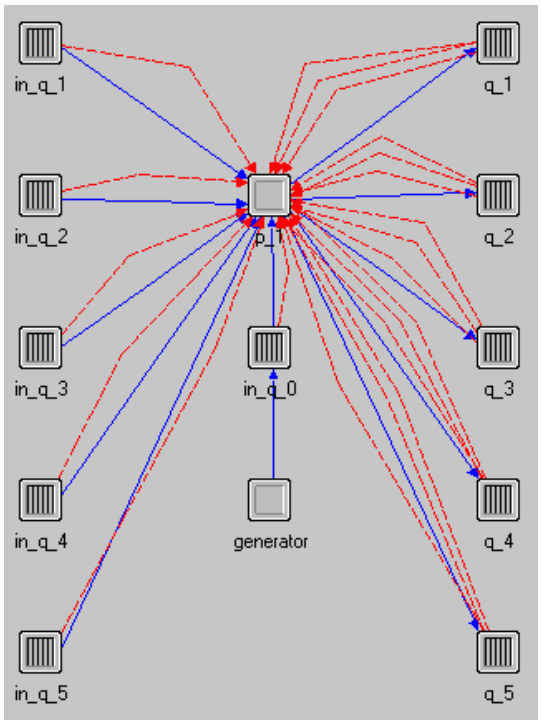


Figure 24: Node model

Each node has an in-queue (labelled 'in_q_no') for each link and a corresponding out-queue (labelled 'q-no'). The node represented in has an in and out queue for each of its five neighbours, plus an in queue for the traffic from the traffic generator. The traffic generator is discussed later. The blue arrows represent the direction of traffic within the node; the red arrows, discussed later, are statistic wires. These are a means in OPNET for a processor to obtain variable values from another processor within the same node.

7.2.1 In-Queues

Since it is assumed that the processor operates at wire speed there are no sub queues within the in-queues and no queue size. Any packet dropping is performed at the out-queues. As each in-queue receives a packet, whether data or signal, it sets a flag. The core processor for each node polls these flags on a round-robin basis[lxi]. If the flag is set the core processor forces an (OPNET remote) interrupt in that in-queue and the packet is forwarded to the processor.

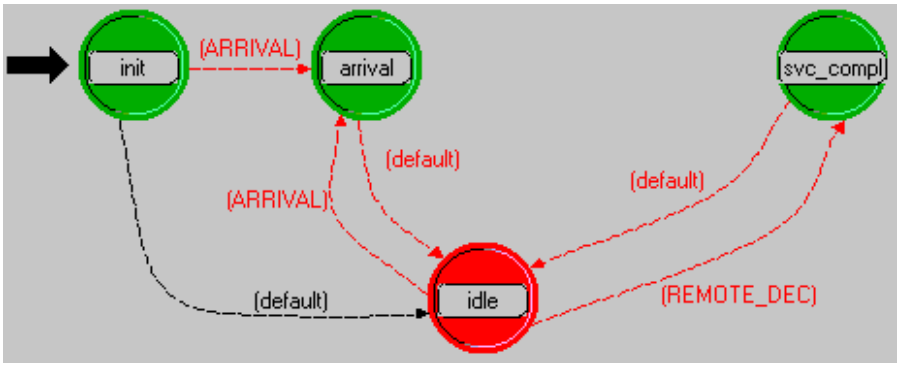


Figure 25: In-Queue Model

7.2.2 Out-Queues

Each out-queue has four sub-queues to buffer packets when they arrive faster than the transmission rate. The default queue limit for a Cisco 7200 router, for example, is 64 packets before a drop policy is initiated. For the 7500 routers the default limit is calculated according to a proportional allowance for each class in the parent buffers [186]. The determination is based on a maximum delay of 500ms with an average packet size of 250 bytes[lxii].

The first (highest priority) sub-queue forwards signal traffic the next three forward the gold, silver and bronze data traffic respectively. The signal sub-queue is serviced ahead of all the other queues. When this is empty a class-based queuing mechanism is employed. Gold traffic is statistically serviced 70% of the time, silver 20% and 10%, when there is traffic in all sub-queues. To mimic packet transmission the queue holds the packet to be sent for the packet service time (packet size/transmission speed seconds), during which time it cannot service any other packets already in the sub-

[lxi] In OPNET there is a statistic wire, red in node model, from each in-queue to the core processor
[lxii] This 'low' figure of packet size is due to the level of TCP service traffic.

queues (though it can add newly arrived packets to the sub-queues). No propagation delay is modelled, which is plausible for access networks.

Each out-queue stores delay statistics for the sub-queues, representing the time spent in a sub-queue (less the service time). These figures can be accessed by the core processor via a statistic wire (one for each sub-queue, excluding the signal sub-queue) and are used to determine the delay from one node to its neighbours for all classes.

7.2.3 Core Processor

In this module delay over outgoing links is monitored, packets reaching their destination are destroyed, packets for forwarding are switched from the in-queue to the appropriate out-queue and any learning is undertaken. The 'agent-like' element of the simulation is located here.

The data structures associated with the OSPF protocol are located in this module. The link state database contains costs for each class/link from every node in the network. The routing table contains the next hop for each class/destination. While an authentic router would maintain both routing and forwarding table, instead the simplified simulation router maintains just the routing table, as conceptually the two tables are similar [187], and the core processor acts as both routing and forwarding engine.

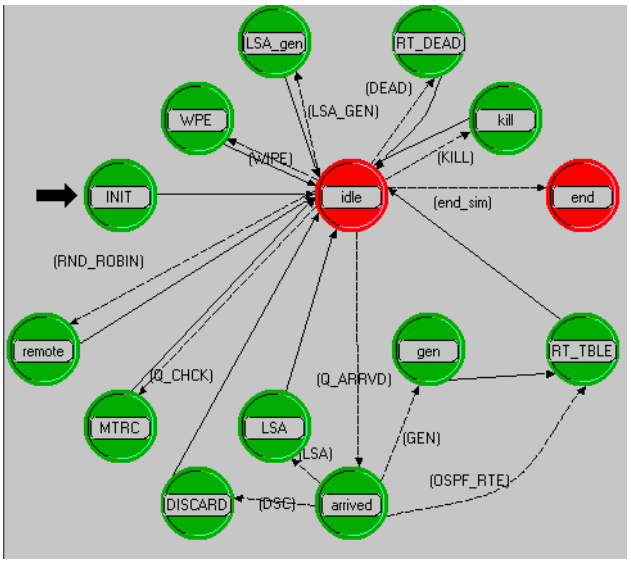


Figure 26: Core Processor Model

7.3 Packet Generation

Traditionally network traffic has been modelled using a Poisson process (ie packet arrival is memory-free and interarrival rate is exponentially distributed). Where a user population is large, with each user only responsible for a small percentage of overall Internet traffic and user sessions are mutually independent, a Poisson session arrival process would be expected. While this model appears correct for modelling network user session arrivals [188], it has latterly been considered unsuitable for describing both the network connections that make up such sessions as well as packet arrivals. The analysis of Ethernet traffic in [189] demonstrated that self-similarity was the prevailing characteristic. Ignoring this inherent burstiness by employing the Poisson model, it is argued, distorts traffic behaviour. For example Poisson models of packet traffic smooth aggregate traffic as the number of sources increase, rather than intensifying it. However, the overview of traffic modelling in [190] now suggests that packet-level behaviour may have shifted to a Poisson process. Indeed the authors suggest that individual links may display variable behaviour.

Another concern for this analysis is whether the assumption of stationary process holds in networks[lxiii]. However, even if it is accepted that traffic follows the Poisson distribution, recent physical analysis of queue output [191] suggests the output of a stable queue is not stationary. Even if the external traffic is stationary (eg its arrival is exponentially distributed) the internal traffic process is not stationary. This raises the question of whether a representative time slice can be found for any learning techniques.

Faced with these findings, it becomes more problematic how to model the traffic across the links. Since much work investigating QoS provision across IP networks still employs the Poisson model for traffic generation, for consistency several of the simulations presented here use this model. When using this latter model an inter-

[lxiii] This is an issue for most artificial intelligence, not solely reinforcement learning

arrival time (ie 1/λ) of 0.0884 is set[lxiv]. Additionally, to simulate burstiness, an ON/OFF packet generation was used in the simulations. The approach delineated in [192] was to employ a standard ON/OFF Markov source (and additionally a periodic source) with fixed transmission rate when ON, arguing that it captures the behaviour where performance is largely determined by bursty congestion. ON/OFF models, with exponentially distributed ON and OFF times, are also utilised in the IST project MESCAL [193] to model VoIP traffic. In the simulations in this work the ON and OFF times are distributed according to a uniform integer distribution, set to 90% to remain in the same state.

Traffic of all classes is generated from an identical source / generation model. Thus these simulations doe not attempt to model classes based on application, for example gold traffic as VoIP and best-effort as email. Instead classes are based on user demand. Here a customer pays for gold-class service, expecting a certain level of guarantee, while the customer who is unwilling to pay for service guarantees accepts the prevailing best-effort service. This is not an unrealistic assumption – although the focus on QoS primarily considers needs of applications with differing demands these still function, albeit often less efficiently, in the traditional best-effort internet. Thus, using the example in [194], a university student may be prepared to accept the shortcomings of standard internet VoIP, while the Principal may both require high-quality calls and have the funds to pay for this. Here the role of the QoS enhancements is to provide a generalised solution rather than one tailored to the assumed needs of various applications.

7.3.1 Random Number Generator

The Mersenne Twister has been used as the Random Number Generator (RNG) due to its rigorous statistical properties, such as a long period of equidistribution [195]. Additionallly it demonstrates efficient memory usage and is four times faster than rand(). The critical importance of selecting an acceptable RNG in order to validate results was stressed in [196].

[lxiv] In an ON/OFF simulation the delay (ie wait time in each state) is set to be packet size/service time. The packet size is 4420 and the service time is 100,000, ie 0.0442. Since the generator is in the send state 50% of the time the comparative poisson interarrival time is 2*0.0442, ie 0.0884

At the start of each simulation a new 'seed' is generated by the RNG (and the value stored in a file for consistency with other simulations). This seed is fed in to the RNG in a succeeding simulation, where multiple runs were required of the 'same' simulation. An example is shown below:

1643545771	1979890667
2024759641	511486124
431267977	1902544604
246090048	1169522929
1861415293	1417352009
324587625	879442603
1152939417	1536053098
563448707	75532875
1731460838	1233963376
1674506578	1235311782
632206830	219358932
1042982389	1379843426

Table 11: Randomly Generated Seeds

7.4 Packet Format

All data packets are identical in size. The size of 440 bytes was chosen as representative of TCP packet size, ignoring the small (40-44 byte) packets, for example TCP acknowledgement [197]. The headers of interest, ie for QoS differentiation, would be absorbed into the SERVICE TYPE header field currently employed by DSC field points, as discussed in section 2.1. The size of service packets, ie LSAs, is set at 35 bytes.

7.5 Multi-class Traffic

Traffic is split into three classes: gold (priority), silver and bronze (best-effort), randomly allocated in the ratio 2:3:5 respectively. The two lower priority classes, silver and bronze, can also be further randomly demarcated into strata, numbered 1-5, if alternative routing is employed. This is to provide a means of routing a percentage (0-100% in multiples of 20%) of the traffic for a class without having to redesign the OSPF forwarding mechanism. Within the out queue class based scheduling is employed to enable favourable treatment for the higher-class traffic. Priority is always given to service traffic, such as OSPF LSAs. The work in [72] demonstrated that such

traffic represents a very small proportion of all traffic so the preferential treatment should not hinder the data traffic. This latter traffic is routed such that in a queue with data traffic of all classes gold is serviced 70% of the time, silver 20% and bronze 10%.

The delay experienced in each out-subqueue is polled every 10 seconds by the core processor. This period is chosen to mimic the OSPF hello timer.

7.6 Simulation Scaling

Ideally the network simulated should be running at Ethernet transmission rate, ie 10 Mbits/sec[lxv]. To mimic this, the out-queue service rate was set to 10,000,000, the interrupt delay in the generator was set to 10,000,000/packet size (ie 4420), the polling rate of the in-queues by the core processor was set to 0.0000001*LinkNumber. However, the simulation as a result ran slowly, due to the number of events to process per second. For malleability the simulation was scaled down by a factor of 100. Later simulations have further scaled this rate down, in order to analyse the efficacy of the algorithms under greater strain.

7.7 Simulation Verification

The aim of verification is to capture programming and coding errors, or more precisely to evaluate how correctly a model's implementation matches the intent of the designer [198].

To verify correct functioning of the input queues the number of events in the event queue of each core process (in OPNET terms: how many local events were in the queue) was monitored. A function, define_interrupts(), was coded to list the events local to each process. Figure 27 shows typical command line output. The stream interrupts are packets (either service or data) and self interrupts are called for monitoring in-queues, generating LSAs etc.

[lxv] It could be argued that speeds across a MAN would be even higher, eg up to 10-gigabit Ethernet

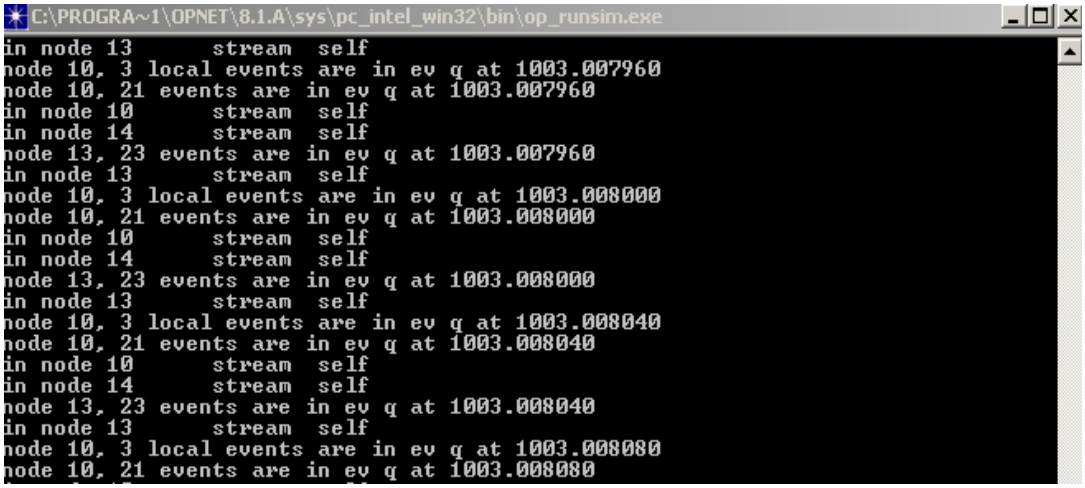


Figure 27: Interrupts

To verify queue servicing a small verification network was built, as shown in . In the network four generating nodes sent traffic to a fifth sink node. Tests were run to confirm that the input queues serviced the packets they received. Figure 29 demonstrates that all the packets received by the input queue (the top graph) are then delivered by that queue (the lower graph). Further tests verified that the round robin scheduling mechanism was fair: Figure 30 shows that the core process of node_0 (the sink node) removes a balanced number of packets from each of its inqueues (in_q_1-in_q_4, which each receive packets from the corresponding nodes 1-4). Correct functioning of the class-based weighted fair queue mechanism was also verified, demonstrated in Figure 31 (albeit for the earlier simulations where a higher percentage of traffic was gold).

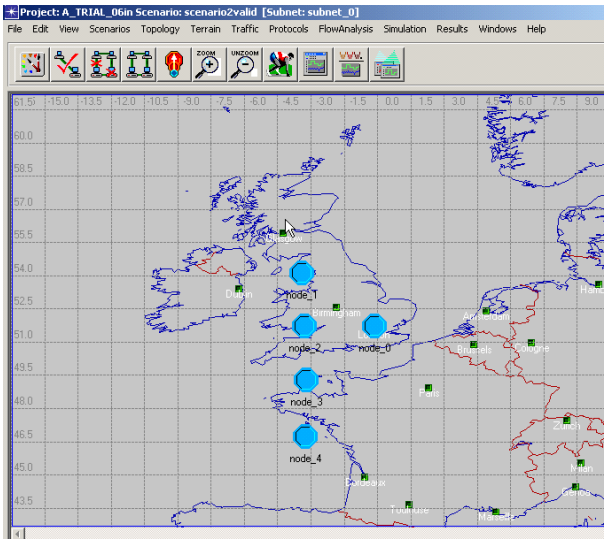


Figure 28: Verification Network

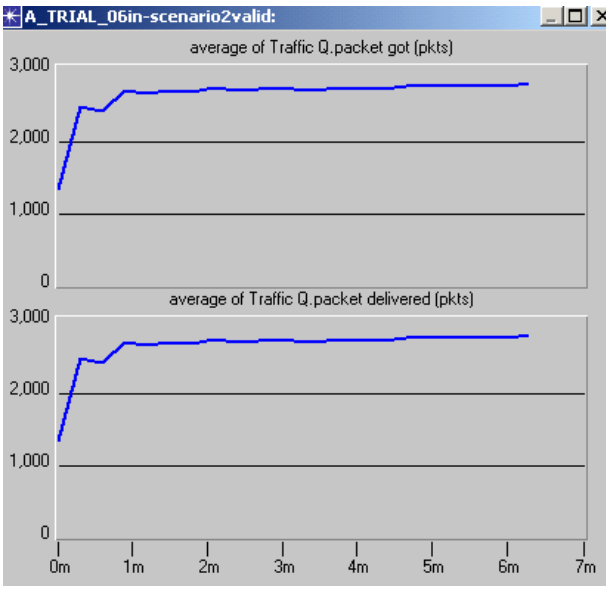


Figure 29: In_Queue Servicing

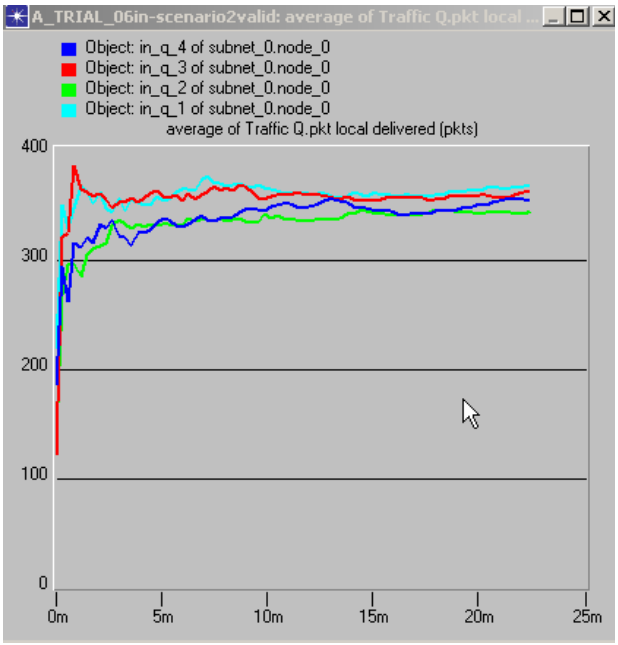


Figure 30: Round Robin Servicing

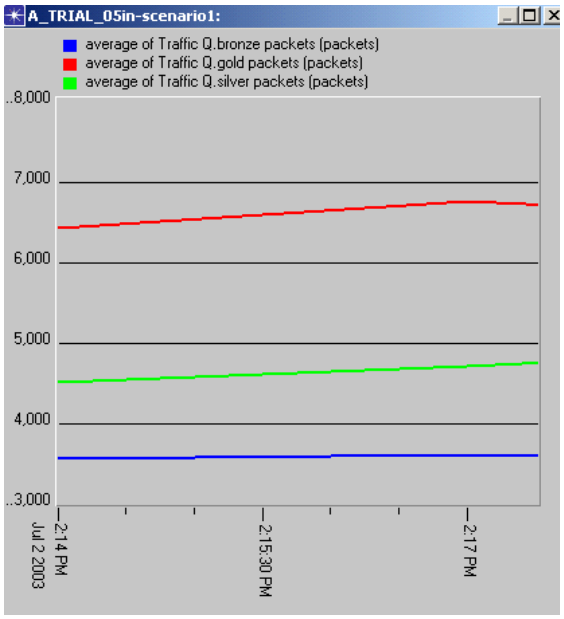


Figure 31: Weighted Fair Queue

The correct functioning of the ON/OFF packet generation mechanism is demonstrated in Figure 32, showing that the generator is either in a wait or a send_packet state:

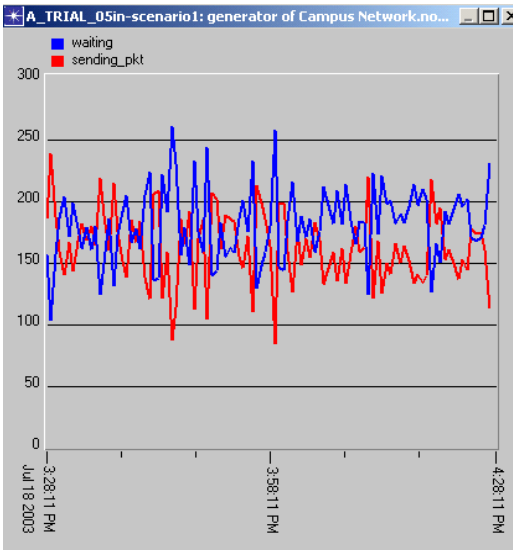


Figure 32: ON/OFF Packet Generation

To quantify whether the network had been placed under strain, the utilisation of the link from nodes 8→11 was measured. Figure 33 shows that at certain points in the simulation this link was heavily utilised (60% and above), representing considerable traffic stress.

