University of Sussex

Trusted Content-Based Publish/Subscribe Trees

Stephen Murugapa Naicken

Submitted for the Degree of Doctor of Philosophy

University of Sussex

October 2011

Declaration

I hereby declare that this thesis has not been and will not be submitted in whole or in part to another University for the award of any other degree.

Signature:

Stephen Murugapa Naicken

Abstract

Publish/Subscribe systems hold strong assumptions of the expected behaviour of clients and routers, as it is assumed they all abide by the matching and routing protocols. Assumptions of implicit trust between the components of the publish/subscribe infrastructure are acceptable where the underlying event distribution service is under the control of a single or multiple co-operating administrative entities and contracts between clients and these authorities exist, however there are application contexts where these presumptions do not hold. In such environments, such as ad hoc networks, there is the possibility of selfish and malicious behaviour that can lead to disruption of the routing and matching algorithms.

The most commonly researched approach to security in publish/subscribe systems is role-based access control (RBAC). RBAC is suitable for ensuring confidentiality, but due to the assumption of strong identities associated with well defined roles and the absence of monitoring systems to allow for adaptable policies in response to the changing behaviour of clients, it is not appropriate for environments where: identities can not be assigned to roles in the absence of a trusted administrative entity; long-lived identities of entities do not exist; and where the threat model consists of highly adaptable malicious and selfish entities.

Motivated by recent work in the application of trust and reputation to Peer-to-Peer networks, where past behaviour is used to generate trust opinions that inform future transactions, we propose an approach where the publish/subscribe infrastructure is constructed and re-configured with respect to the trust preferences of clients and routers. In this thesis, we show how Publish/Subscribe trees (PSTs) can be constructed with respect to the trust preferences of publishers and subscribers, and the overhead costs of event dissemination. Using social welfare theory, it is shown that individual trust preferences over clients and routers, which are informed by a variety of trust sources, can be aggregated to give a social preference over the set of feasible PSTs. By combining this and the existing work on PST overheads, the Maximum Trust PST with Overhead Budget problem is defined and is shown to be in NP-complete. An exhaustive search algorithm is proposed that is shown to be suitable only for very small problem sizes. To improve scalability, a faster tabu search algorithm is presented, which is shown to scale to larger problem instances and gives good approximations of the optimal solutions.

The research contributions of this work are: the use of social welfare theory to provide a mechanism to establish the trustworthiness of PSTs; the finding that individual trust is not interpersonal comparable as is considered to be the case in much of the trust literature; the Maximum Trust PST with Overhead Budget problem; and algorithms to solve this problem.

Acknowledgements

Foremost, I would like to express my sincere gratitude to my supervisor, Dr. Ian Wakeman for his time, constructive criticisms, guidance and encouragement. I must also extend my thanks to the rest of my thesis committee: Dr. Des Watson and Dr. Dan Chalmers for their encouragement, insightful thoughts, and constructive criticisms of my work.

I consider myself fortunate to have had the opportunity to work alongside past and present members of the Foundations of Software Systems group. During my time here, I learnt a great deal from them that will stand me in good stead for many years to come. My discussions about the intricacies of trust with Dr. Anirban Basu and Dr . Jian Li helped me to develop my understanding of trust, while the strenuous badminton sessions with Dr .Lachman Dhomeja and Dr. Yasir Malkani ensured that I maintained a somewhat reasonable work-life balance. I thank them for their time, as I do of the others: Dr. Jon Robinson, Dr. David Ellis, James Stanier, Simon Fleming, Renan Krishna, Barney Livingston and Dr. Roya Feizy.

I am deeply indebted to my mother and father, Kistnamah Naicken and Vadivel Naicken. Without their emotional and financial support, I would not have been able to undertake and complete this work. I dedicate this thesis to my mother, to my late Aunt Biga and to my late maternal grandparents, Soopaya Rungasamy and Veeramah Nallee.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Research Motivation

Given the scalability of the Internet and online services, it is common for users to interact with others who are unknown to them. Such interactions are at greater risk of failure due to the increased likelihood of malicious and selfish behaviour. Consider an online auction system where a buyer must decide if he wishes to enter into an agreement to purchase an item from a seller, but has not conducted any transactions with him in the past. The buyer is clearly unable to reach an informed decision on the reputation of the seller and consequently it may choose not to trust it. In the absence of information on the past behaviour of the seller, it may be able to make use of indicators, such as attributes of the item's textual description that is provided by the seller to determine if it should place its trust in the seller. For some individuals this may be sufficient for a transaction to proceed, while for others it may not. In the absence of trust in an online auction system, the number of high-value transactions that take place will be greatly reduced, as users' perceived risk will be too great.