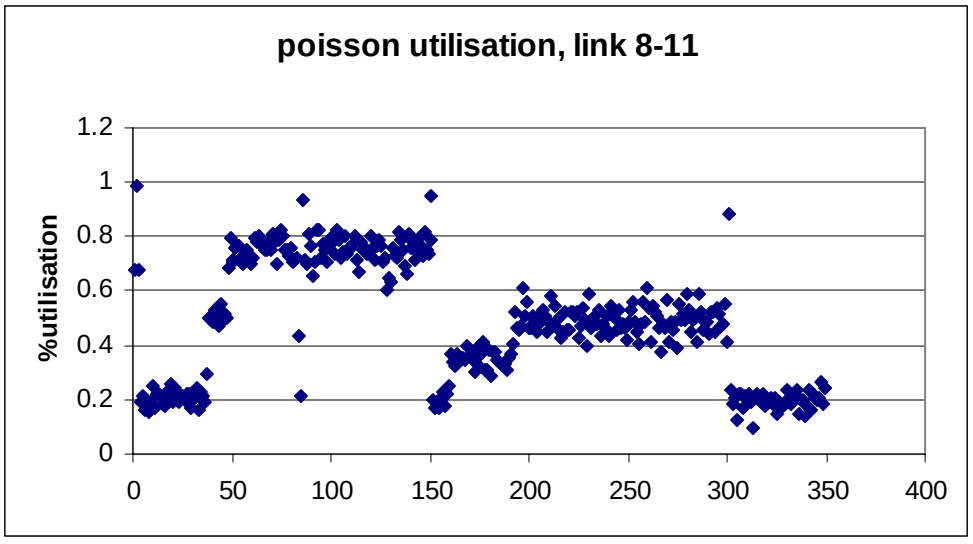


Figure 33: Link Utilisation

Further simulation verification can be found in Appendix 1.

8 Results

The motivation behind this work was to investigate whether agents could have a role in IP network resource allocation. Given the tension between the close coupled nature of IP networks and the autonomy demanded of the agent paradigm the role of agents is potentially compromised. However, since learning is one of the key features that characterise agent intelligence it was resolved to employ this property in order to add more responsiveness to a dynamic environment. The behaviour of the intelligent network is contrasted to that where a heuristic (ie the non-learning pseudo-delay mechanism presented in section 5.1) is used to modify traffic.

The majority of results shown are for single simulation runs. Confidence intervals (over multiple runs differing by RNG seed) are omitted as much of the evidence from various simulations has indicated a limited spread for the intervals. However, the following figure (12 runs, ON/OFF traffic generation, reward of five for not-flooding and 100,000 bits/second transmission rate) illustrates an issue with simulations that employ reinforcement learning. Exploration is a vital ingredient to this learning mechanism. A risk associated with this is increased variability – as can be seen by some of the confidence intervals. Reducing exploration would lead to a reduction in variability, however this would negate one of the inherent learning properties.

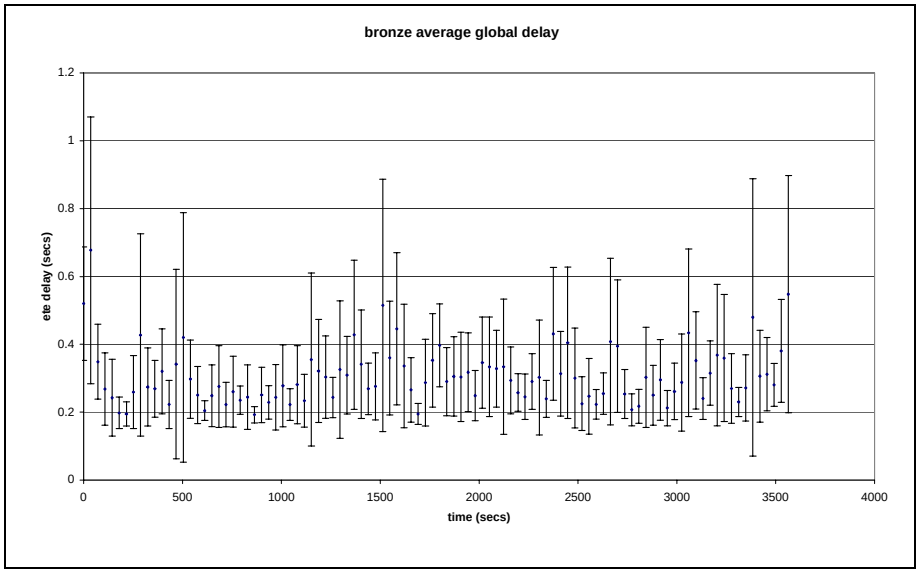
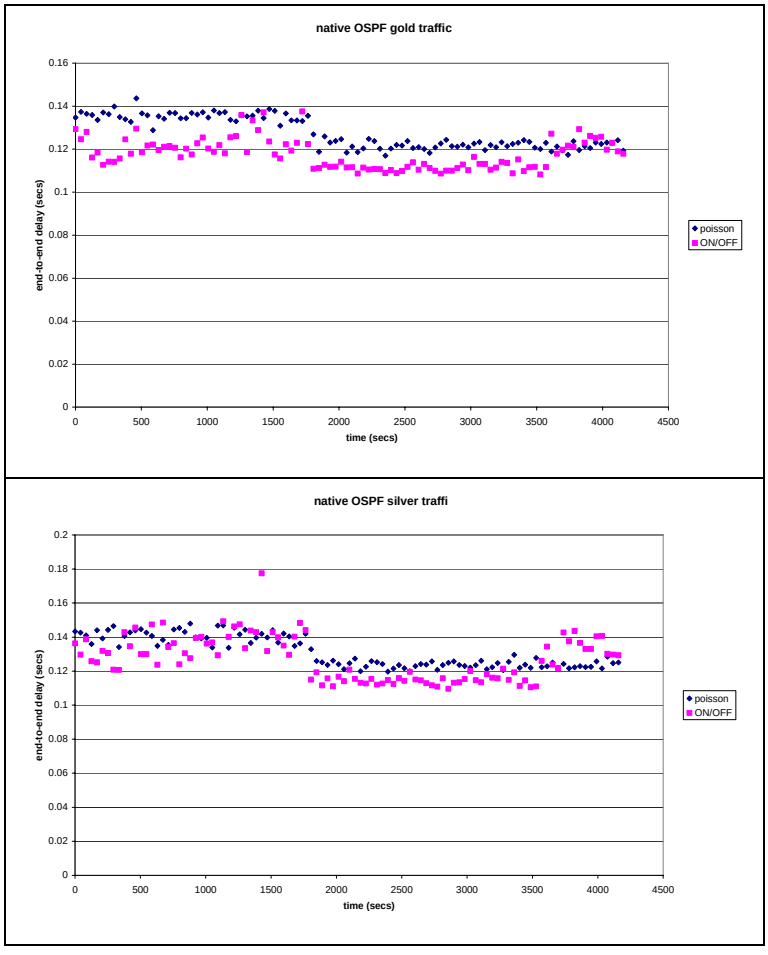


Figure 34: Bronze Delay with Confidence Intervals

8.1 OSPF

The control status of the network is displayed in Figure 35 where the performance under a benchmark OSPF is shown for both Poisson and ON/OFF generated traffic. With this version of OSPF only periodic flooding occurs; no LSAs are sent out in response to increased network delay. Thus these periodic floods are the only points where nodes are updated with network conditions. The network shifts in response to the periodic flooding can clearly been seen in the graphs, most notably around the first (1800 second) flood.



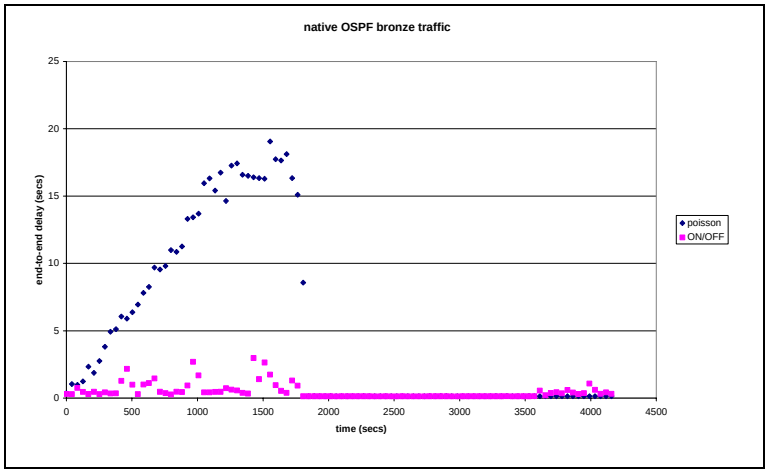
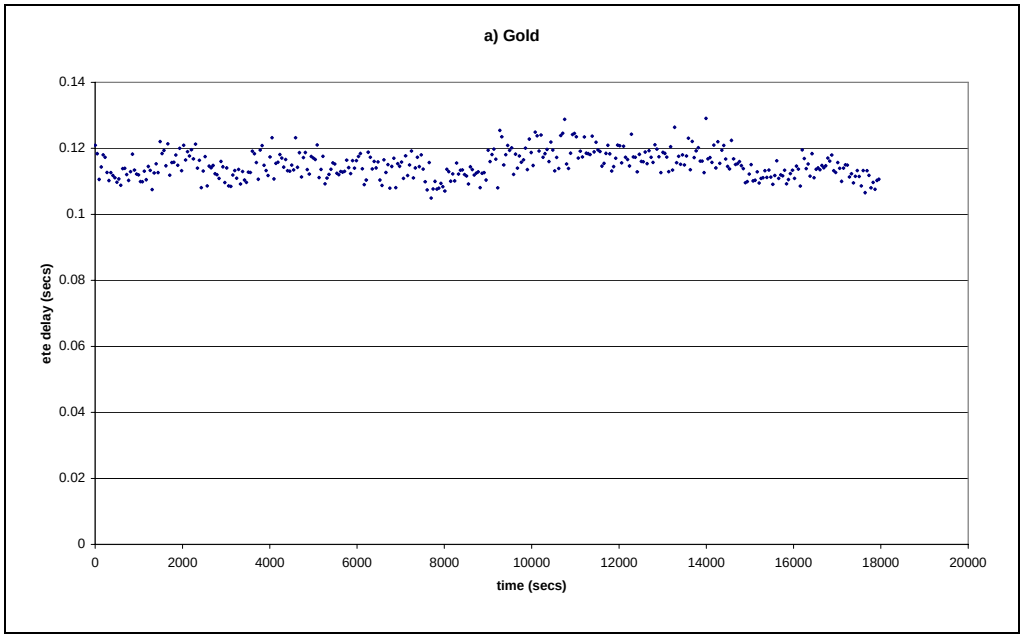


Figure 35: Benchmark OSPF

Although this is a benchmark OSPF model, it is perhaps unfairly crude as a control given that it is highly unresponsive to network stresses. A more sensitive OSPF model was created with high and low watermark thresholds: if delay reaches a critical threshold an LSA flood was generated, setting link cost (for that class) to 10; if the delay over this link was restored to a lower threshold a new LSA flood, with class/link cost of 1, was propagated.



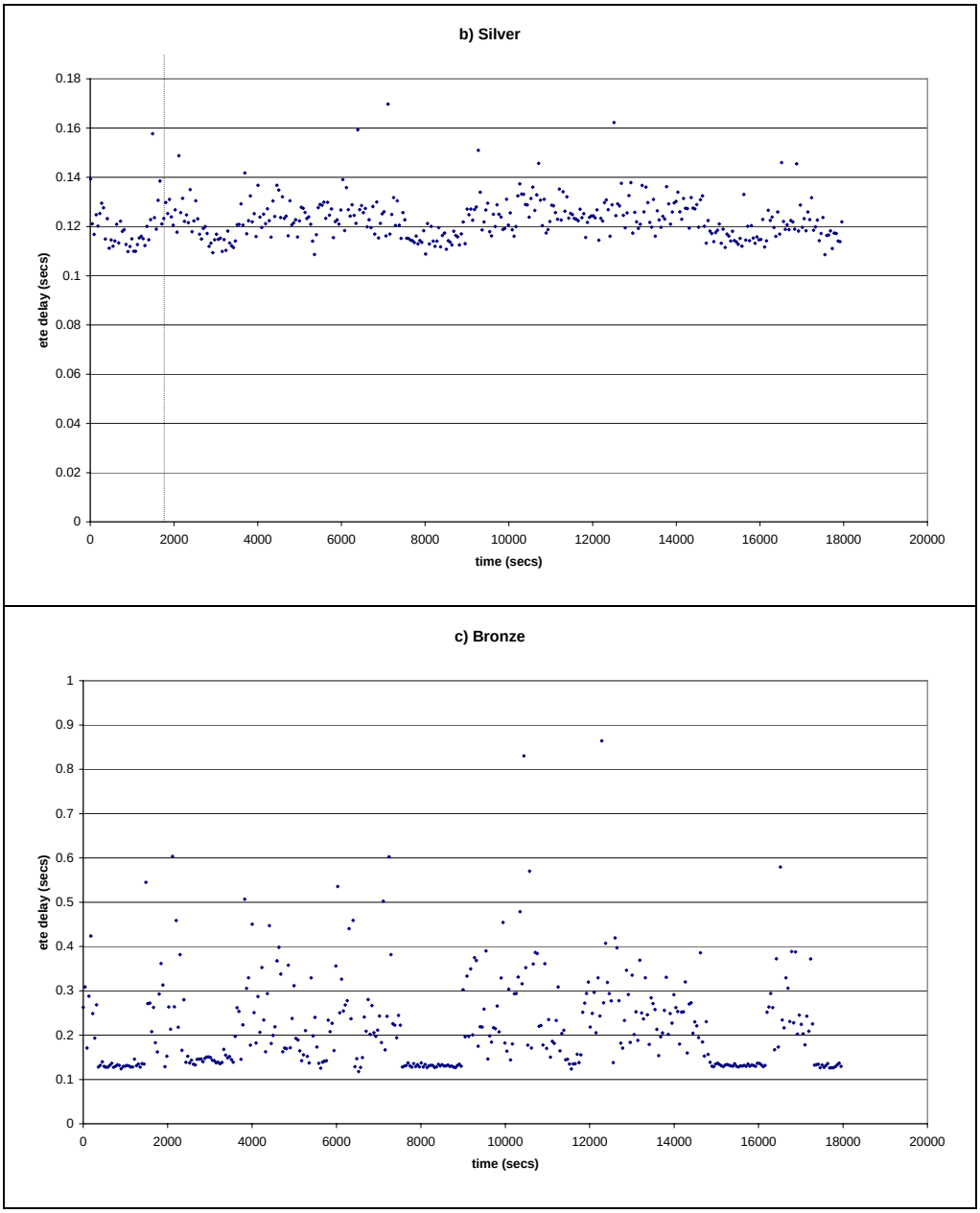
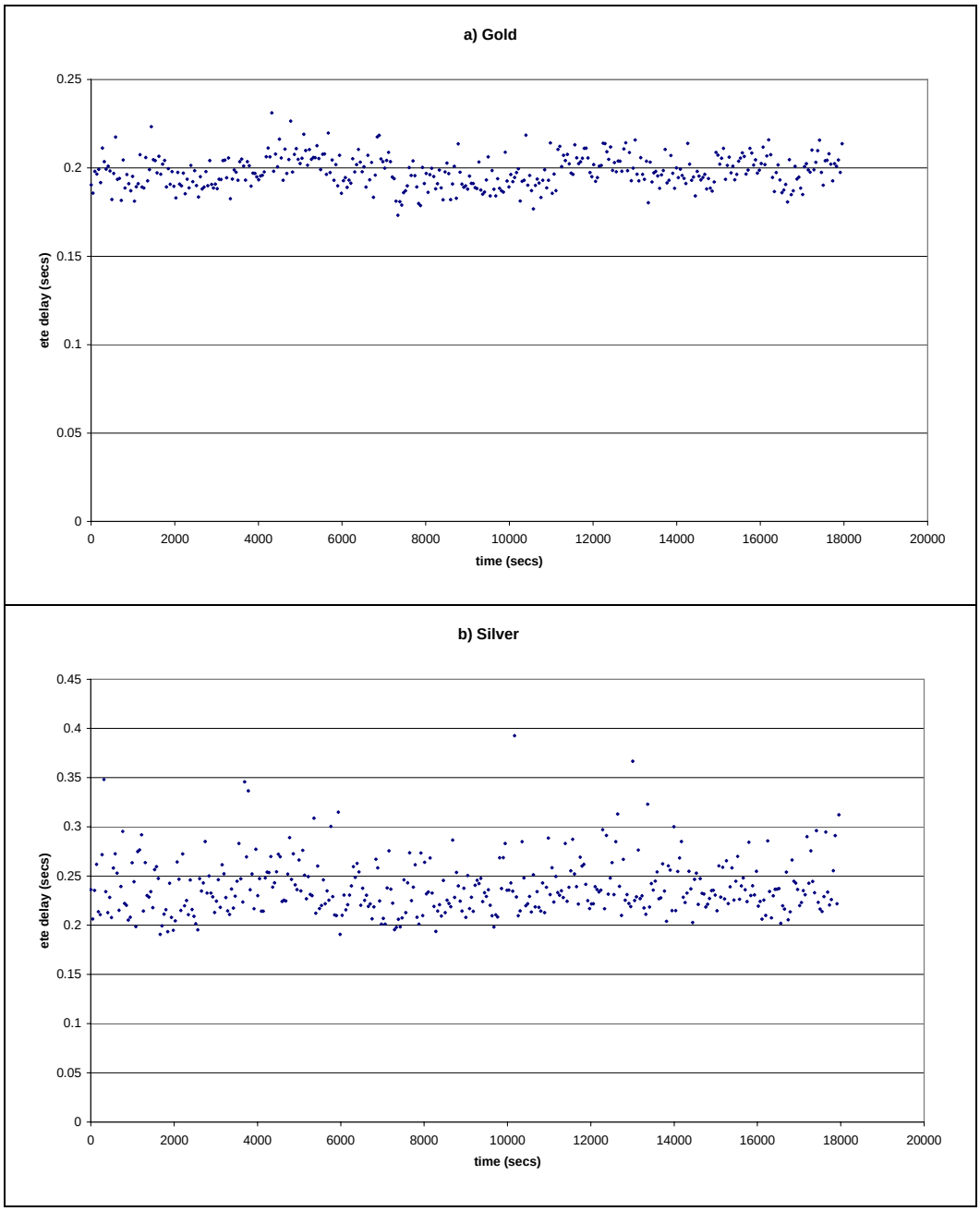


Figure 36: OSPF with Responsive Flooding

This responsive model represents an additional yardstick against which the more sophisticated algorithms must be tested. The network average end-to-end delay figures for gold, silver and bronze are 0.115, 0.123 and 0.219 seconds respectively, in a network with the standard transmission rate of 100,000 bits/second. The corresponding standard deviation measurements are 0.004, 0.008 and 0.106. However, once strain is placed on the network the performance of bronze traffic degrades. The following graphs show network average end-to-end delay in a network where the transmission rate has been lowered to 75,000 bits/second. In this congested

network the network average delay is now 0.198, 0.239 and 11.076 respectively for gold, silver, bronze traffic. The standard deviation figures are 0.009, 0.028 and 17.716.



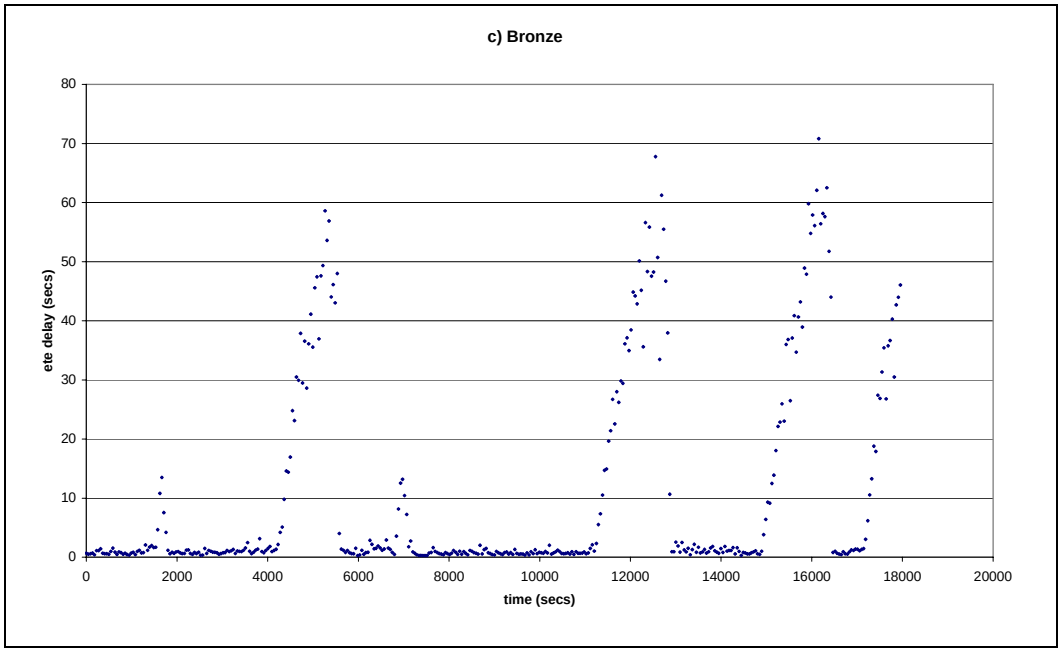
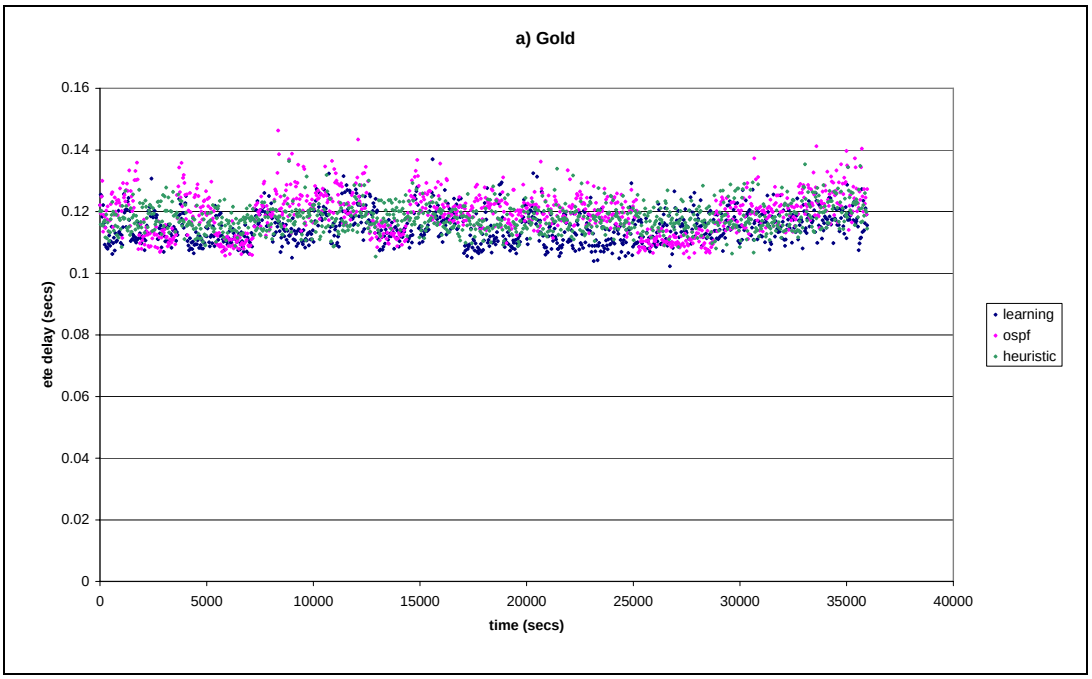


Figure 37: OSPF with Responsive Flooding in a Congested Network

8.2 Average Network Delay



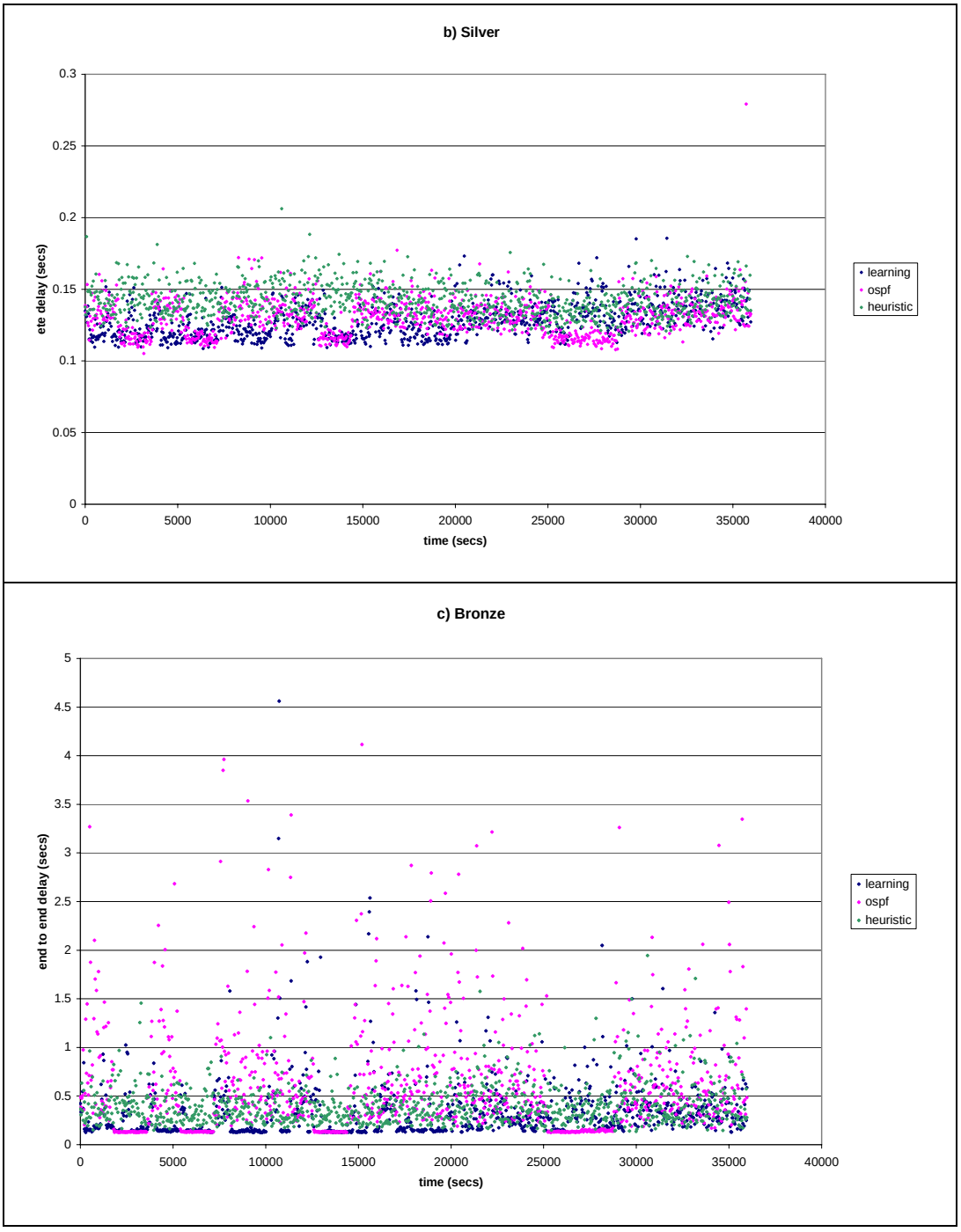


Figure 38: Network End-to-End Delay[lxvi]

A comparison of the average end-to-end delay across the network is presented in Figure 38. The benchmark OSPF simulation (ie with no flooding other than every 30 minutes), used as a control, merely performs regular 30 minute updates – there is no other responsiveness to congestion. The average delay for gold traffic across the three

[lxvi] Simulation parameters: ON/OFF traffic generation; transmission speed 100,000 bits/second

models showed negligible difference: learning 0.115, heuristic (ie the pseudo-delay mechanism with no learning) 0.118, OSPF 0.119 seconds. The average for silver traffic again only showed a slight improvement due to learning, and a marginally worse performance for the heuristic: learning 0.128, heuristic 0.142, OSPF 0.130. However, bronze traffic achieves lower delay across both the learning and heuristic networks: learning 0.34, heuristic 0.41, OSPF 0.61. This suggests that the performance of the low cost traffic can be enhanced without compromising on the handling of the premium traffic. However, traffic appears more volatile across the intelligent (learning) network, compared to that employing the heuristic. The maximum delay exhibited by bronze traffic in the learning network was 4.56 seconds; the equivalent for the heuristic network was 1.94. The standard deviation confirms the relative instability of the learning mechanism: 0.328 for learning, 0.209 for heuristic. An explanation for this could be found in the exploratory nature of a reinforcement learning policy – an ε greedy strategy will occasionally follow less apparently advantageous action choices. The heuristic will, however, always be guided by its rule of thumb and not exhibit any exploratory behaviour.

However, the heuristic and the learning simulation results compare less favourably to the more responsive model of OSPF – with the high and low watermark thresholds – shown in Figure 36. This could suggest that the extremely simple algorithm that ignores the delay values across a link in favour of applying a crude high cost metric may prove a more successful strategy. The heuristic and learning mechanisms sought to respond more sensitively to prevailing (and anticipated) network conditions, yet adding an artificially high cost appears to outperform their sophistication. While conceding that such simplicity is apparently more successful in the lightly congested network the benefits of the more complex approach will be considered in the following section.

8.3 Responsiveness to Congestion

As stated in section 7.6, the transmission rate was originally downgraded from 10 Mbits/second to 100,000 bits/second in order to make the simulations more amenable.

In order to investigate how the system behaves under greater traffic stress the link transmission rate in several of the later simulations were further scaled down to 75,000 bits/second. Figure 39 displays average network link utilisation across both a slow (ie scaled down to 75,000 bits/second, as shown by the dark red bars) and fast (ie standard 100,000 bits/second) network. Additionally, differing reward functions were employed: a no-flood reward of one (r1) across the congested network and a no-flood reward of eight (r8) across the faster network. As would be expected, with more strain on the slow network, three links became highly congested, displaying utilisation rates of 60-70%.

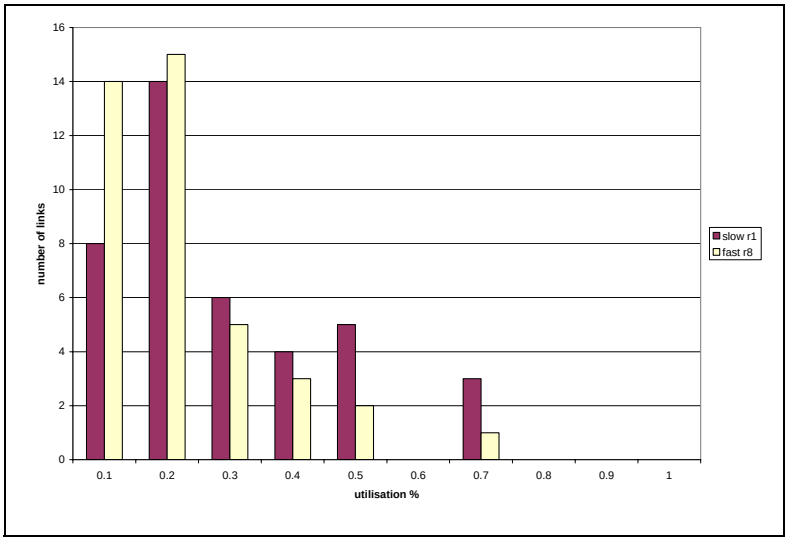


Figure 39: Slow and Fast Network Link Utilisation[lxvii]

In an ON/OFF network with links slowed to 75,000 bit/sec, link utilisation is shown in Figure 40. Utilisation is shown for a range of conditions, for learning with differing reward functions (where r1 and r2 are a non-flood reward of one and two respectively[lxviii]) and for a simulation running the pseudo-delay heuristic.

[lxvii] Simulation parameters: ON/OFF traffic generation
[lxviii] The impact of the reward function is examined more closely in section 8.7.

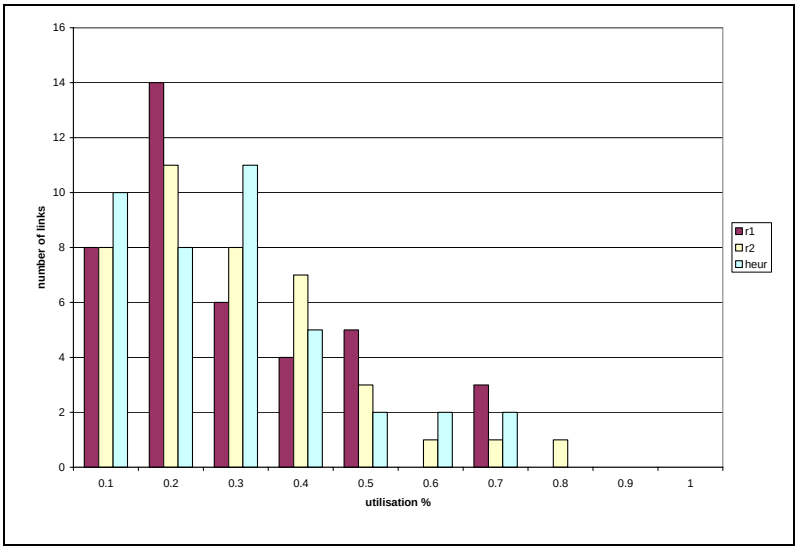


Figure 40: Slow Network Link Utilisation

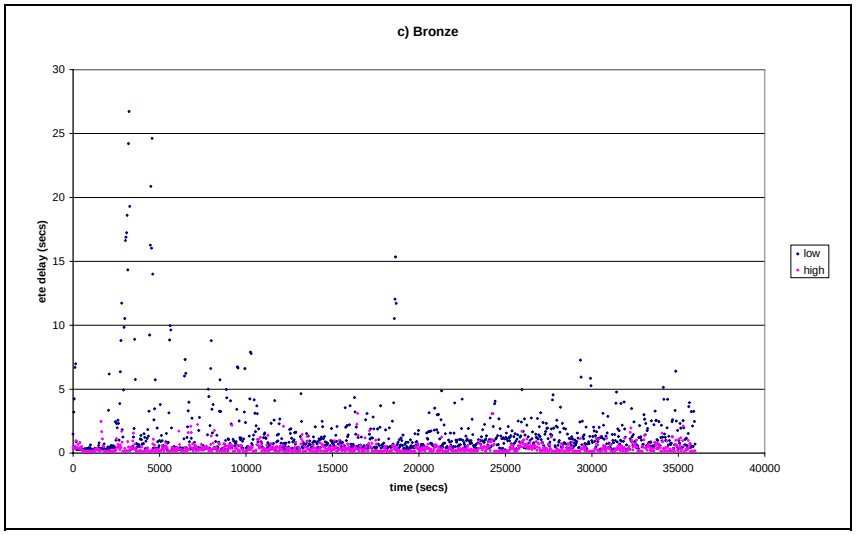


Figure 41: Impact of Slow Links on Delay[lxix]

The traffic impact of the slower links is shown in Figure 41, with the learning mechanism employed over both low and high speed links. Notably much higher average delay, standard deviation and peak end-to-end delay is observed for bronze traffic (as expected). A shorter simulated run over a network with the lower speed links – bronze traffic is presented in Figure 42 – reveals both the benchmark OSPF protocol and the learning struggling with the network conditions. The OSPF network can only send out periodic LSAs (every 30 minutes) so cannot respond to increasing

[lxix] Simulation parameters: ON/OFF traffic generation; transmission rate 75,000; learning reward=1

network stresses outside these flood times. While the responsiveness of the learning algorithm aids it in avoiding the massive delay figures associated with the benchmark OSPF, once the vast delay surges die down, the OSPF network becomes associated for some time with lower average delay.

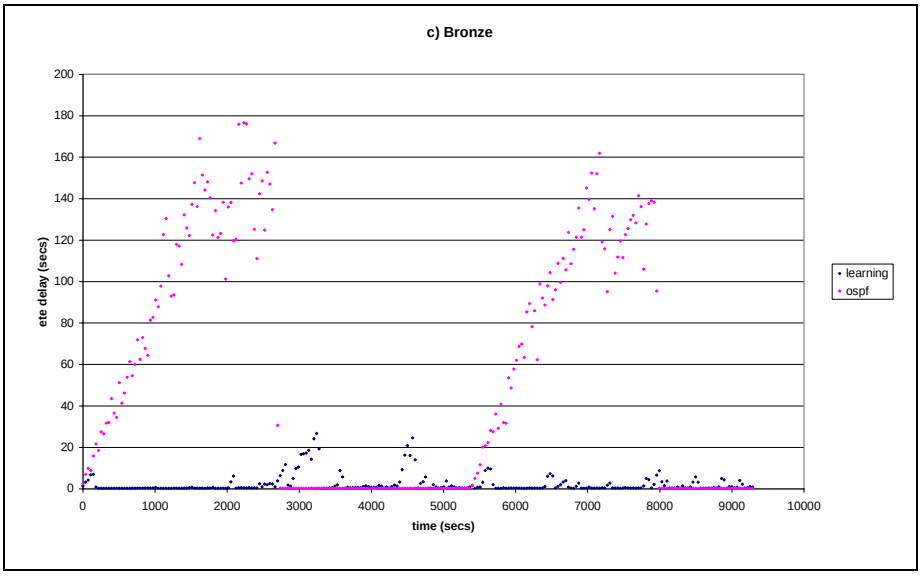


Figure 42: OSPF v. Learning over Slow Links

The end-to-end delay for traffic with destination node 11 is shown in Figure 43, suggesting that traffic to this node is responsible for much of the congestion in the network.

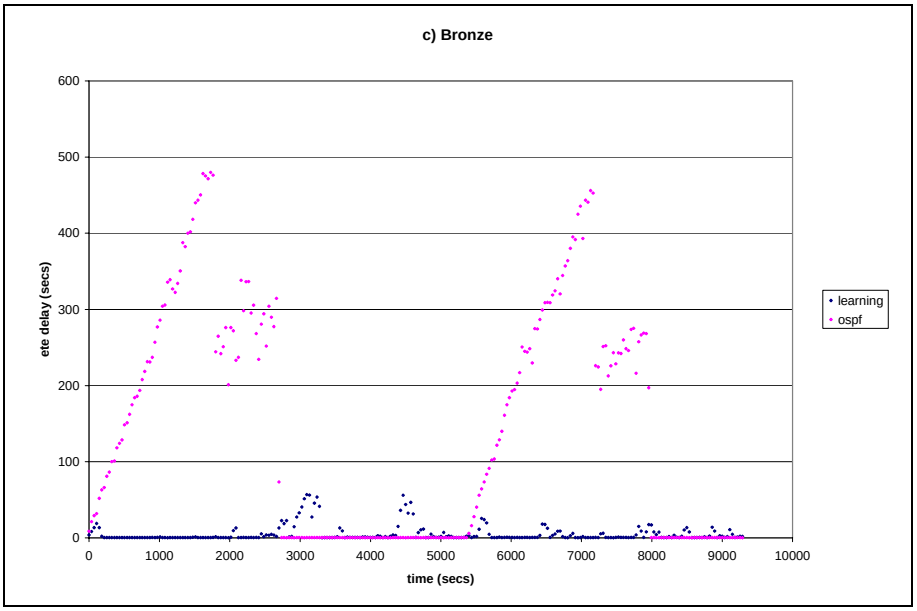
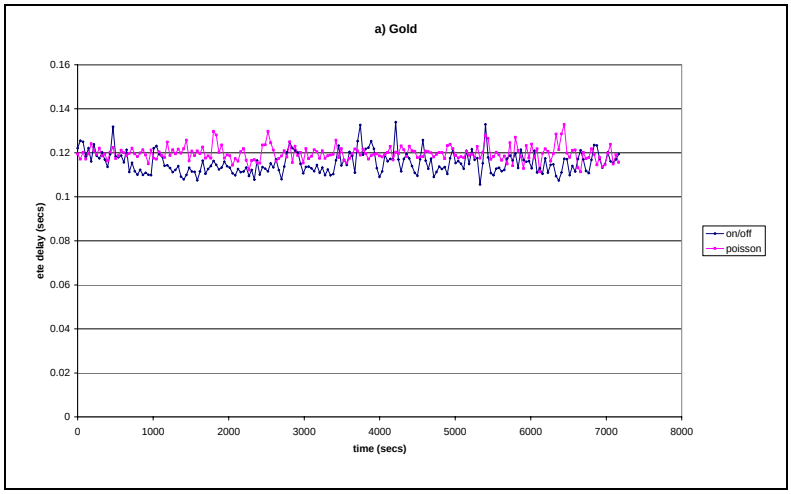


Figure 43: Impact of Slow Links on Node 11 Traffic

These results demonstrate that although neither mechanism can adequately solve the routing problem in a very congested network, the peak and average performance of the learning mechanism outweighs that of the non-responsive (ie benchmark) OSPF. Results for the more responsive OSPF model in Figure 37 also showed that a previous successful, albeit simple algorithm, struggled once the network underwent greater strain. The bronze network average delay for the learning mechanism of 1.560 seconds (with standard deviation of 2.613) compares favourably with the responsive OSPF figures of 11.076 (17.716). This indicates that seeking to add sensitivity to network routing is a valuable goal.