The use of reputation systems is a widely adopted approach to addressing the issue of the absence of trust between users of Internet services (Resnick et al., 2000). These systems utilise the past behaviour of entities to determine their reputation and consequently their trustworthiness, which can then be used by users to determine the risk entailed in future interactions with others. A reputation system consists of three components (Marti and Garcia-Molina, 2006): information gathering; scoring and ranking; and response. Following on from the online auction example given above, a reputation system would gather information by requesting that users provide feedback on transactions, with positive and negative feedback aggregated to give a score for each user that would then be used by

others to decide upon the risk entailed in future transactions. Aside from the disincentive of being ostracised due to poor reputation, incentives could be introduced by the service provider to encourage good behaviour. An area in which there has been significant research in the use of reputation management is Peer-to-Peer (P2P) networks. A number of reputation systems have been proposed to address the issues of malicious and selfish behaviour (Kamvar et al., 2003) (Xiong and Liu, 2004), such as the distribution of corrupted files and freeloading, respectively. Another example of the use of reputation to secure a system is CONFIDANT (Buchegger and Le Boudec, 2004), which secures mobile ad hoc network (MANET) routing by monitoring the routing behaviour of nodes and using this information as input to a Bayesian trust model. Paths consisting of untrusted nodes are ignored at the path selection stage of routing and requests for routes from untrusted nodes are dropped.

Trust is not restricted to individuals and agents, non-atomic structures can also be trusted. To say that one trusts the government to govern the state implies that the one sufficiently trusts the governance provided by the components of government. For the British government, these are: the Sovereign; the Crown; the Cabinet; the Privy Council; and the Civil Service. The same is also true when one says that we trust society[1], however as it is not well defined, it is much harder to determine its components. To date, computational trust models have not addressed composite trust, that is trust in an entity as a function of the trust of its components, and yet it is central to following question, do users trust the network? A network consists of inter-connected nodes of various types, all responsible for the execution of various algorithms to ensure reliable, efficient and when required, secure communications. Interactions take place within the network and not with it, so it is reasonable to assume that if a node trusts a network of which it is a member, then it trusts the other nodes to implement their roles and responsibilities correctly. Each node will have a preference over the set of feasible network topologies, which is informed by its trust of the nodes and paths for each topology. The preferences of nodes may be in conflict, but despite this, they must all be aggregated to yield a single, socially acceptable preference over the network topologies. This must be done in a manner that is as equitable as possible unless some policy dictates otherwise.

The motivation for trusted network structures is that the risk posed by malicious and selfish nodes is reduced, as their presence is minimised or eradicated from the network. The construction of trusted networks and more specifically trusted publish/subscribe trees

[1]This is of course under the assumption that it exists, which would contradict a former British Prime Minister.

is the crux of this thesis. The motivation for the choice of publish/subscribe is given in the remainder of this section.

Publish/Subscribe is a communication paradigm where events generated by producers are delivered to interested consumers. A publish/subscribe system consists of publishers, subscribers and an event notification service of brokers whose responsibility is to facilitate event dissemination from publishers to interested subscribers. A subscriber issues a subscription that expresses its interests to the event notification service. Publishers submit events to the event notification service, which are delivered to subscribers who have issued matching subscriptions. The event notification service is responsible for the following: the matching of subscriptions and notifications; and the routing of notifications from publishers to those subscribers where notifications and subscriptions match.

The primary advantage of publish-subscribe over other forms of network communication models such as the client-server model is decoupling (Eugster et al., 2003), which is defined along the following three dimensions:

- **Time** - Publishers and subscribers do not need to be participating in the interaction at the same time. Events may be published when some subscribers are disconnected and events may be received by subscribers when the publisher is disconnected.