8.4 Traffic Model

Most of the results (including all those above) are from networks with ON/OFF generated traffic. Nevertheless, as stated earlier [see section 7.3], Poisson distributed traffic is frequently used in related research, despite doubts about its suitability to model IP network traffic. Due to its widespread usage, several simulations were run using Poisson traffic – a comparison of network average end-to-end delay is shown in Figure 44. Across this two-hour long run the average bronze delay is lower for Poisson traffic (0.261 seconds) compared to ON/OFF (0.341 seconds). However, due to the some large spikes in the Poisson simulation the standard deviation is higher than that for ON/OFF (0.490 compared to 0.319).



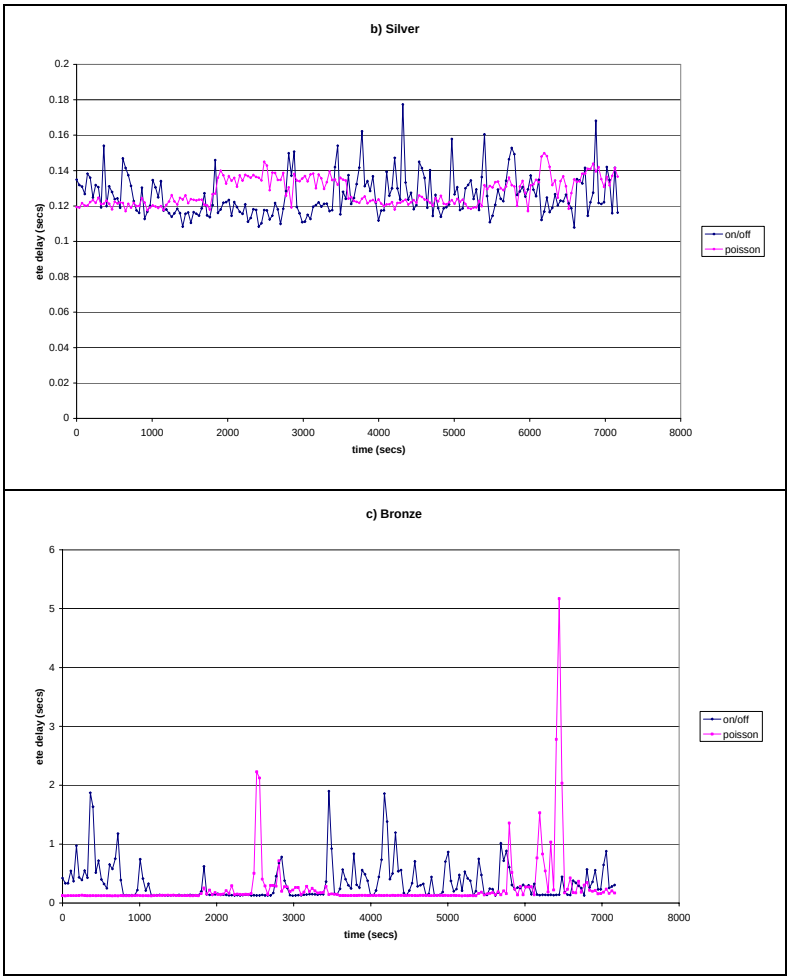


Figure 44: ON/OFF & Poisson Traffic

8.5 Node Level Analysis

Many of the results examine average network performance. However, it is valuable to examine performance at specific nodes – notably those which will suffer excessive congestion or will receive rerouted traffic.

8.5.1 Node 9

With the current network configuration, the effect of the theta factor should be to spread more traffic towards node 9. The critical link of interest is node_9→node_11, ie the link served by queue_4 at this node.

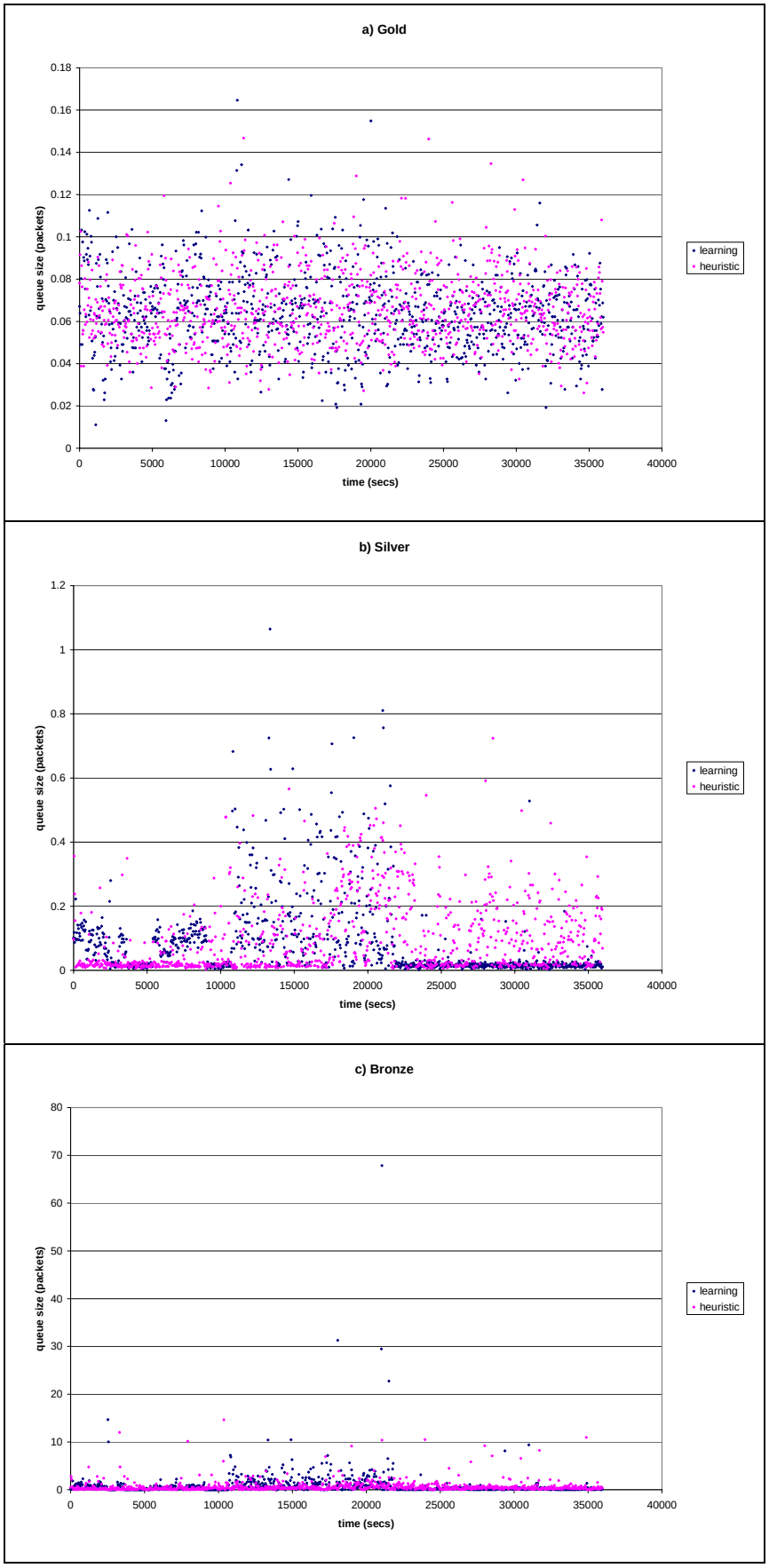
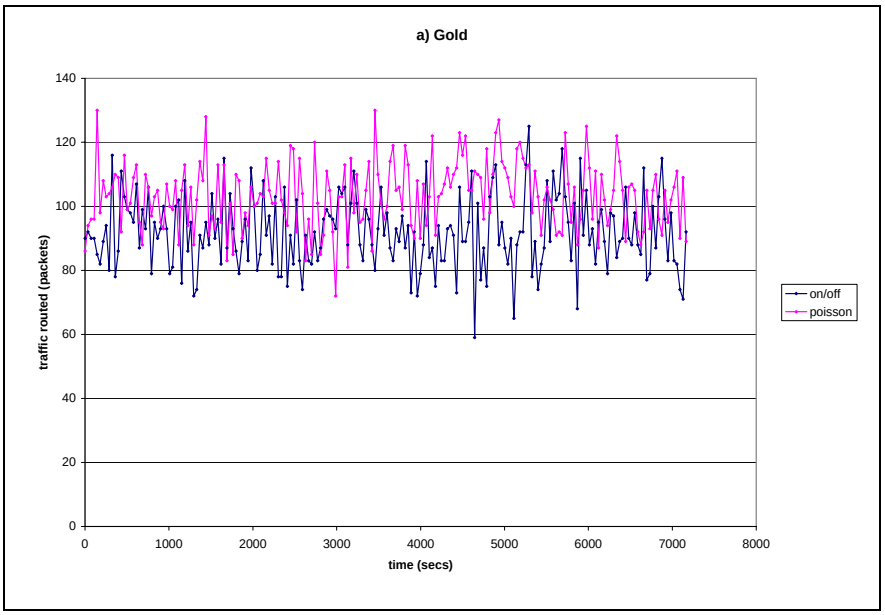


Figure 45: Node 9 Queue 4 - Queue Size

The packet size in the data (ie not signal) traffic subqueues is shown in Figure 45. There is a subqueue for each traffic class, to allow for differential servicing. In these runs the transmission rate was lowered to 75,000, to generate higher congestion. The behaviour in the bronze subqueue is of most interest. The average queue length over the learning run was 0.854 packets, standard deviation of 2.898, peak size 67.9; the heuristic run generated corresponding figures of 0.652, 1.216, 14.6. Again, there is increased variability in the learning simulation.

A shorter simulation was run, increasing the reward figure to eight, in a standard speed network (ie transmission rate of 100,000 bits/sec). Figure 46 shows the volume of traffic routed at node 9, again with much variation in the bronze traffic, although volatility was considerably high for silver Poisson generated traffic (standard deviation of 19.302 for ON/OFF, 54.736 for Poisson).



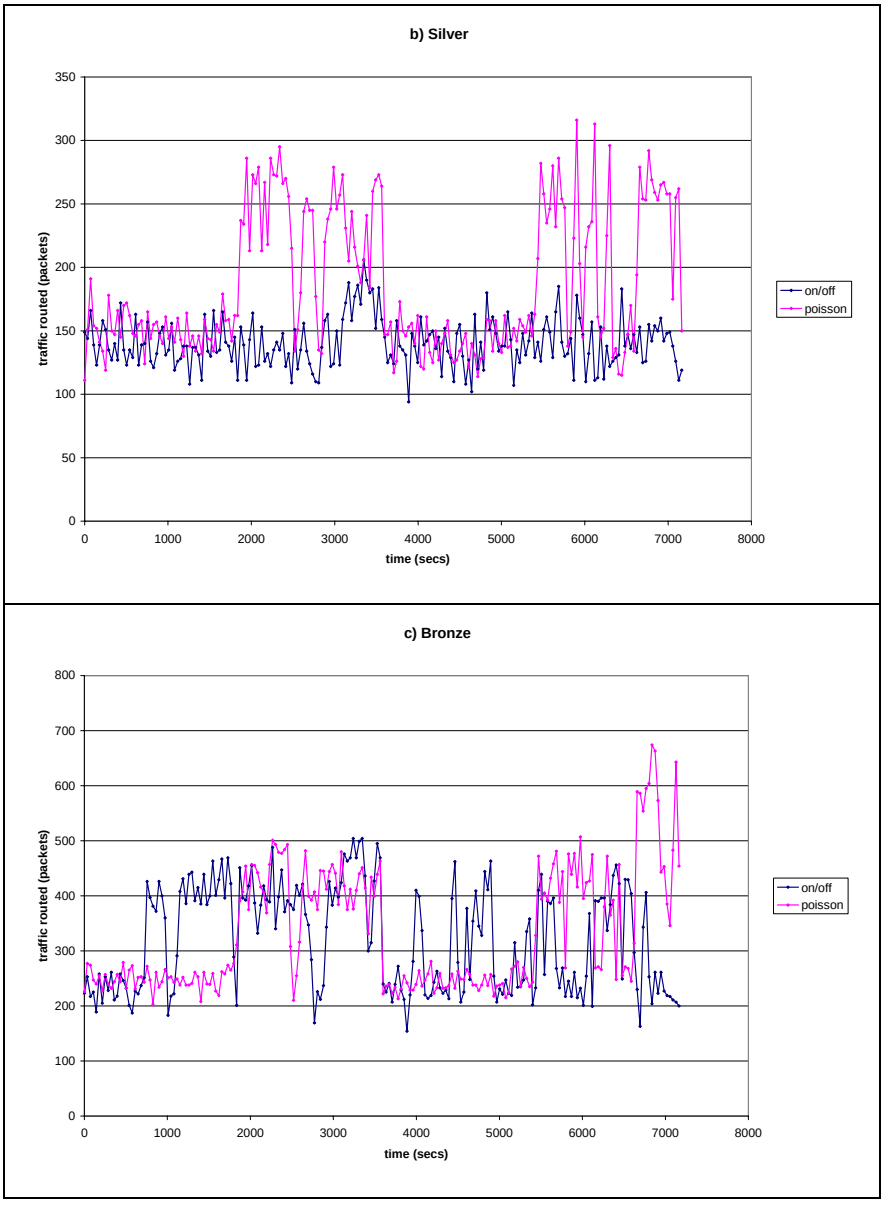
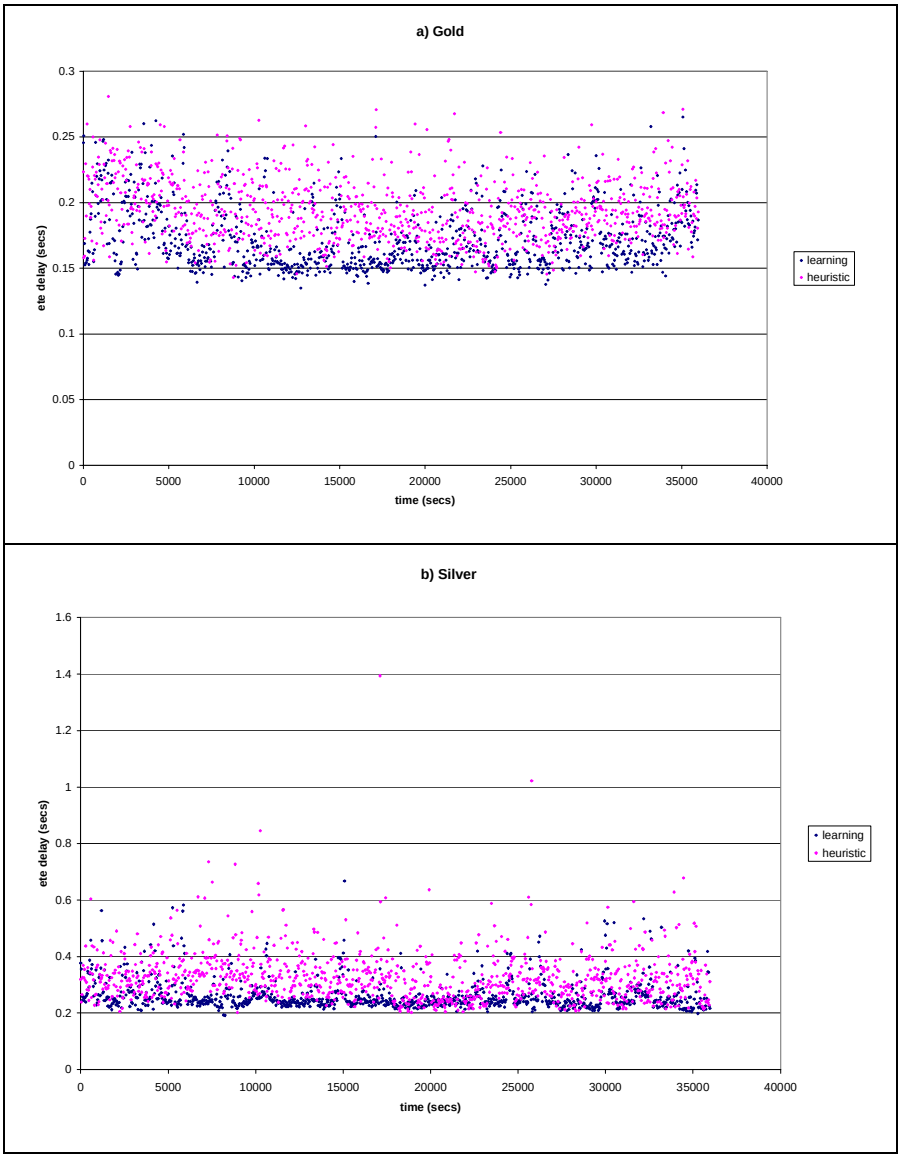


Figure 46: Node_9 Traffic Routed

Several of the spikes occur around 1800, 3600, 5400 seconds (etc). This is most pronounced for the Poisson generated traffic. These times coincide with the OSPF regular update floods, ie at these points nodes received the latest link conditions for the entire network. Since the reward function is biased against flooding (examined in more depth in section 8.7) this suggests it rendered the network relatively irresponsive to traffic conditions.

8.5.2 Node 11

Behaviour at node 11 presents a useful analysis of the effectiveness of any routing policies. The network traffic pattern is deliberately skewed so that all traffic generated from nodes 0 and 7 is sent to node 11. This places stress on the immediate links, notably node_8→node_11 and node_12→node_11. The aim of both the heuristic and the learning (intelligence) is to smear the lower class traffic away from these links by presenting alternative links (eg routing via node_13→node_9) as 'optimal'.