- **Space** - Publishers and Subscribers need not be aware of one another. Publishers push events to subscribers using the event notification service without the need for the identifiers of the subscribers. Subscribers do not require the identities of publishers in order to receive events that they issue. Space decoupling gives a degree of sender and receiver anonymity, as only the brokers to which the subscribers and publishers are connected will be able to associate their identity with their subscriptions and notifications, respectively.

- **Synchronisation** - Publish/Subscribe communication is asynchronous. Publishers do not block when pushing notifications as no acknowledgement is required from the subscriber. Subscribers can be notified using callback of the arrival of an event of interest.

It is this decoupling that makes publish/subscribe advantageous for a number of applications, such as real-time sports scores distribution[2], stock quotations services and online auctions. The drawback to this decoupling, particularly the space decoupling, is that there

[2]http://www.research.ibm.com/distributedmessaging/gryphon.html

is a restriction on accountability and consequently there is the potential for a number of security issues and also difficulties in addressing them.

Although there has been little research in identifying and understanding the security vulnerabilities of publish/subscribe systems, a taxonomy of denial-of-service (DoS) attacks (Wun et al., 2007) and confidentiality issues (Wang et al., 2002) have been presented. Typically, designers of event-based notification services have assumed that implicit trust exists between clients, clients and the event notification service, and the components of the event notification service. Where this assumption is relaxed, the research emphasis has been on the use of Role-based Access Control (RBAC) (Belokosztolszki et al., 2003) (Pesonen et al., 2006) (Pesonen et al., 2007), and matching and routing algorithms using encrypted notifications and subscriptions (Li et al., 2004) (Srivatsa and Liu, 2005) (Raiciu and Rosenblum, 2006) to address the trust issues. One significant disadvantage of RBAC and computing on encrypted data techniques is that both are unable to adapt autonomously to changes in the behaviour of participants. In these systems, changes in the trust relationships require the re-keying of encryption functions and for RBAC, the re-specification of policies.

In this thesis, it is proposed that the publish/subscribe tree (PST) - the tree topology used for event dissemination - is constructed and reconfigured when necessary with respect to both the trust preferences of the publishers and the subscribers, and the communication overhead costs. A trusted PST minimises the risk posed by selfish and malicious nodes by attempting to eliminate them from the network, whilst ensuring that efficient communication may take place. Should any changes in behaviour by nodes of a PST be observed by a monitoring service, the PST can be reconfigured to reflect the updated reputation of the nodes. As noted above, this adaptation to changing behaviour in existing security systems for publish/subscribe is complex.

The use of trusted PSTs for event dissemination is suitable for a number of application areas. Where there is an absence of long-lived identities or administrative entities responsible for the publish/subscribe infrastructure, the use of RBAC is not feasible and there is an increased risk from malicious and selfish behaviour. Examples of such environments are publish/subscribe applications in MANETs and publish/subscribe based Internet routing.

In addition to publish/subscribe in MANETs, other application areas that are suitable for RBAC and confidential publish/subscribe may also be suitable for the use of trusted PSTs. For real time publish/subscribe systems, such as air traffic control data distribution services, trust is a function of quality of service (QoS). In these systems, the PST can be

constructed and reconfigured to maximise trust and consequently performance. With respect to air traffic control, an area control centre (ACC) can publish radar data using a trusted PST to other ACCs, where the publishing ACC is the root and subscribing ACCs are the terminal nodes. The internal nodes of the PST are those of the event distribution service (e.g. EuroControl's Radar Network[3]) that may consist of brokers under the responsibility of one or many organisations.

The emphasis of the research presented in this thesis is the trust evaluation function for PSTs and algorithms to find the PST with the greatest trust subject to some overhead communication budget. Monitoring of PSTs, a component of any reputation system, is discussed at the end of this thesis as an avenue for future research.

1.2 Research Contributions

Although the use of trust and reputation in P2P applications and ad hoc routing is one of the inspirations for this research, the application of trust is rather different in those contexts. In P2P applications, trust is typically used to determine if one-to-one transactions should take place, for example file transfers. Here, rather than use trust to determine if interactions should occur, the objective is to create a trusted network structure, that is a PST that is trusted by the publisher and its subscribers.

Trust is a mental state and is likely to differ across individuals, as each will use different trust sources combined with their own individual personality traits (captured by an individual's trust evaluation function) to come to a trust evaluation of some entity. More often than not, the trust preferences of publishers and subscribers will differ, so how can individuals' preferences over some set of feasible PSTs be aggregated to single ordering that appeases the desires of all the relevant entities? Fortunately, aggregation of individual preferences is the core research issue of social welfare and choice theory (Arrow et al., 2002) and concepts from this field are used extensively to address this issue. In the absence of policies that prioritise the trust preferences of some subset of nodes over others, it is evident to ask how the individuals' trust preferences can be fairly aggregated to a single social trust preference. A utilitarian approach is rejected, as it is shown by example that this can punish those that have little trust in the socially chosen PST. Instead, rather than appease the majority, the motivation for the proposed aggregation function is Rawls' principles of justice (Rawls, 1971), in particular the difference principle that states that social and economic inequalities satisfy the condition that they are to be to the greatest

[3] http://www.eurocontrol.int/surveillance/public/standard_page/sur_SDDS.html

benefit of the least advantaged members of society. Of course, there are philosophical argument for and against each approach, but we argue that a PST that results in great risk to one node and little risk to others is not preferable to one where the risk to the least well off is reduced at the expense of others. The leximin social welfare functional models Rawls' difference principle and an analytical form of the functional (Yager, 1997) is used to aggregate individuals' trust preferences over PSTs.

The use of the leximin social welfare functional to aggregate individuals' trust functions raises an important issue that has been neglected in many computational trust models. If trust is considered to be cardinal, as is required for the trust maximisation problem addressed in this thesis, then it must be inter-personal comparable. It is argued that trust is not inter-personal comparable, unless the unrealistic assumption is made that individuals determine the trustworthiness of others in the same manner. It is an assumption that is reluctantly made in this work, but it is hoped that this neglected research issue will be considered by designers of future computational trust models.

Aside from the issue of aggregating individuals' trust preferences over a set of feasible PSTs, there remains the problem of how publishers and subscribers can determine the trustworthiness of a PST. It seems logical to assume that one's trust in the PST is given by the relationships that exist between it and other nodes in the tree, that is an individual trust evaluation of a PST is a function of the trust of some subset of paths in the PST of which it is an endpoint. The communication paths for a node in a PST are dependent upon its role and its depth. For the publisher, the paths that must be considered are the paths to each subscriber. For subscribers that are terminal nodes, the only path of interest is that to the publisher, but for an internal node subscriber, not only the path to the publisher must be evaluated but also the paths to each descendant node that is a subscriber. Individual PST trust evaluation functions for publishers, internal subscribers and terminal subscribers are defined. There is, to the best knowledge of the author, no existing work that addresses the issue determining the trustworthiness of a network structure.

Having proposed a mechanism to determine the trustworthiness of PSTs, the question of finding the PST that maximises trust within some overhead budget can be addressed. The problem is defined and shown to be in NP-complete, so an exhaustive search approach can not be expected to scale to large problem instances. Instead, a tabu search algorithm is proposed that has greater scalability than the exhaustive search. The quality of the solutions found by the tabu search are evaluated and shown to be good approximations of

the optimal solutions. Running times are shown to be superior to those of the exhaustive search except for very small problem sizes.

1.3 Thesis Outline

Chapter 2 consists of two parts: an introduction to PSTs; and a discussion of the security issues in publish/subscribe systems and various approaches to addressing them. The PST preliminaries include the definition of a PST and the overhead metric that are used in the remainder of the thesis. Existing approaches to the minimum overhead PST problem are also reviewed. The second part of the chapter covers publish/subscribe security, both the security issues and proposed solutions to address them. The surveyed approaches can be classified as being of either access-control or computing on encrypted data based. A number of limitations to these approaches are highlighted and justify the need for an adaptive trust based mechanism to address security challenges.

Chapter 3 begins with a predominantly mathematical introduction to social welfare theory. Only the necessary prerequisites are presented with particular emphasis on the leximin social welfare functional and inter-personal comparability, both central to the PST trust function that is defined later in the chapter. Following on from the social welfare theory preliminaries, a trust evaluation function for PSTs is defined, which gives a socially acceptable ordering over a set of feasible PSTs that maximises the trust in the PSTs held by the least trusting. As the function is a leximin social welfare functional it requires inter-personal comparability of utilities, in this case trust. We argue that the inter-personal non-comparability of trust has been disregarded in existing computational trust models, and we present a number of assumptions to address this issue for the proposed approach.