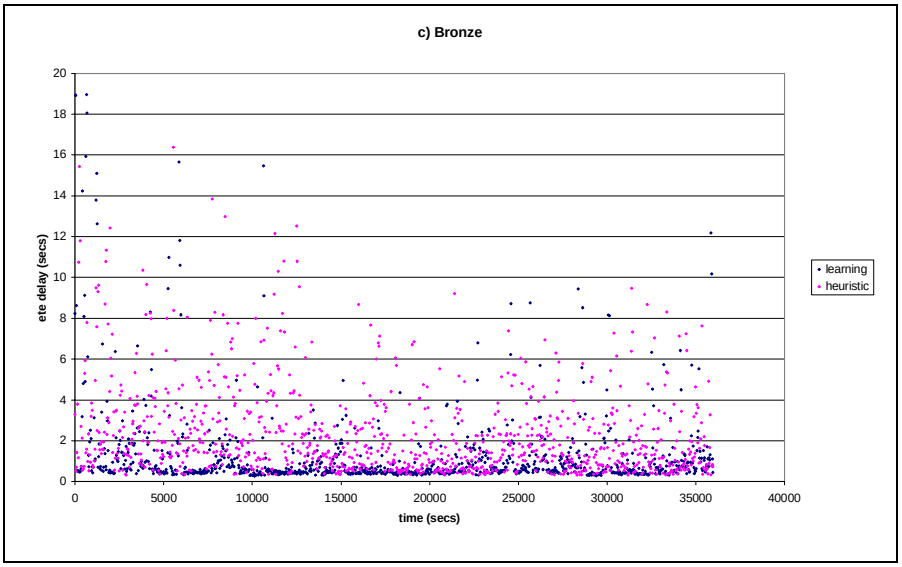


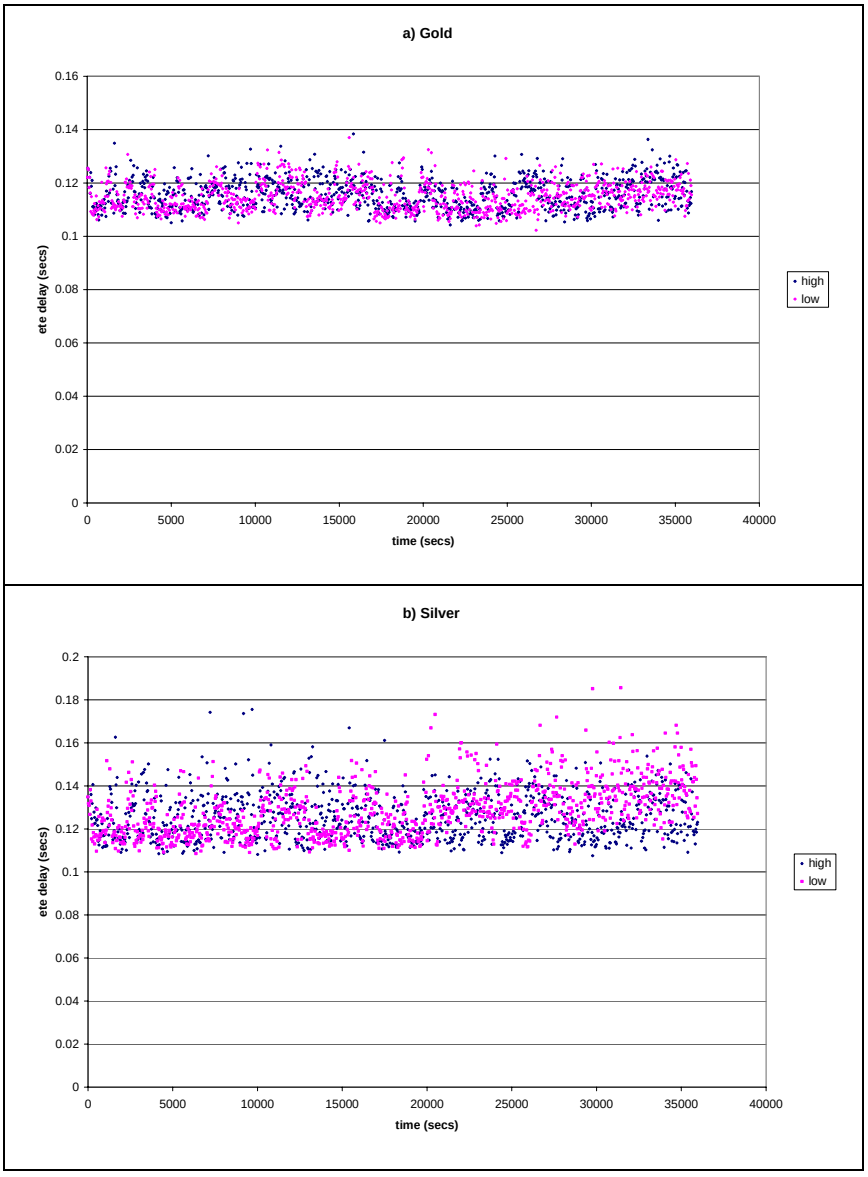
Figure 47: Node 11 Traffic End-To-End Delay

Figure 47 shows the end-to-end delay experienced by all traffic with final destination node_11. The link transmission rate for each link was depressed to 75,000 bits/second in order to add more strain to the network. The average bronze delay (shown in graph c) is lower (1.29 seconds) for traffic carried over the intelligent network, compared to the one operating a heuristic (2.39 seconds). That for silver is again lower (0.263 intelligent compared to 0.328 heuristic), while the difference is less significant for gold (0.194 intelligent compared to 0.173 heuristic). The purpose of the heuristic and learning mechanisms is to ensure that the performance of high-grade traffic is not impaired by the lower-traffic. At the same time the aim is to demonstrate enhanced handling of low-grade (especially best-effort) traffic by spreading it away from the 'optimal' links. Were bronze traffic still sent down these links, higher end-to-end delay would be expected due to disadvantageous handling by the queue servicing mechanism. These results support the proposition that adding the intelligence to the pseudo-delay routing improves the performance of lower grade traffic.

8.6 Calibration of the Fuzzy Sets

The calibration of the fuzzy sets for delay (LOW, MEDIUM and HIGH) for all three traffic classes was critical for system responsiveness. If, for example, these were fixed too low then too many observed delay measurements would have high membership

(μ) in the HIGH set (and conversely few high memberships would be observed in the LOW set). As a result the system would be in permanent flux due to continuous flooding, with limited network convergence. However, if the sets were mapped too high then delays would rarely register – most high membership readings would occur in the LOW set – making the system too sluggish and irresponsive to congestion. Two simulations were run to explore the impact of shifting the fuzzy sets to the right. A rightward shift, as explained above, has the effect of making a wider range of (crisp) delay observations correspond to high membership of a LOW fuzzy set. By extension, the level of observed delay that corresponds to a membership greater than zero of the HIGH fuzzy set is raised.



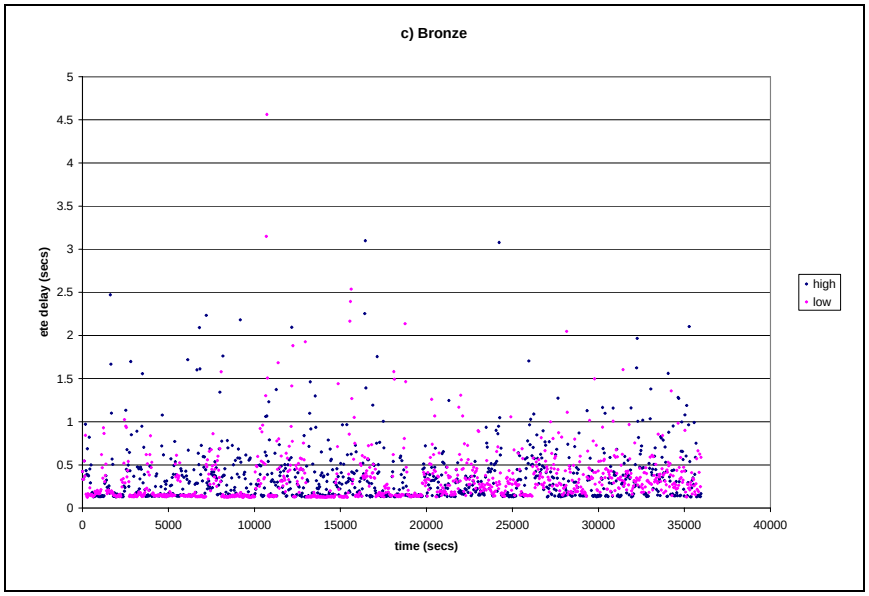


Figure 48: Shifting Fuzzy Sets[lxx]

Since shifting the sets rightward had a minimal effect on the delay figures (gold/silver/bronze average delay of 0.115, 0.128, 0.339 for the low sets compared to 0.116, 0.126 and 0.375 for the high sets) it was decided to employ these higher sets for the simulation results. The rationale behind this was to minimise the number of floods.

8.7 Reward Function

OSPF is characterised as a quiet protocol. Flooding only takes place where necessary – at times of network stress or due to the regular link state update procedure. As a result the reinforcement learning reward function was biased towards not flooding, to avoid the problems associated with network convergence. A potential consequence of this is that network congestion could increase, as the system would be less responsive to congestion. However, work cited earlier [72] indicated that the low-flood/inaccurate database trade-off could be an acceptable price to pay for a quieter network.

[lxx] Simulation Parameters: transmission rate 100,000 bits/second

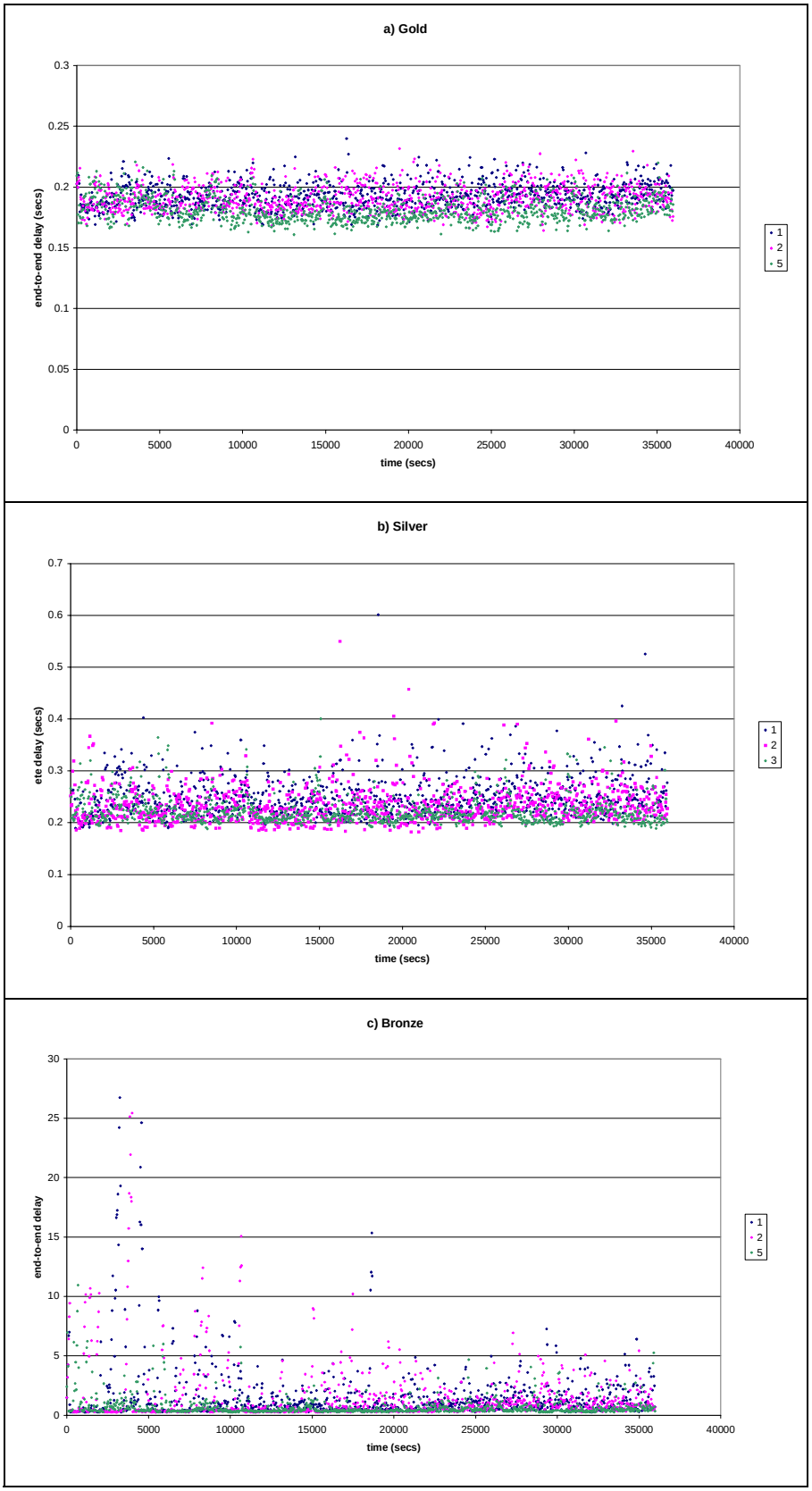


Figure 49: Network Average End-to-End Delay with Shifting Reward

Figure 49 shows the impact on end-to-end delay across the network of manipulating the non-flooding bias in the reward function. Originally the function was set so that a non-flooding action choice returned a reward of one. Later simulations (with the slower transmission rate of 75,000 bits/second) investigated the network impact of setting the non-flooding reward to two, then to five. The maximum reward possible after a flood is kept stationary at one, regardless of the level of congestion. There is a negligible (albeit positive) effect on gold traffic, and a minimal positive effect on silver traffic due to the increase in the reward figure. The most notable finding is the effect on bronze traffic – both volatility and average delay decrease, notably once the reward rises to five. For bronze traffic with reward of 1, then 2, then 5 the average delay was 1.595, 1.467 and 0.712 respectively, with standard deviation of 2.613, 2.526 and 0.908 respectively. Results for reward of eight have been shown earlier, when comparing Poisson with ON/OFF in Figure 44 and when investigating the behaviour at node 9 in Figure 46[lxxi].

Finally, a decaying reward function was introduced to add hysteresis after flooding. The mechanism greatly enhanced the reward for not-flooding LSAs in response to congestion across the link immediately after a flood. This figure then decayed to a minimum of one. The results depicted in Figure 50 (with transmission rate of 100,000 bits/second) can be compared to those in Figure 38 (where reward in the learning model is set to one for not flooding). The average bronze delay in the earlier chart is 0.339 seconds with standard deviation of 0.328, contrasted to an average of 0.241 seconds with standard deviation of 0.144 with the decaying reward function.

[lxxi] Although these shorter simulations were run across a less congested network, with transmission speed of 100,000

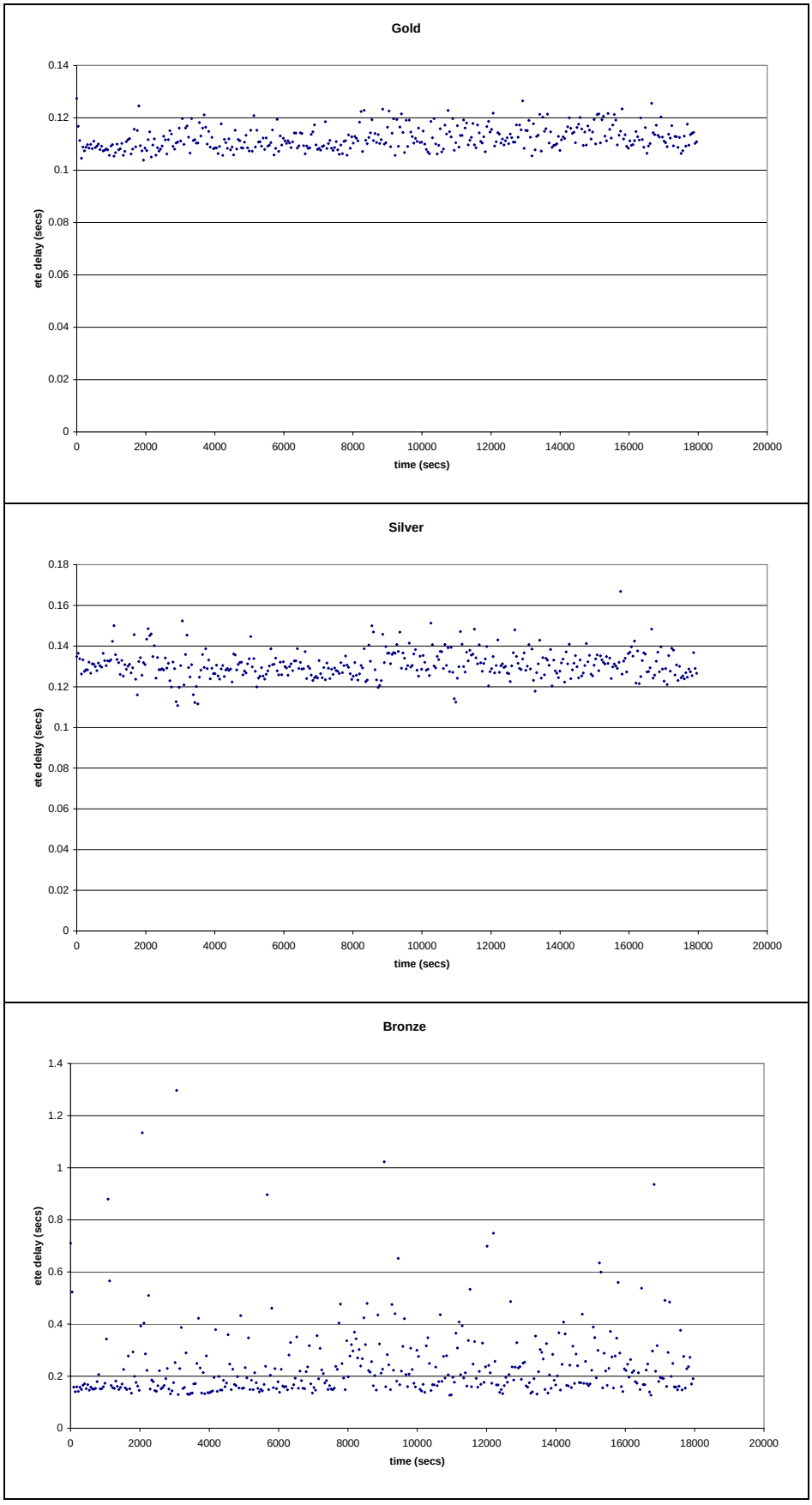
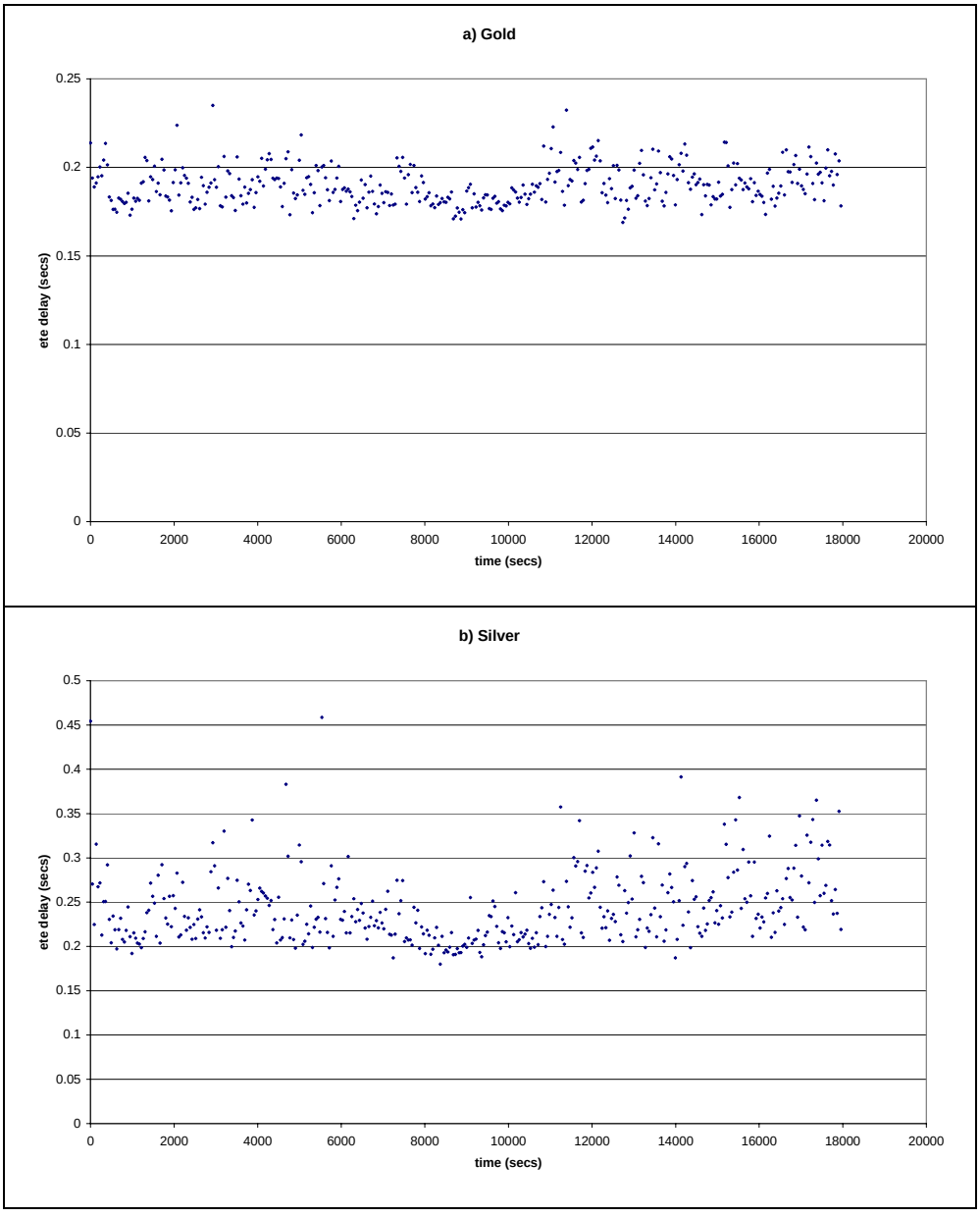


Figure 50: Decaying Reward Function

However, the operation of this decay mechanism was ineffective in more congested networks. When the transmission rate was reduced to 75,000 bits/second the bias towards not flooding resulted in an inability to respond to network conditions. In such networks the system was unable to flood LSAs so traffic continued to be sent along overcrowded links. The results of this, showing network average end to end delay, are depicted in Figure 51. Here bronze average end to end delay is 2.318 seconds with a standard deviation of 3.745 seconds.



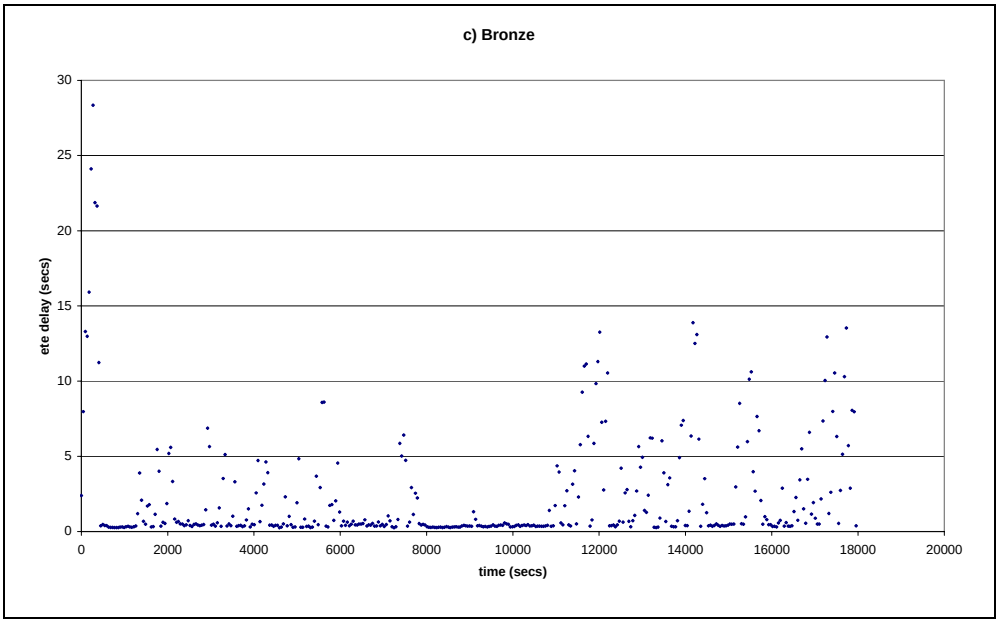


Figure 51: Decaying Reward Function over Slow Links

9 Discussion and Further Work

A significant proportion of this thesis was consumed in establishing how agents could operate across IP networks. This involved distinguishing an intelligent or software agent from many entities that are termed 'agents', especially in the protocol literature. Yet an agent is not a silver bullet [199]; different solutions exploit different agent desiderata and other non-agent solutions may provide equal flexibility. This resulted in a more nuanced argument for an agent-like solution. An illustration in this work is leveraging the existing communication structure – the 'agents' deployed into each node currently communicate using protocol traffic, ie LSUs/LSAs – rather than developing a complex communication protocol. Through exploiting the existing protocol, the results of the learning garnered at each node could be shared, thus providing an agent-like (or agent-based) solution.

The enhancements / results from this research sought to confirm two premises: that sub-optimal routing of traffic in a multi-class network is a viable resource management strategy; that adding intelligence is beneficial. The purpose underlying this section is to validate whether these have been met. Both concerns – the role of agents in IP networks and the success of the sub-optimal routing strategy – will be addressed through answering two questions:

"Is OSPF an agent-based system?"

"Does this work demonstrate an agent-based system?"

Although section 4.3 specifically investigated the use of the term agent in network protocols OSPF makes no explicit claim to be employing agents. However, it could be argued that (superficially) many of the properties of this routing protocol overlap with an agent system. Furthermore, since both the pseudo-delay heuristic and the intelligent strategy piggyback on the OSPF messaging protocol can a clear distinction be made between these approaches and OSPF? Not only must it be proved that the intelligent modification is more successful than OSPF, but that by its design it forms an agent-based approach.

In [100] the authors examined the pragmatics underlying the development of agent systems. Agents, they argued, were another addition to the wealth of software

engineering abstractions that aided the management of complex systems: "*Just as many systems may naturally be understood and modelled as a collection of interacting but passive objects, so many other systems may be naturally understood and modelled as a collection of interacting autonomous agents*". However, they identified that this distinction could be lost, for example at the time of writing there was in their opinion a tendency to view ***any*** distributed system as a multi-agent system. While rejecting complexity of design, eg overuse of AI when not necessary, for its own sake they nevertheless demanded some AI component to agents. This will form a partial guideline to the resolution of whether OSPF and the enhancements presented here operate as agent systems.

A network running OSPF is comprised of distributed nodes operating a discovery, communication and authentication mechanism. Nodes are unaware of network topology when the system originates, ie there is no centralised network-view imposed on the distributed nodes. Through exchanging Hello packets to establish neighbours, followed by LSAs each node establishes an identical topological view (represented by the link state database) and can then generate a routing table. The communication mechanism for example allows nodes to discover: when links are down, when link costs have changed, when packets have been received. Whether the communication entity is thought of as a node/router or abstracted to an object operating at that node, it is still insufficient to describe this as an agent system. Although OSPF is very powerful, and although it does not rely on a centralised object imposing a fixed topology on the nodes, it is nevertheless an AI-free (ie not an intelligent) distributed system.

Very recent (as yet unpublished) developments in agent research promote 'agents as a design metaphor' [200]. Researchers focussing on the applicability of software agents have rejected the bottom-up approach to agent definition discussed in this thesis – ie one building on characteristics such as autonomy, reactiveness, proactiveness – in favour of a top-down approach. As such this top-down approach concentrates the analysis on the design methodologies, architectures and supporting infrastructures required for complex, dynamic (often heterogeneous) environments. The bottom-up approach would perhaps focus on whether OSPF routers are truly autonomous, or

whether the sociability demands of the Wooldridge/Jennings approach is satisfied by IP communication protocol.

Yet having established that OSPF could not legitimately be called an agent system, without making such a term redundant, it may not be necessarily evident that the enhancements presented in this work could legitimately be seen as agent-based. Certainly the aim behind the pseudo-delay heuristic was to establish the validity of spreading less vital traffic away from 'optimal' routes rather than investigating the role of agents. However, the goal of the intelligent routing was to present an agent-based strategy for routing. A 'fragile' model of agents was presented (using the bottom-up approach) in section 4.6, rejecting the full complexity of communication / negotiation strategies and protocols. Learning was proposed as a key agent characteristic. This sets the intelligent routing model apart from a straightforward distributed system – this approach could instead be said to successfully negotiate the agent-level pitfall of failing to employ AI. Nevertheless, perhaps the richness of communication and support architectures that form the top-down approach are lacking. A limitation to considering this agent system as a design metaphor is that heterogeneity is compromised in connectionless networks: nodes must share their network vision in order to avoid routing loops. However, in section 9.2, future work that could enhance the effectiveness of the approach adopted is discussed. Adding further, longer-term learning to the system will require a more elaborate agent architectural design both within and between nodes.

9.1 Evaluation of Results

It is clear from the results that adding intelligence to IP routing does not produce overwhelming advantages. However, it will be argued that there is sufficient evidence to support further investigation into applying agent-based techniques.

The early research developed in section 5 investigated manipulating the perceived cost metrics across links in order to spread lower-cost traffic away from the optimal paths. Favourable results encouraged adding an agent flavour to this research, ie adding intelligence in the form of learning. Reinforcement learning was chosen due to

the advantage of not requiring the imposition of a preconceived framework or model. The necessary exploratory element of this form of learning, however, may explain the relative variability of results when compared to the heuristic. Strategies to ameliorate this are discussed in the future work section.

In the later set of simulations (ie those presented as results in section 8) two models of OSPF are presented: the highly irresponsive benchmark model and one with both high and low watermark thresholds. The benchmark model forms a useful control, to examine how the network performs without responding to any traffic surges. However, the watermark version represents a more critical challenge to any enhancements. It especially presents a useful critique of the current learning model as the latter is founded around learning when to flood (and to a lesser extent the value of the theta factor). Here flooding is in response to increased congestion, but because of the state space used the learning model cannot differentiate between a flood that increases link cost and one that decreases it (ie when network congestion has resolved). Thus, should link cost fall to a level below that which provoked a flood no LSA would be flooded resetting the cost. An advantage of this is it helps minimise flooding and the associated convergence overhead, but it limits responsiveness.

In the standard network, ie with link transmission rate of 100,000 bits/second the OSPF mechanism appeared the most successful strategy. However, adding strain to the network revealed its critical shortcoming – without the sophistication of the heuristic or learning mechanisms its strategy proved too crude. Although the learning mechanism outperforms OSPF in a congested network end-to-end delay figures were still poor. However, with very high utilisation over several links, such figures would be expected.

Some results indicate that the learning outperforms the heuristic (notably those for end-to-end delay for node_11); others indicate higher variability. This is, as discussed, most likely due to the exploratory nature of the learning. Manipulating the reinforcement reward function was shown to improve performance, most significantly when biasing the mechanism against flooding. However, as was demonstrated in the more congested network this can lead to the system failing to respond adequately to

network stress. Thus, although the results point to the advantage of adding intelligence they also suggest that the current solution is not intelligent enough.

9.2 Future Work

Having established that an agent system may prove effective in connectionless networks, future work is required to investigate whether further responsiveness can be added to enhance system optimisation. Such work would involve developing both a richer agent architecture and an augmented learning strategy. Arguments for giving inter-AS interactions an agent label are possibly more valid – feasibly, future work could explore inter-AS routing although the focus in this section, as within this project, will be within an AS.

Ideally additional agent behaviour would have been extended to each node – although mobile agents were considered outside the scope of this research future work would allow monitoring ants (ie those operating in the higher layer) to be sent from each node. These would operate at a strategic level, to complement the current reinforcement learning mechanism. Earlier architectures under consideration had also incorporated a centralised Network Agent responsible for longer-term learning (eg Bayesian Reasoning). The findings from the lower-level learning agents would be fed to this agent. Should this agent fail the network could still function, so robustness would be guaranteed.

A shortcoming of the reinforcement learning algorithm presented here is that FQ values are necessarily short term, being updated every time step. This may be appropriate in a smooth (Poisson) environment, but in a bursty Internet environment it may prove advantageous to feed the results of the learning into more long-term rules. Although beyond the current scope of this research, future work could investigate the advantage of incorporating case-based reasoning [201] into the learning. Case-based reasoning agents would be distributed to each node (and employed in place of the agents proposed in the previous paragraph), leading to a potentially more powerful agent architecture.

Additionally, many of the limitations to the learning are a consequence of the limited state space adopted. Although, as stated early, a benefit of adding fuzzy sets to the reinforcement learning algorithm is to control the expansion of state space, it could be advantageous to include extra parameters into the learning. Incorporating further information such as the time since the last flood and then making the rewards dependent on this time could prove useful. Another feature to factor in would be high and low watermark thresholds. The case-base reasoning agents could be used to formulate, for example, high and low rules. These rules in turn could be corrected by the findings of the reinforcement learning agents, ensuring the model worked effectively in a dynamic environment.

Finally it would be useful to explore a wide range of network topologies to test for scalability and to discover whether surges would invalidate the benefits of spreading traffic away from the optimal paths. A more realistic router architecture could also be included in the simulation by making the queue buffers finite. Currently the buffers in each node are infinite. This was considered a necessary artifice since the learning mechanism concentrated on a single cost metric: end-to-end delay. The rationale behind not imposing finite queues was that this could result in packets being dropped at times of high link utilisation. In turn this packet loss could mask the traffic stress on the network, decreasing (and hence masking) the delay rates.

10 Summary

Section 4.4 queried the viability of employing the term agent in network environments. The over usage of this term had led, it was argued, to redundancy – anything could be, and indeed claimed to be, an agent. Mere agency was often sufficient to some authors. This was perhaps not surprising – even within the agent community can prove difficult to agree on what constitutes this special software engineering abstraction. Nevertheless, by the end of the chapter the notion of 'agent-like' had been explored. This has resulted in a recasting of the agent:network relationship, with an attempt to answer a new question:

> *What elements of agent-like behaviour/practice can prove advantageous in a tightly-coupled distributed environment?*

Autonomy, the prevailing characteristic of an agent, is undermined in such environments: nodes in an AS need identical link state databases to run OSPF effectively. Communication already exists, albeit in a protocol form, without the flexibility promised by agent designers/theorists. Since the means by which the nodes in the network can respond to changes in the environment is limited to flooding, ie sharing new link state information with all other nodes, the means by which an 'agent' can act upon the environment is by affecting when such floods are triggered and the contents of the LSA (ie manipulating link costs). This still concurs with the notion of agents as "situated problem solvers" in [202].

Of course, having limited autonomy, communication and social interaction imposed by the close coupling of the system, this only left learning as a vehicle for augmenting behaviour. This thesis looked at viable approaches to learning in the constrained environment, but even here the problem had to be formulated essentially as a control problem. There is no scope for radically new behaviours. Reinforcement learning does have exploratory steps but only within a defined range and reward framework.

Results indicated that routing sub-optimally, using pseudo-delay figures, could result in improved network optimality. This finding is pertinent given that so-called optimal

strategies have been demonstrated to result in non-optimal networks. Thus what may appear as a contrary, indeed contradictory, strategy may prove an efficient means of traffic engineering. Adding learning, ie the agent behaviour, increases the responsiveness of the pseudo-delay mechanism, with potentially greater sensitivity through expanding into a more complex agent learning architecture. Whether termed agent-like or agent-based, adding learning to a tightly-coupled network is here presented as an advantageous strategy.

Appendix A: Simulation Verification

This appendix lists further examples of simulation verification mechanisms and experiments. The mechanisms can be grouped into:

1. Halting the simulation
2. Printouts to screen – using printf() statements.
3. Printouts to file

Combinations of the above approaches were employed to verify accurate simulation. Validation experiments were run to confirm the changes in link costs and the correct propagation of LSAs

Printouts were applied to trace, for example, correct procedure when propagating LSAs. When a node propagates a single LSA (ie not when performing a periodic flood for all link states) the procedure should be to update the node's own linkstate database, generate the new routing table, create an LSA, encapsulate it and send it to neighbouring nodes. Printouts to the screen inside each function would demonstrate the order of function calls and routing table update. Simulations would be terminated – using the OPNET procedure op_sim_end() – to allow analysis of screen printout. Additionally termination would be used to trap illegal states – for example incorrect LSA delivery, unidentified LSAs, incorrectly terminating algorithms.

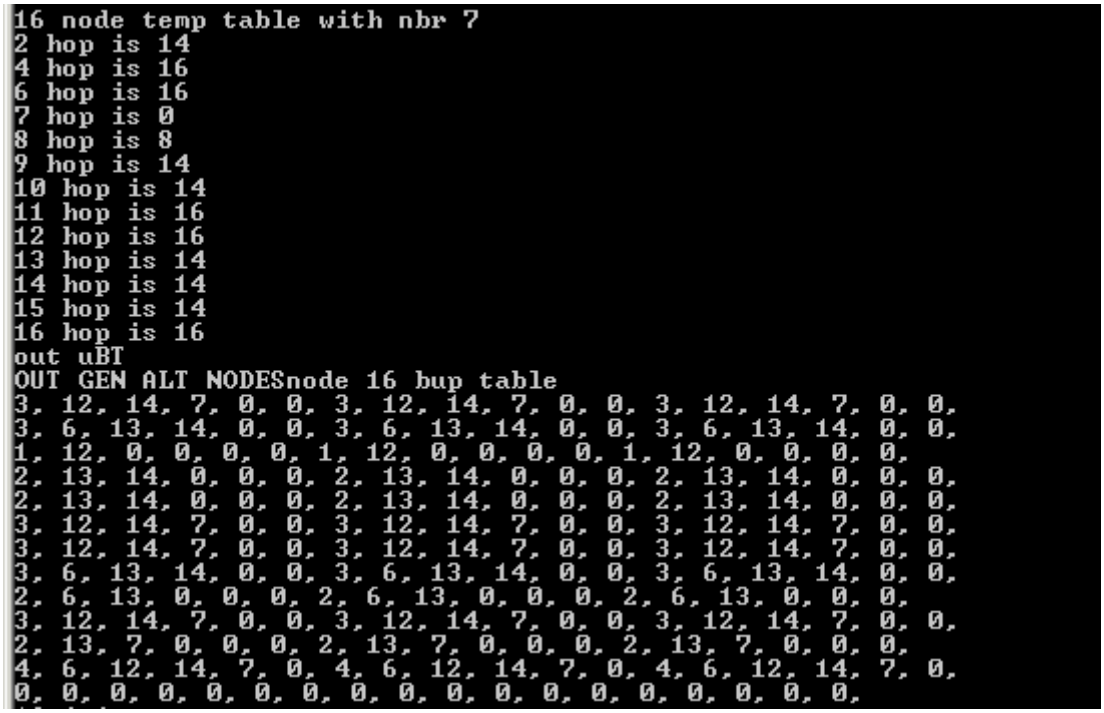
File printouts are employed to trace, for example, correct allocation and deallocation of resources (pointers and packets), network utilisation, how often floods are triggered and in response to what network conditions

Artificially rising link cost

The purpose of this is to demonstrate that modifying a link cost alters the spf calculation. This causes all routing tables to modify their recommended next hop (where appropriate) for destinations that previously routed across this prohibitively expensive link. This exercise also demonstrates the propagation of each LSA (showing where each LSA has travelled and when it gets destroyed to prevent continuous flooding)

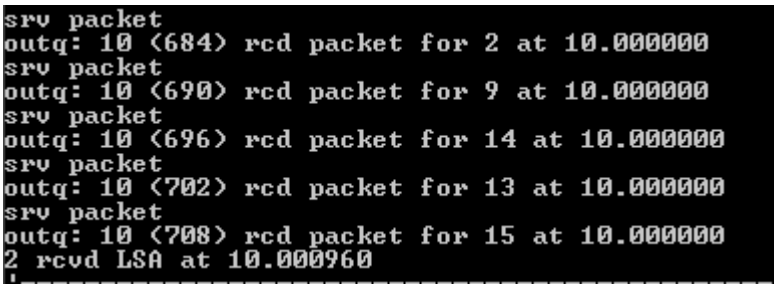
Back up Table

The following screen grab provides an example backup table, for node 14 (numbered 16 by the simulation kernel):



The first entry shows the number of alternative next hops for a given destination (corresponding to the destinations in the traditional routing table)

The next screen capture demonstrates that the out-queues for node 10 (ie 8 in the topology diagram) have received packets from the core processor for all five neighbours. The simulation is terminated once the first neighbour receives its LSA.



To confirm correct propagation of LSAs their paths through the network were traced by printing out to a file. This demonstrates that after the original node (10) sends out

the LSA to its five neighbours all nodes receive the LSA and no node sends it out more than once (ie the flooding is self-limiting). On receipt of an LSA (after confirming this is an original LSA) a node forwards it on to all its neighbours (including the one which previously forwarded it). However, the nodes with only one neighbour do not propagate LSAs further[lxxii].

10 sending LSA to 2	14 forwarding LSA to nbr 16	16 forwarding LSA to nbr 13
10 sending LSA to 9	14 forwarding LSA to nbr 7	16 forwarding LSA to nbr 14
10 sending LSA to 14	11 forwarding LSA to nbr 4	16 forwarding LSA to nbr 7
10 sending LSA to 13	11 forwarding LSA to nbr 12	13 forwarding LSA to nbr 14
10 sending LSA to 15	11 forwarding LSA to nbr 15	13 forwarding LSA to nbr 16
2 forwarding LSA to nbr 15	11 forwarding LSA to nbr 13	13 forwarding LSA to nbr 10
2 forwarding LSA to nbr 10	8 forwarding LSA to nbr 14	13 forwarding LSA to nbr 11
15 forwarding LSA to nbr 2	8 forwarding LSA to nbr 7	12 forwarding LSA to nbr 6
15 forwarding LSA to nbr 11	7 forwarding LSA to nbr 16	12 forwarding LSA to nbr 11
15 forwarding LSA to nbr 10	7 forwarding LSA to nbr 14	12 forwarding LSA to nbr 16
14 forwarding LSA to nbr 8	7 forwarding LSA to nbr 8	6 forwarding LSA to nbr 12
14 forwarding LSA to nbr 10	16 forwarding LSA to nbr 6	6 forwarding LSA to nbr 16
14 forwarding LSA to nbr 13	16 forwarding LSA to nbr 12	

The following printout to file shows the updated routing table at node 10 (numbered 8 in the topology). At time 0, packets to the neighbouring node 14 are routed directly. However, at time 10, although observed delay is negligible an artificial load is imposed on the link 10-14. The new routing table shows that packets from 10 to 14 are now routed via hop 13 as link 10-14 is prohibitively expensive.