Having devised a mechanism to evaluate the trustworthiness of a PST, the maximum trust PST with overhead budget problem is defined and approaches to solve it are presented in chapter 4. Although the problem is shown to be in NP-complete, an exhaustive search algorithm is presented, followed by an algorithm that uses the tabu search metaheuristic.

An evaluation of the proposed algorithms is given in chapter 5. Three problem sets are defined where the problems increase in complexity with the number of vertices and edges in the connectivity graph. The mathematical methods used to generate these sets are also described. The exhaustive search algorithm is used to determine the optimal solutions for problems of low complexity only, as it does not scale to larger problems, which is to be expected given that the problem is in NP-complete. Solutions found by the tabu search algorithm under various configurations are compared to the optimal solution where

available. The results of the evaluation are as expected: the tabu search algorithm scales to larger problem instances than the exhaustive search; and the tabu search algorithm finds the optimal solution for the majority of problems in execution times that are comparable to those of the exhaustive search algorithm for problems of considerably less complexity.

In chapter 6, the thesis concludes with a summary of the research contributions. A discussion of avenues for future research is also given.

Chapter 2

Background

2.1 Introduction

The concept of publish/subscribe was originally proposed for the implementation of group inter-process communication in the System V kernel (Cheriton and Zwaenepoel, 1985), while the news service of ISIS2 by Frank Schmuck (Birman and Joseph, 1987) is the first documented implementation of a topic-based publish/subscribe system (definition 1). Since these initial developments, a great deal of research on publish/subscribe systems has been conducted. The most notable research developments are the evolution of publish/subscribe data models from topic-based to the more expressive content-based model (definition 2) (Rosenblum and Wolf, 1997) and the decentralisation of event distribution architectures (Cugola et al., 2002) (Carzaniga et al., 2001). More recently, the emphasis of publish/subscribe research has been on its application in mobile environments (Cugola and Jacobsen, 2002) (Meier and Cahill, 2002) (Caporuscio et al., 2003) (Fiege et al., 2003), the use of P2P architectures for the implementation of the event distribution architecture (Pietzuch and Bacon, 2002) (Gupta et al., 2004) (Triantafillou and Aekaterinidis, 2004) (Baldoni et al., 2005) (Zhu and Hu, 2005) (Choi and Park, 2006) and optimisation of the content-based routing and matching algorithms (Fabret et al., 2001) (Carzaniga and Wolf, 2003) (Muhl et al., 2003). An introduction to publish/subscribe is given by Eugster et al. (Eugster et al., 2003).

Definition 1 (Topic-Based Publish/Subscribe)**.** In topic-based publish/subscribe, each publisher publishes each of its events to a topic or subject, while a subscriber subscribes to a topic to receive all events published to it. This scheme typically utilises either flat-addressing or hierarchical-addressing of topics. Flat-addressing subdivides the event space into disconnected subspaces. Hierarchical-addressing allows for a structured topic-space,

where a subscription to a topic in the hierarchy also implies a subscription to all sub-topics.

Definition 2 (Content-Based Publish/Subscribe)**.** Content-based publish/subscribe addresses the lack of expressiveness of the topic-based model by defining a subscription as a function over the event's content. The cost of this expressiveness is that as the subscription complexity increases, the message state and processing complexity at broker nodes which must perform content-based matching and routing.

The publish/subscribe literature presented and discussed in this chapter covers the two areas that are most relevant to the research presented in this thesis: publish/subscribe trees as described by Huang and Garcia-Molina (Huang and Garcia-Molina, 2003), that is an event-distribution topology where there is no concept of a broker network with clients external to it, but instead a tree spanning the publisher and its subscribers where brokers and subscribers may be internal nodes that are responsible for routing notifications; publish/subscribe security with particular emphasis on the two types of solutions to various security issues described in (Wang et al., 2002), computing on encrypted data and access control.