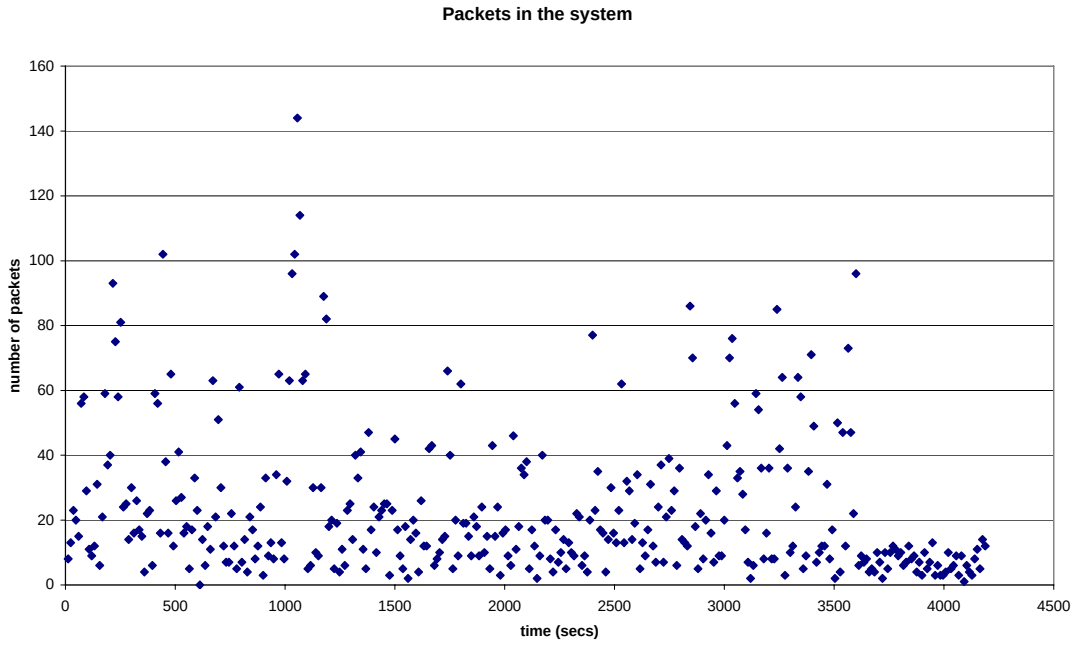
```
time: 0.000000
2 hop is 2
4 hop is 13
6 hop is 13
7 hop is 14
8 hop is 14
9 hop is 9
10 hop is 0
11 hop is 13
12 hop is 13
13 hop is 13
14 hop is 14
```

[lxxii] This means their neighbour does not receive an indirect acknowledgement, but that is acceptable in such a controlled network

```
15 hop is 15
16 hop is 13

at 10.000000: node 10 to link 14:
 delay G: 0.000000, S: 0.000000, B: 0.000000, theta 0.000000
delay metrics: 5000000, 5000000, 5000000
MTRC sending LSA, seq 0 re theta
time: 10.000000
2 hop is 2
4 hop is 13
6 hop is 13
7 hop is 13
8 hop is 13
9 hop is 9
10 hop is 0
11 hop is 13
12 hop is 13
13 hop is 13
14 hop is 13
15 hop is 15
16 hop is 13
```

The following diagram shows the number/volume of packets in the system (with points taken at 12 second intervals). This is used to verify the packet creation and destruction process.



Author's Publications

Bourne, Rachel, Shoop, Karen & Jennings, Nicholas "Dynamic Evaluation of Coordination Mechanisms for Autonomous Agents" in 'Progress in Artificial Intelligence', LNAI 2258, December 2001, ISBN 3-540-43030-X

Shoop, K. & Bigham J. "A Hybrid Agent-Based Architecture for Network Resource Management", Proceedings PGNET 2002, Liverpool, UK, August 2002

Shoop, K., Bigham, J. & Phillips, C. "Resource management employing pseudo-delay for IP networks", Proceedings 1st IEEE Consumer Communications and Networking Conference (CCNC), Las Vegas, Nevada, USA, January 2004

Shoop, Karen, Phillips, Chris & Bigham, John "Agent-Based Sub-Optimal Routing in Multi-Class IP Networks", Proceedings International Conference on Computing, Communications and Control Technologies: CCCT'04, Austin, Texas, USA, August 2004

Shoop, K., Bigham, J. & Phillips, C. "Resource Management Employing Learned Pseudo-Delay for Multi-Service IP Networks", Proceedings 10th European Conference on Networks and Optical Communications, NOC2005, London, UK, July 2005

References

[1] Ma, Qingming & Steenkiste, Peter "Supporting Dynamic Inter-Class Resource Sharing: A Multi-Class QoS Routing Algorithm", INFOCOM '99, New York, March 1999

[2] Hochkar, Hedia, Ikenaga, Takeshi, Kawahara, Kenji & Oie, Yuji "Multi-class QoS Routing Strategies Based on the Network State", Computer Communications, Vol. 28, Issue 11, 5 July 2005

[3] Nwana, Hyacinth S. "Software Agents: An Overview", Knowledge Engineering Review, Vol. 11, No. 3, October/November 1996

[4] Nichols, K., Blake S., Baker, F. & Black, D. "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, December 1998

[5] Deering, S. & Hinden, R. "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998

[6] Huitema, Christian "IPv6: The New Internet Protocol", 2nd Edition, Prentice Hall, 1998, ISBN 0-13-850505-5

[7] Moy, J. "OSPF Version 2", RFC 1247, July 1991

[8] Doyle, Jeff "Routing TCP/IP Volume I", Cisco Press, 2001, ISBN 1-57870-041-8

[9] Huitema, Christian "Routing in the Internet", 2nd Edition, Prentice Hall, 2000, ISBN 0-13-022647-5

[10] Weiss, Mark Allen "Data Structures and Algorithm Analysis in C", The Benjamin/Cummings Publishing Company, 1993, ISBN 0-8053-5440-9

[11] URL: http://www.cisco.com/univercd/cc/td/doc/product/software/ios113ed/113aa/113aa_2/58cfeats/ospfpace.htm

[12] Cisco "Configuring OSPF", 1997

[13] URL: www.cisco.com/en/US/products/sw/iosswrel/ps1831/products_configuration_guide_chapter09186a00800b3f2b.html

[14] Awduche, D et al "Requirements for Traffic Engineering over MPLS", RFC 2702, Sept 1999

[15] ITU-T Recommendation E.800 "Terms and Definitions Related to Quality of Service and Network Performance Including Dependability", August 1993

[16] E. Crawley, R. Nair, B. Rajagopalan and H. Sandick "A Framework for QoS-based Routing in the Internet", RFC 2386, August 1998

[17] Chalmers, Dan & Sloman, Morris "A Survey of Quality of Service in Mobile Computing Environments", IEEE Communication Surveys, 2nd Quarter 1999

[18] van der Zee, Martin & Heijenk, Geert "Quality of Service in Bluetooth Networking. Part I", Technical Report University of Twente, TR-CTIT-01-01, January 2001, http://ing.ctit.utwente.nl/WU1/

[19] de Castro, Miguel F., M'hamed, Abdallah, Gaiti, Dominique & Oliveira, Mauro "Simulated Internet Traffic Behaviour under Different QoS Management Scenarios", Proceedings ISCC'03, Antalya, Turkey, June/July 2003

[20] Lu, Hui-Lan & Faynberg, Igor "An Architectural Framework for Support of Quality of Service in Packet Networks", IEEE Communications Magazine, Vol. 41, No. 6, June 2003

[21] Bonald, T., Ouselati-Boulahia, S. & Roberts, J. "IP traffic and QoS control: towards a flow-aware architecture", Proceedings WTC 2002, Paris, September 2002

[22] Schollmeier, Gero & Winkler, Christian "Providing Sustainable QoS in Next-Generation Networks", IEEE Communications Magazine, Vol. 42, No. 6, June 2004

[23] ITU-T Recommendation "End-User Multimedia QoS Categories", G.1010, November 2001

[24] ITU-T Recommendation "Network Performance Objectives for IP-based Services", Y.1541, May 2002

[25] Cuthbert, L. G. & Sapanel, J. C. "ATM, The Broadband Telecommunications Solution", IEE Telecommunications Series 29, 1993, ISBN 0-85286-815-9

[26] Roughgarden, Tim & Tardos, Éva "How Bad is Selfish Routing?" Journal of the ACM, Vol. 49, Issue 2, March 2002

[27] Fraleigh, C., Tobagi, F. & Diot, C. "Provisioning IP Backbone Networks To Support Latency Sensitive Traffic" Proceedings INFOCOM 2003, San Francisco, USA, April 2003

[28] Smith, J.M. "Selected Challenges in Computer Networking", Computer, Vol. 32, Issue 1, January 1999

[29] Odlyzko, A. M. "Data Networks are Lightly Utilized and Will Stay That Way", Review of Network Economics, Vol. 2, Issue 3, September 2003

[30] Moore, Sean S. B. & Siller, Curtis A. Jnr "Packet Sequencing: A Deterministic Protocol for QoS in IP Networks", IEEE Communications Magazine, Vol. 41, No. 10, October 2003

[31] Christin, Nicolas & Liebeherr, Jorg *A QoS Architecture for Quantitative Service Differentiation*, IEEE Communications Magazine, Vol. 41, No. 6, June 2003

[32] Huston, Geoff, “Internet Performance Survival Guide: QoS Strategies for Multiservice Networks”, Wiley, 2000, ISBN 0-471-37808-9

[33] Jain, R “Myths About Congestion Management in High-Speed Networks”, Internetworking: Research & Experience, Vol. 3, 1992

[34] Feldman, Anja et al “Performance of Web Proxy Caching in Heterogeneous Bandwidth Environments”, Proceedings INFOCOM ’99, Vol. 1, New York, March 1999

[35] Yang, Shanchieh Jay & de Veciana, Gustavo “Enhancing Both Network and User Performance for Networks Supporting Best Effort Traffic”, IEEE/ACM Transactions on Networking, Vol. 12, No. 2, April 2004

[36] Feldmann et al “Deriving Traffic Demands for Operational IP Networks: Methodology and Experience”, IEEE/ACM Transactions on Networking, Vol. 9, No. 3, June 2001

[37] Gozdecki, Janusz, Jajszczyk, Andrzej & Stankiewicz, Rafal “Quality of Service Terminology in IP Networks”, IEEE Communications Magazine, Vol. 41, No. 3, March 2003

[38] Giresh, Muckai K. “Quality of Service in the Internet: The State-of-the-art and Challenges”, Proceedings of IEEE 38th Conference on Decision and Control, Phoenix, USA, December 1999

[39] Braden, R., Clark D. & Shenker S. “Integrated Services in the Internet: an overview”, RFC 1633, June 1994

[40] Braden, R. et al “Resource Reservation Protocol (RSVP), version 1 – functional specification”, RFC 2205, September 1997

[41] Mankin, A. (ed) et al “Resource ReSerVation Protocol (RSVP) Version 1 Applicability Statement Some Guidelines on Deployment”, RFC 2208, September 1997

[42] Blake, S. et al “An Architecture for Differentiated Services”, RFC 2475, December 1998

[43] Liao, Raymond R.-F. & Campbell, Andrew T. “Dynamic Core Provisioning for Quantitative Differentiated Services”, IEEE/ACM Transactions on Networking, Vol. 12, No. 3, June 2004

[44] Chen, Yang, Qiao, Chunming, Hamdi, Mounir & Tsang, Danny H. K. “Proportional Differentiation: A Scalable QoS Approach”, IEEE Communications Magazine, Vol. 41, No. 6, June 2003

[45] Machiraju, Sridhar, Seshadri, Mukund & Stoica, Ion “A Scalable and Robust Solution for Bandwidth Allocation”, Proceedings of IWQoS’02, New York, May 2002

[46] Achir, Nadjib et al “Active Technology as an Efficient Approach to Control DiffServ Networks: The DACA Architecture”, LNCS 2496, 2002, ISSN 0302-9743

[47] Huston, G. “Next Steps for the IP QoS Architecture”, RFC 2990, November 2000

[48] Stoica, Ion & Zhang, Hui “Providing Guaranteed Services Without Per-flow Management”, Proceedings of ACM SIGCOMM ’99, Boston, USA, August 1999

[49] Welzl, Michael & Franzens, Leopold “Scalability and Quality of Service: a Trade-off?”, IEEE Communications Magazine, Vol. 41, No. 6, June 2003

[50] Rosen, E., Viswanathan, A. & Callon, R. “Multiprotocol Label Switching Architecture”, RFC 3031, January 2001

[51] Jamoussi, B. et al “Constraint-Based LSP Setup Using LDP”, RFC 3212, January 2002

[52] Awduche, D. et al “RSVP-TE: Extensions to RSVP for LSP Tunnels”, RFC 3209, December 2001

[53] Le Faucheur, F. et al “Multi-Protocol Label Switching (MPLS) Support of Differentiated Services”, RFC 3270, May 2002

[54] Estrin, Judy “Clouds Versus Strings: why IP will continue to provide the foundation of the Internet”, White Paper, Packet Design Inc, 2000

[55] URL: http://www.wirelessiq.info/content/qa/4.html

[56] Chen, Shigang & Nahrstedt, Klara “An Overview of Quality of Service Routing for Next-Generation High-Speed Networks: Problems and Solutions”, IEEE Network, Vol. 12, Issue 6, November/December 1998

[57] Lorenz, Dean H. & Orda, Ariel “QoS Routing in Networks with Uncertain Parameters”, IEEE/ACM Transactions on Networking, Vol. 6, No. 6 December 1998

[58] Labovitz, C., Malan, G. R. & Jahanian, F. “Internet Routing Instability”, IEEE/ACM Transactions on Networking, Vol. 6, No. 5, October 1998

[59] Apostolopoulos, G. et al “QoS Routing Mechanism and OSPF Extensions”, RFC 2676, August 1999

[60] Ma, Q. & Steenkiste P. “Quality of Service Routing for Traffic with Performance Guarantees” Proceedings of IFIP 5th International Workshop of Quality of Service, New York, May 1997

[61] Apostolopoulos, G., Guérin, R., Kamat, S. & Tripathi, S. “Improving QoS Routing Performance Under Inaccurate Link State Information”, Proceedings of 16th International Teletraffic Congress (ITC’16), Edinburgh, UK, June 1999

[62] Guérin, Roch A. & Orda, Ariel “QoS Routing in Networks with Inaccurate Information: Theory and Algorithms”, IEEE/ACM Transactions on Networking, Vol. 7, No. 3, June 1999

[63] Chu, Jian, Lea, Chin-Tau & Wong, Albert “Cost-based QoS Routing”, Proceedings of ICCCN 2003, Dallas, USA October 2003

[64] Das, S. et al “A QoS Network Management System for Robust and Reliable Multimedia Services”, Proceedings of Multimedia on the Internet, MMNS 2002, LNCS 2496, 2002, ISSN 0302-9743

[65] Lim, S.H., Yaacob, M.H., Phang, K.K. & Ling, T.C. “Traffic Engineering Enhancements to QoS-OSPF in DiffServ and MPLS Networks”, IEE Proceedings – Communications, Vol. 151, No. 1, February 2004

[66] Ma, Qingming & Steenkiste, Peter “On Path Selection for Traffic with Bandwidth Guarantees”, International Conference on Network Protocols, Atlanta, USA, October 1997

[67] Floyd, Sally & Jacobson, Van “Link-Sharing and Resource Management Models for Packet Networks”, IEEE/ACM Transactions on Networking, Vol. 3, No. 4, August 1995

[68] Sridharan, Ashwin, Guérin, Roch & Diot, Christophe “Achieving Near-Optimal Traffic Engineering Solutions for Current OSPF/IS-IS Networks”, Proceedings INFOCOM 2003, San Francisco, April 2003

[69] Fortz, B. & Thorup, M. “Increasing Internet Capacity Using Local Search”, Technical Report IS-MG 2000/21, Université Libre de Bruxelles, 2000, http://www.ulb.ac.be/polytech/smg/publications/Preprints/FullText/Fortz00_21.pdf

[70] Fortz, Bernard & Thorup, Mikkel “Internet Traffic Engineering by Optimizing OSPF Weights”, Proceedings INFOCOM 2000, Vol. 2, Tel Aviv, Israel, March 2000

[71] Fortz, Bernard & Thorup, Mikkel “Robust Optimization of OSPF/IS-IS Weights”, Proceedings of INOC 2003, Paris, France, October 2003

[72] Apostolopoulos, G., Guérin, R & Kamat, S. “Implementation and Performance Measurements of QoS Routing Extensions to OSPF”, Proceedings INFOCOM ’99, Vol. 2, March 1999

[73] Xiao, Xipeng & Ni, Lionel “Reducing Routing Table Computation Cost in OSPF”, Proceedings INET’99, San Jose, USA, June 1999

[74] Choe, Myongsu, Wybenga, Jack, Kang, Byung Chang & Boukerche, Azzendine “A Routing Coordination Protocol in a Loosely-Coupled Massively Parallel Router”, Proceedings of IEEE HPSR 2002, Tokyo, May 2002

[75] Fortz, B. & Thorup. M. “Optimizing OSPF/IS-IS Weights in a Changing World”, IEEE Journal on Selected Areas in Communications, Vol. 20, No. 4, May 2002

[76] Basu, Anindya & Riecke, Jon G. “Stability Issues in OSPF Routing”, Proceedings SIGCOMM 2001, San Diego, USA, August 2001

[77] Devel, Manasai et al “Distributed Control Plane Architecture for Network Elements”, Intel Technology Journal, Volume 7, Issue 4, November 2003

[78] Katz, D. & Ward, D. “BFD for IPv4 and IPv6 (single hop)”, draft-ietf-bfd-v4v6-1hop-00.txt, July 2004

[79] Dubrovsky, Alex, Gerla, Mario, Lee, Scott S. & Cavendish, Dirceu "Internet QoS Routing with IP Telephony and TCP Traffic", ICC 2000, New Orlean, USA, June 2000

[80] Spitler, Stephen L. & Lee, Daniel C. "Integrating Effective-Bandwidth-Based QoS Routing and Best Effort Routing", Proceedings INFOCOM 2003, San Francisco, April 2003

[81] Coltun, R. "The OSPF Opaque LSA Option", RFC 2370, July 1998

[82] Katz, D., Kompella, K. & Yeung, D. "Traffic Engineering (TE) Extensions to OSPF Version 2", RFC 3630, September 2003

[83] Alnuweiri, Hussein M., Wong, Lai-Yat Kelvin & Al-Khasib, Tariq "Performance of New Link State Advertisement Mechanism in Routing Protocols with Traffic Engineering Extensions", IEEE Communications Magazine, Vol. 42, No. 5, May 2004

[84] Sibal, S. & Desimone, A. "Controlling Alternate Routing in General-Mesh Packet Flow Networks", Proceedings of ACM SIGCOMM, 1994, London

[85] Russell, Stuart J. & Norvig, Peter "Artificial Intelligence: a Modern Approach", 2nd Edition, Pearson Education Inc, 2003, ISBN 0-13-080302-2

[86] Hayzelden, Alex, L.G. & Bigham John (Eds) "Software Agents for Future Communication Systems", Spring-Verlag, 1999, ISBN 3-540-65578-6

[87] Franklin, Stan & Graesser, Art "Is it an Agent, or just a Program? A Taxonomy for Autonomous Agents", *Intelligent Agents III: Agent Theories, Architectures, and Languages* (eds. Muller, J.P., Wooldridge, M. & Jennings, N.R.) LNAI 1193, Springer-Verlag, Berlin, 1997, ISBN 3-540-62507-0

[88] Turing, A. "Computing Machinery and Intelligence", Mind, Vol. 59, No. 236, October 1950

[89] Muller, Jorg P. "The Design of Intelligent Agents – A Layered Approach", LNAI 1177, Springer 1996, ISBN 3-540-62003-6

[90] Weiss, Gerhard (ed) "Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence", MIT Press, 1999, ISBN 0-262-232-3-0

[91] Sutton, Richard S. & Barto, Andrew, G. "Reinforcement Learning: An Introduction", MIT Press, Cambridge, USA, 1998, ISBN 0-262-19398-1

[92] Keshava, S. & Sharma, R. "Achieving Quality of Service through Network Performance Management", Proceedings of NOSSDAV'98, Cambridge, UK, July 1998

[93] Jennings, B. et al "FIPA-compliant Agents for Real-time Control of Intelligent Network Traffic", Computer Networks 31, pp 2017-2036, 1999

[94] Luck, Michael, McBurney, Peter & Priest, Chris "Agent Technology: Enabling Next Generation Computing", AgentLink, 2003, ISBN 0854 327886

[95] Biezczad, A., White, T. & Pagurek, B. “Mobile Agents for Network Management”, IEEE Communications Surveys, Vol. 1, No. 1, September 1998

[96] Wooldrige, M. & Jennings, N. “Intelligent Agents: Theory and Practice”, Knowledge Engineering Review, Vol. 10, No. 2, January 1995

[97] Wooldridge, Michael “An Introduction to Multiagent Systems”, John Wiley & Sons Ltd, 2003, ISBN 0-471-49691-X

[98] Brooks, R.A. “A Robust Layered Control System for a Mobile Robot”, IEEE Journal of Robotics and Automation, Vol. 2, 1986