2.2 Publish/Subscribe Trees

Publish/Subscribe Tree (PST) construction with respect to overhead costs was considered by Huang and Garcia-Molina in the context of publish/subscribe in wireless ad hoc networks (Huang and Garcia-Molina, 2003). They proposed an overhead metric for PSTs and the SHOPPARENT algorithm to construct a PST. Their work makes the simplifying assumption that there need only be one tree spanning the connectivity graph where any node can publish an event by routing it to the root of the tree, which then distributes it to the subscribers.

The SHOPPARENT algorithm adopts a greedy approach to the construction of the least overhead PST by having each node periodically search for a parent that reduces the overhead at that node. The algorithm allows for the mobility of nodes by using beacon messages to inform nodes of potentially better parents that have moved within range and to notify descendant nodes in the PST of loss of connectivity due to node failure or movement of the parent to some other part of the connectivity graph. Although the SHOPPARENT algorithm solves a different problem to the one addressed in this work, the overhead metric is used as defined by the authors and is discussed futher in section 2.2.2.

2.2.1 Publish/Subscribe Tree Roles

Each node in a PST may assume only one of three roles: publisher; subscriber; and router. As there may be many advertisements from many publishers at a given time and each advertisement is associated with a unique PST, it is important to note that a node may assume different roles in different PSTs. For example, in one tree a node may be a router that facilitates the operation of the event routing and matchmaking, while in another PST it is the root publishing events. The role of a node is always specified within the context of a given PST, $T_{A_p} = (V_{A_p}, E_{A_p})$. These roles can be defined as:

Definition 3 (Publisher Role)**.** The publisher, p, issues an advertisement A_p. The PST, T_{A_p}, for A_p, is rooted at p. p propagates events matching A_p on this tree to the routers and subscribers in T_{A_p}.

Definition 4 (Subscriber Role)**.** The set of all subscribers is S such that $S \subseteq V$. $S = V$ is possible due to the multiple roles a node may assume in different PSTs. A node is a subscriber if it has one or more subscription functions, $S = \{v \mid SF_v \neq \emptyset \wedge v \in V\}$ where SF_v is the set of subscriptions functions held by v. Given a PST T_{A_p}, the set of subscriber nodes in T_{A_p} is $S_{A_p} = \{s \mid sf_s(A_p) = true \ \wedge s \in S\}$ where sf_s is the subscription function of s. If a subscriber s is an internal node of the PST, it not only receives events to which it is subscribed to, but also routes events to descendent nodes.

Definition 5 (Router Role)**.** All nodes in a PST that are neither the publisher nor subscribers are routers. Their role is to forward events to downstream subscribers and ensure connectivity to subscribers. The set of possible router nodes for a PST T_{A_p} is given by $R_{A_{p_c}} = V \setminus (S_{A_p} \cup \{p\})$ where V is the set of vertices in the connectivity graph. The set of router nodes in the PST T_{A_p} is given by $R_{A_p} = V_{A_p} \setminus (S_{A_p} \cup \{p\})$.

2.2.2 Publish/Subscribe Tree Overheads

Before presenting the overhead metric for PSTs, the following three types of subscription are defined: inherent subscription; effective subscription; and proxied subscription (Huang and Garcia-Molina, 2003).

Definition 6 (Inherent Subscription)**.** The inherent subscription s_i of a subscriber i is given by its subscription function sf_i.

Definition 7 (Effective Subscription)**.** The effective subscription S_i of a subscriber i is given by the disjunction of its inherent subscription s_i and its proxied subscription $s_i^{'}$, $S_i = s_i \vee s_i^{'}$.

Definition 8 (Proxied Subscription)**.** The proxied subscription $s_i^{'}$ of a subscriber i is given by $s_i^{'} = \bigcup_{j=1,\ldots,n} S_j$ for each child $1, \ldots, n$ of i.

Huang and Garcia-Molina (Huang and Garcia-Molina, 2003) define the cost of a PST T, $C_T(E)$, as:

$$C_T(E) = \sum_i C_{T_i}(E) \tag{2.1}$$

where E is the set of events to be published and $C_{T_i}(E)$ is the cost of receiving, processing and forwarding the events in E at node i of T. The cost C_{T_i} at a node i, is given by:

$$C_{T_i}(E) = (r + pr) \cdot \Phi E(s_i \wedge \neg s_i^{'}) + (r + f) \cdot \Phi E(\neg s_i \wedge s_i^{'}) + (r + pr + f) \cdot \Phi E(s_i \wedge s_i^{'}) \tag{2.2}$$

where r is the cost to receive an event, pr is the cost to process an event, f is the cost to forward the event, s_i is the subscription function at node i, $s_i^{'}$ is the aggregation of the subscription functions of all child nodes (and by recursion, all descendant nodes) of i, and $\Phi E(\alpha)$ gives the number of events from the set E that match the subscription function α.

Huang and Molina go on to show that the equations 2.1 and 2.2 can be simplified to define the overhead rather than cost. In equation 2.2, the first component of the sum remains constant for all trees, while for the third component, we can remove the processing and forwarding constants as these are not overheads since the node has an interest in these events. For the second component, node i has no interest in these events, so costs associated with this component are entirely overheads. This gives the following functions to replace equations 2.1 and 2.2:

$$O_T(E) = \sum_i O_{T_i}(E) \tag{2.3}$$

$$O_{T_i}(E) = (r + f) \cdot \Phi E(\neg s_i \wedge s_i^{'}) + f \cdot \Phi E(s_i \wedge s_i^{'}) \tag{2.4}$$

While Huang and Molina state that it is possible to determine for some applications the event distribution, E, it can be shown that a priori knowledge of the event distribution is not required for the implementation of tree construction algorithms with respect to equations 2.3 and 2.4. By inspection of equation 2.4, nodes with subscriptions not covered by their ancestors' subscriptions incur greater overheads on their ancestors than those

where there is covering. To lower the total overhead at each node, the tree construction algorithm should attempt to maximise subscription covering in the PST, tending $\Phi E(s_i \wedge s_i^{'})$ in 2.4 to larger values and $\Phi E(\neg s_i \wedge s_i^{'})$ to lower, as greater overheads $(r + f)$ are incurred with the latter.

Definition 9 (Publish/Subscribe Tree Overhead)**.** Let E be a set of events, r be some cost associated with receiving an event, f be a cost associated with forwarding an event, s_i be the inherent subscription and $s_i^{'}$ be the proxied subscription. For a PST T_{A_p}, its overhead is defined as $O_{T_{A_p}}(E) = \sum_{i \in V_{A_p}} O_{T_{A_{p_i}}}(E)$ where $O_{T_{A_{p_i}}}(E) = (r + f) \cdot \Phi E(\neg s_i \wedge s_i^{'}) + f \cdot \Phi E(s_i \wedge s_i^{'})$.

2.2.3 Publish Subscribe Tree Definition

In this work, a different definition of PST is adopted. For each advertisement, there is a PST rooted at the publisher of the advertisement. The tree spans all subscribers whose subscription function matches the advertisement associated with the PST and subset of router nodes who are not subscribers to the advertisement, but are included in the PST to facilitate connectivity and reduce overheads.

PSTs are not explicitly defined by Cao and Shen, so the following definition has been devised. Note that only subscribers may be terminal nodes at any given time in a PST, as it is illogical to forward events to a router who has no descendants that are subscribers, as this only increases the overhead cost of the PST.

Definition 10 (Publish Subscribe Tree (PST))**.** Given an undirected connected connectivity graph $G = (V, E)$, a publisher p such that $p \in V$, an advertisement A_p held by publisher p, a set of subscribers $S_{A_p} = \{s \mid sf_s(A_p) = true \ \wedge s \in V \setminus \{p\}\}$ where sf_s is the subscription function of s, and a set of routers $R_{A_p} = V \setminus (S_{A_p} \cap \{p\})$ is the set of candidate router nodes. A PST T_{A_p} for the advertisement A_p is a tree routed at p that spans all subscribers in S_{A_p} and a subset of R_{a_p} nodes where all $r \in R_{a_p}$ can not be a terminal node of the PST and for all $s \in S_{a_p}$, s may be either a branch node or a terminal node of the PST.