[99] Jennings, Nicholas R. & Bussmann, Stefan “Agent-Based Control Systems: why are they suited to engineering complex systems?”, IEEE Control Systems Magazine, Vol. 23, No. 3, June 2003

[100] Wooldridge, Michael & Jennings, Nicholas R. “Pitfalls of Agent-Oriented Development”, Proceedings 2nd International Conference on Autonomous Agents (Agents – 98), Minneapolis, USA, 1998

[101] Wooldridge, Michael J. & Jennings, Nicholas R. “Software Engineering with Agents: Pitfalls and Pratfalls”, IEEE Internet Computing, 3 (3), May-June 1999

[102] Perkins, C. (ed) “IP Mobility Support”, RFC 2002, October 1996

[103] Poslad, Stefan & Charlton, Patricia “Standardizing Agent Interoperability: The FIPA Approach”, in *Multi-Agent Systems and Applications*, LNAI 2086, April 2001, ISBN 3-540-42312-5

[104] Case, J., Mundy, R. Partain, D. & B. Stewart “Introduction and Applicability Statements for Internet Standard Management Framework”, RFC 3410, December 2002

[105] Pavlou, George, Flegkas, Paris, Gouveris, Stelios & Liotta, Antonio “On Management Technologies and the Potential of Web Services”, IEEE Communications Magazine, Vol. 42, No. 7, July 2004

[106] Comer, Douglas E. “Internetworking with TCP/IP Vol 1: Principles, Protocols and Architecture” 4th Edition, Prentice Hall, ISBN 0-13-018380-6, 2000

[107] Muller, Nathan J. “Improving Network Operations With Intelligent Agents”, International Journal of Network Management, Vol. 7, No. 3, May/June 1997

[108] Nichols, K., Jacobson, V. & Zhang, L. “A Two-bit Differentiated Services Architecture for the Internet”, RFC 2638, July 1999

[109] Shelén, Olov, Nilsson, Andreas, Norrgard, Joakim & Pink, Stephen “Performance of QoS Agents for Provisioning Network Resources”, Proceedings of IFIP 7th International Workshop on QoS (IWQoS’99), London, 1999

[110] Schelén, Olov “Quality of Service Agents in the Internet”, PhD Thesis, Luleå University of Technology, Sweden, August 1998

[111] Border, J., Kojo, M., Griner, J., Montenegro, G. & Shelby, Z. “Performance Enhancing Proxies Intende to Mitigate Link-Related Degredations”, RFC 3135, June 2001

[112] Galloway, Alexander R. “Protocol: How Control Exists After Decentralization”, MIT Press, 2004, ISBN 0-262-07247-5

[113] Durfee, E.H “Practically Coordinating” AI Magazine, Vol. 20, Issue 1, 1999

[114] Bourne, Rachel, Shoop, Karen & Jennings, Nicholas “Dynamic Evaluation of Coordination Mechanisms for Autonomous Agents” in Progress in Artificial Intelligence, LNAI 2258, Dec 2001, ISBN 3-540-43030-X

[115] Faratin, P., Sierra, C. & Jennings, N.R. “Using Similarity Criteria to Make Negotiation Trade-Offs”, Proceedings of 4th International Conference on Multiagent Systems, Boston, USA, July 2000

[116] Bodanese, E.L. & Cuthbert, L. “A Multi-Agent Channel Allocation Scheme for Cellular Mobile Networks”, Proceedings of 4th International Conference on Multiagent Systems, Boston, USA, July 2000

[117] Hayzelden, Alex & Bigham, J. “Heterogeneous Multi-Agent Architecture for ATM Virtual Path Network Resource Configuration”, in “Intelligent Agents for Telecommunications Applications (IATA ’98)”, LNAI 1437, Albayrak, S & Garijo, F.J (eds), Springer-Verlag, 1998, ISBN 3-540-64720-1

[118] Vayia, E., Soldatos, J., Bigham, J., Cuthbert L. & Luo, Z. “Intelligent Agents for ATM Network Control and Resource Management: Experiences and Results from an Implementation on a Network Testbed”, Journal of Network and Systems Management, Vol. 8, No. 3, September 2000

[119] Hayzelden, Alex, Bigham, John & Luo, Zhiyuan “Multi-Agent Interactions for a Network Management System (Tele-MACS Approach)” in Hayzelden, Alex, L.G & Bigham, John (eds)

[120] Ryan, Damian, Bigham, John, Cuthbert, Laurie & Tokarchuk, Laurissa “Intelligent Agents for Resource Management in Third Generation Networks”, Proceedings of Twenty-first SGES International Conference on Knowledge Based Systems and Applied Artificial Intelligence (ES2001), Cambridge, UK, Dec 2001

[121] Vilà, Pere, Marzo, José L. & Calle, Eusebi “Dynamic Bandwidth Management as Part of an Integrated Network Management System Based on Distributed Agents”, Proceedings of GLOBECOM 2002, Taipei, Taiwan, November 2002

[122] Monteiro, Paulo & Correia, Luís “Adaptive Telecommunication Network Traffic Control - A Multi-Agent System Approach”, 2nd Ibero-American Workshop on DAI and MultiAgent Systems, Toledo, Spain, October 1998

[123] Willmott, Steven & Faltings, Boi “The Benefits of Environment Adaptive Organisations for Agent Coordination and Network Routing Problems”, Proceedings IEEE ICMAS, Boston, USA, July 2000

[124] ATM Forum Technical Committee "Private Network-Network Interface Specification, Version 1.0 (PNNI 1.0)", March 1996

[125] Vidal, José M. & Durfee, Edmund H. "Learning Nested Agent Models in an Information Economy", Journal of Experimental and Theoretical Artificial Intelligence (special issue on learning in distributed artificial intelligence systems), Vol. 10, No.3, 1998

[126] Willmott, Steven et al, "Agentcities: A Worldwide Open Agent Network", URL: http://www-lsi.upc.es/~ia/aia/agentcities.pdf

[127] Wolpert, David, Kirshener, Sergey, Merz, Chris J & Tumer, Kagan "Adaptivity in Agent-Based Routing for Data Networks", Proceedings of 4th International Conference on Autonomous Agents, Barcelona, Spain, June 2000

[128] Korilis, Y.A, Lazar, A.A & Orda, A "Achieving network optima using Stackelberg routing strategies", IEEE/ACM Transactions on Networking, Vol. 5, No. 1, Feb 1997

[129] Peshkin, Leonid & Savona, Virginia "Reinforcement Learning for Adaptive Routing", Proceedings of 2002 International Joint Conference on Neural Networks, Honolulu, Hawaii, May 2002

[130] Nowé, A, Steenhaut, K., Fakir, M. & Verbeeck, K. "Q-learning for Adaptive, Load Based Routing", IEEE International Conference on Systems, Man and Cybernetics, San Diego, USA, October1998

[131] Tillotson, P.R.J, Wu, Q.H. & Hughes, P.M. "Multi-Agent Learning for Control of Internet Traffic Routing", IEE Seminar: Learning Systems for Control (Ref. No. 2000/069), Birmingham, UK, May 2000

[132] Papaioannou, T.G., Sartzetakis, S. & Stamoulis, G.D. "Efficient Agent-Based Selection of DiffServ SLAs over MPLS Networks within the ASP Service Model", Journal of Network and Systems Management, Special Issue on Management of Converged Networks, Vol. 10, Issue 1, March 2002

[133] Gibney, M.A., Jennings, N.R., Vriend, N.J. & Griffiths, J.M. "Market-based call routing in telecommunications networks using adaptive pricing and real bidding", In A.L.G.Hayzelden & R.A.Bourne "Agent Technology for Communication Infrastructures" p234-248, John Wiley & Sons, UK, 2001, ISBN0-471-49815-7

[134] Prouskas, K., Patel, A., Pitt, J. & Barria, J. "A Multi-agent System for Intelligent Network Load Control Using a Market-based Approach", 2000, IEEE Proceedings of 4th International Conference on MultiAgent Systems, 2000, 10-12 July 2000, Boston, USA

[135] Arvidsson A., Jennings B., Angelin L. & Svensson, M. "On the use of agent technology for IN load control", Proceedings of 16th International Teletraffic Congress (ITC-16), Edinburgh, UK, June 1999

[136] Bourne, Rachel A. & Zaidi, Rehan "A Quote-Driven Automated Market", Symposium on Information Agents for e-Commerce at the AISB'01 Convention, March 2001, York, UK,

[137] Gibney, M.A. & Jennings, N.R. "Dynamic Resource Allocation by Market-Based Routing in Telecommunications Networks", Proceedings IATA'98, LNAI 1437, ISBN 3-540-64720-1

[138] Prouskas, K., Patel, A., Pitt, J. & Barria, J. "A Multi-agent System for Intelligent Network Load Control Using a Market-based Approach", 2000, IEEE Proceedings of 4th International Conference on MultiAgent Systems, 2000, 10-12 July 2000, Boston, USA

[139] Wellman, M.P. "A Market-Oriented Programming Environment and its Application to Distributed Multicommodity Flow Problems", Journal of Artificial Intelligence Research, Vol. 1, No.1,1993

[140] URL: www.fipa.org

[141] Dorigo, Marco, Di Caro, Gianni & Gambardella, Luca M. "Ant Algorithms for Discrete Optimization", Artificial Life, Vol.5, no.2, 1999

[142] Schoonderwoerd, R., Holland, O.E. & Bruten, J.L. "Ant-like agents for load balancing in telcommunications networks" Proceedings of the 1st International Conference on Autonomous Agents, Marina Del Ray, USA, 1997

[143] Liang, Suihong, Zincir-Heywood, A.Nur & Heywood, Malcolm I. "Intelligent Packets for Dynamic Network Routing Using Distributed Genetic Algorithm", Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2002), New York, USA, 9-13 July 2002

[144] Legge, David & Baxendale, Peter "The Strategic Control of an Ant-Based Routing System using Neural Net Q-Learning Agents", AISB'04, Leeds, UK, March 2004

[145] Di Caro, G. & Dorigo, M. "AntNet: Distributed stigmergic control for communications networks", Journal of Artificial Intelligence Research (JAIR), 9, pp 317-365, December 1998

[146] Wu, Jin & Djemame, Karim "An Expert-System-Based Structure for Active Queue Management", Proceedings of 2nd International Conference on Machine Learning and Cybernetics, Xi'an, China, 2-5 November 2003

[147] Flegkas, Paris, Trimintzios, Panos & Pavlou, George "A Policy-Based Quality of Service Management System for IP DiffServ Networks", IEEE Network, Vol. 16, Issue 2, March/April 2002

[148] Rajan, Raju et al "A Policy Framework for Integrated and Differentiated Services in the Internet", IEEE Network, Vol. 13, Issue 5, September/October 1999

[149] Law, K.L.E. & Saxena, A. "Scalable Design Of A Policy-Based Management System And Its Performance", IEEE Communications Magazine, Vol. 41, No. 6, June 2003

[150] Dugeon, Olivier & Diaconescu, Ada "From SLA to SLS up to QoS Control: The CADENUS framework", Proceedings of WTC 2002, Paris, September 2002

[151] Trimintzios, Panos et al "Service-Driven Traffic Engineering for Intradomain Quality of Service Management", IEEE Network, Vol. 17, Issue 3, May/June 2003

[152] Trimintzios, P et al "Quality of Service Provisioning for Supporting Premium Services in IP Networks", Proceedings of IEEE GLOBECOM 2002, Taipei, Taiwan, November 2002

[153] EURESCOM "Inter-operator interfaces for ensuring end to end QoS", P1008, May 2001

[154] Boyle, J. et al "The COPS (Comment Open Policy Service) Protocol", RFC 2748, Jan 2000

[155] Chieng, David, Ho, Ivan Marshall, Alan & Parr, Gerard "An Architecture for Agent-Enhanced Network Service Provisioning through SLA Negotiation", Proceedings of Soft-Ware 2002: Computing in an Imperfect World, Belfast, April 2002, LNCS 2311, Springer Verlag, ISBN 3-540-43481-X

[156] Vilà, Pere "Dynamic Management and Restoration of Virtual Paths in Broadband Networks Based on Distributed Software Agents", PhD Thesis, University of Girona, 2004

[157] Liu, Nelson X. & Baras, John S. "Modelling Multi-Dimensional QoS: Some Fundamental Constraints", International Journal of Communication Systems, Vol. 17, Issue 3, April 2004

[158] Guerin, R., Orda, A. & Williams, D. "QoS Routing Mechanisms and OSPF Extensions", Proceedings IEEE Globecom 1997, Phoenix, USA, Nov 1997

[159] Orda, Ariel & Sprintson, Alexander "QoS Routing: The Precomputation Perspective", Proceedings IEEE INFOCOM 2000, Tel Aviv, Israel, March 2000

[160] Gopalan, Kartik, Chiueh, Tzi-cker & Lin, Yow-Jian "Load Balancing Routing with Bandwidth-Delay Guarantees", IEEE Communications Magazine, Vol. 42, No. 6, June 2004

[161] Jia, Yanxia, Nikolaidia, Ioani & Gburzynski, Pawel "On the Effectiveness of Alternative Paths in QoS Routing", International Journal of Communication Systems, Vol. 17, Issue 1, February 2004

[162] Awduche, D et al "Overview and Principles of Internet Traffic Engineering", RFC 3272, May 2002

[163] Kaya, Mehmet & Alhajj, Reda "Modular Fuzzy-Reinforcement Learning Approach with Internal Model Capabilities for Multiagent Systems", IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics, Vol. 34, Issue 2, April 2004

[164] Watkins, Christopher J.C.H & Dayan, Peter "(Technical Note) Q-Learning", Machine Learning, Vol. 8, No. 4, May 1992

[165] Singh, Satinder, Jaakkola, Tomit, Littman, Michael L. & Szepesvari, Csabla "Convergence Results for Single-Step On-Policy Reinforcement-Learning Algorithms", Machine Learning, Vol. 38, Issue 3, March 2000

[166] Tokarchuk, L., Bigham, J. & Cuthbert, L. "Fuzzy Sarsa: An Approach to Fuzzifying Sarsa Learning", Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation, Gold Coast, Australia, July 2004

[167] Jantzen, Jan "Fuzzy Control", Lecture notes in On-Line Process Control (5354), Publication No. 9109, Electric Power Engineering Department, Technical University of Denmark, October 1991 (revision 4, 1994)

[168] Verbruggen, H.B. & Babuska, R. (Eds) "Fuzzy Logic Control Advances in Applications", World Scientific, 1999, ISBN 981-02-3825-8

[169] Chrysostomou, C., Pitsillides, A., Hadjipollas, G., Sekercioglu, A. & Ploycarpou, M. "Fuzzy Logic Congestion Control in TCP/IP Best-Effort Networks", ATNAC 2003, Melbourne, Australia, December 2003

[170] Sivaramakrishna Mopati & Dilip Sarkar "Call Admission Control in Mobile Cellular Systems Using Fuzzy Associative Memory", Proceedings of IC3N'03, Dallas, USA, October 2003

[171] Aboelela, E. & Douligeris, C. "Routing in Multimetric Networks Using a Fuzzy Link Cost", Proceedings of 2nd IEEE Symposium on Computers and Communications, July 1997

[172] Chemouil, Prosper, Khalfet, Jelila & Lebourges, Marc "A Fuzzy Control Approach for Adaptive Traffic Routing", IEEE Communications Magazine, Vol. 3, No. 7, July 1995

[173] Qiu,Bin "Intelligent Algorithms for QoS Management in Modern Communication Networks", Proceedings of ICT2003, French Polynesia, February 2003

[174] He, Minghua & Jennings, Nicholas R. "Designing a Successful Trading Agent: A Fuzzy Set Approach", IEEE Transactions on Fuzzy Systems, Vol. 12, No. 3, June 2004

[175] Negnevitsky, Michael "Artificial Intelligence: a guide to intelligent systems", 2nd Edition, Addison Wesley, 2005, ISBN 0-321-20466-2

[176] Mamdani, E.H. & Assilian, S. "An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller", International Journal of Man-Machine Studies, Vol. 7, No. 1, January 1975

[177] Bonarini, A., Bonacina, C., & Matteucci,M. (2000) "Fuzzy And Crisp Representation of Real-Valued Input for Learning Classifier Systems" in Lanzi, P.L., Stolzmann, W. & Wilson, S. W. (eds) "Learning Classifier

System: from foundations to applications", LNAI 1813, Springer-Verlag, 2000, ISBN 3-540-67729-1

[178] Anderson, Charles W. "Learning to Control an Inverted Pendulum Using Neural Networks", IEEE Control Systems Magazine, Vol. 9, No. 3, April 1989

[179] Carlström, Jakob & Nordström, Ernst "Reinforcement Learning for Control of Self-Similar Call Traffic in Broadband Networks", Proceedings of 16th International Teletraffic Congress (ITC'16), Edinburgh, UK, June 1999

[180] Bonarini, Andrea "Reinforcement Distribution for Fuzzy Classifiers: a Methodology to Extend Crisp Algorithms", Proceedings IEEE International Conference on Evolutionary Computation, Anchorage, USA, May 1998

[181] Ariza, A., Casilari, E. & Sandoval, F. "Strategies for Updating Link States in QoS Routers", Electronics Letters, Vol. 36, No. 20, 28 September 2000

[182] Ariza, A., Casilari, E. & Sandoval, F. "QoS Routing With Adaptive Updating of Link States", Electronics Letters, Vol. 37, No. 9, 26 April 2001

[183] OPNET Modeler documentation, OPNET Technologies, Inc., Bethesda, USA

[184] Qui, Lili, Yang, Yang Richard, Zhang, Yin & Shenker, Scott "On Selfish Routing in Internet-Like Environments", Proceedings ACM Sigcomm '03, Karlsruhe, Germany, August 2003

[185] URL: http://www.juniper.net/techpubs/hardware/m160/m160-hwguide/m160-hwguide-TOC.html

[186] Cisco "Understanding the Transmit Queue Limit With IP to ATM CoS" Document ID: 6190, updated May 2004

[187] URL: http://www.cisco.com/en/US/tech/tk827/tk831/technologies_tech_note09186a00800946f7.shtml

[188] Floyd, Sally & Paxson, Vern "Difficulties in Simulating the Internet", IEEE/ACM Transactions on Networking, Vol. 9, No. 4, August 2001

[189] Leland, Will E., Taqqu, Murad S., Willinger, Walter & Wilson, Daniel V. "On the Self-Similar Nature of Ethernet Traffic (Extended Version)", IEEE/ACM Transactions on Networking, Vol. 2, Issue 1, February 1994

[190] Karagiannis, T., Molle, M. & Faloutsos, M. "Long-Range Dependence: Ten Years of Internet Modelling", IEEE Internet Computing, Vol. 8, Issue 5, September/October 2004

[191] Li, Guang-Liang & Li, Viktor O.K. "Networks of Queues: Myths and Reality" Proceedings of IEEE 18th Workshop on Computer Communications, Dana Point, USA, October 2003

[192] Xu, Ying & Guerin, Roch "Individual QoS versus Aggregate QoS: A Loss Performance Study", Proceedings IEEE INFOCOM 2002, New York, USA, June 2002

[193] Geogoulas, Stylianos, Triminitzios, Panos & Pavlou, George "Joint Measurement- and Traffic Descriptor-based Admission Control at Real-Time Traffic Aggregation Points" Proceedings ICC 2004, Paris, France, June 2004

[194] Carpenter, Brian & Nichols, Kathleen "Differentiated Services in the Internet", IBM Research Report, RZ3395, November 2002

[195] URL: http://www.math.sci.hiroshima-u.ac.jp/%7Em-mat/MT/emt.html

[196] Pawlikowski, Krzysztof, Jeong, Hae-Duck Joshua & Lee, Jong-Suk Ruth "On Credibility of Simulation Studies of Telecommunication Networks", IEEE Communications Magazine, Vol. 40, No. 1, January 2002

[197] Claffy, K. & Miller, Greg "The Nature of the Beast: Recent Traffic Measurement from an Internet Backbone", Proceedings inet98, Geneva, Switzerland, July 1998

[198] Heidemann, John, Mills, Kevin & Kumar, Sri "Expanding Confidence in Network Simulations", IEEE Network, Vol. 15, Issue 5, September/October 2001

[199] Brooks, Frederick P. Jnr "The Mythical Man-Month", Addison-Wesley, 1995, ISBN 0-201-83595-9

[200] Luck, Michael, McBurney, Peter, Shehory, Onn & Willmott, Steve "Agent Technology Roadmap draft: agent based computing", DRAFT to be published by the University of Southampton, UK, 2005

[201] Kolodner, Janet L. "Case-Based Reasoning", Morgan Kauffmann, 1993, ISBN 1-558-60237-2

[202] Jennings, N.R. "Agent-Based Computing: Promises and Perils", Proceedings 16th IJCAI, Stockholm, Sweden, August 1999