2.2.4 The Minimum Overhead Publish/Subscribe Problem

Following the work of Huang and Garcia-Molina (Huang and Garcia-Molina, 2003), Cao and Shen (Cao and Shen, 2009) address the same problem of finding the minimum overhead PST tree, however their approach differs, as they assume that the PST is not a spanning tree on the connectivity graph, but a Steiner tree that is rooted at the publisher and spans

its subscribers. With this definition of a PST, the minimum overhead PST problem (see problem 1) is shown to be NP-complete (see theorem 1). The authors then go on to present the DSAPST algorithm where each node selects a parent as a function of the hop count to the publisher and a measurement of the subscription covering provided by the parent's subscription i.e. the overlap of the parent's subscription with the node's subscription. The subscription-aware DSAPST algorithm is compared to a multicast subscription-unaware MAODV[1] approach to event distribution, with the former outperforming the latter with respect to PST overheads and connectivity of subscribers.

Problem 1 (The Minimum Overhead Publish/Subscribe Problem (MOPST))**.** Given an undirected connectivity $G_c = (V_c, E_c)$ where V_c is the set of vertices in G_c and E_c is the set of edges. Find a minimum overhead tree that is routed at r and spans all subscribers (Cao and Shen, 2009).

Theorem 1 (The Minimum Overhead Publish/Subscribe Problem in NP-complete)**.** The MOPST problem is NP-complete (Cao and Shen, 2009).

2.3 Publish/Subscribe Security

Much of the content-based publish/subscribe research has focused on the development of content-based matching and routing algorithms with the aim of increasing both performance and scalability. Until recently, much less attention had been paid to the issues of fault-tolerance, security, mobility and congestion control. Security in particular has yet to be fully explored and addressed, with much of the work constrained to addressing confidentiality through the use of access control mechanisms.

The design of content-based publish/subscribe systems predominantly assumes that there are implicit trust relationships between clients, clients and the event distribution service, and between all brokers within the event distribution service. These assumptions render the consideration of security issues extraneous as there are no malicious participants and consequently no potential threats. The application context, an event distribution service under the control of a single or multiple co-operating administrative entities, and contractual obligations between clients are just some of the reasons why such assumptions can be justified. Should these not hold true, all components of the system become vulnerable to a number of security concerns.

The security issues in publish/subscribe can be encapsulated into four groups: generic;

[1] http://tools.ietf.org/id/draft-ietf-manet-maodv-00.txt

confidentiality; accountability; and availability (Wang et al., 2002). The generic issues are similar to those that exist in other network paradigms and additionally they are also addressable by the use or modification of existing techniques. Authentication, anonymity and integrity are classified as generic issues. Confidentiality assumes that implicit trust no longer holds between publisher, subscribers and infrastructure, raising questions as to how to perform content-based matching and forwarding when publishers and subscribers trust neither one another nor the brokers with their events and filters. Accountability addresses the issue of how to ensure subscribers are correctly billed for events they consume. Availability of the system is susceptible to denial of service attacks that may exploit resource limitations of the clients and brokers, conceptual and theoretical design flaws of the content-based scheme, and any implementation vulnerabilities. Although Wang et al. provide an overview of the security issues of content-based publish/subscribe systems, there is only discussion of existing research that may provide possible solutions and no novel solutions are proposed.

This section describes the security issues that are specific to content-based publish/subscribe and relevant to the trust approach that is presented in this thesis, hence the exclusion of a discussion on accountability. Having described these issues, a number of proposed schemes are reviewed, which predominantly make use of either access-control or computing on encrypted data techniques.

2.4 Confidentiality

2.4.1 Definitions of Confidentiality

Definition 11 (Information, Subscription and Publisher Confidentiality)**.** There are three types of confidentiality in publish/subscribe systems. These are defined as follows (Wang et al., 2002):

- The infrastructure can perform content-based routing, without the publishers trusting the infrastructure with the content **(Information Confidentiality)**;
- The subscribers can obtain dynamic, content-based data without revealing their subscription functions to the publishers or infrastructure**(Subscription Confidentiality)**;
- Publishers can control which subscribers may receive particular publications **(Publication Confidentiality)**;

The implementation of a content-based publish/subscribe system that respects these confidentiality definitions requires the infrastructure to perform content-based routing and matching without access to the plaintext notifications and filters. Information and subscription confidentiality clearly conflict with content-based publish/subscribe, and this implies that the only suitable scheme is one where content-based matching algorithms can be implemented to match encrypted notifications and filters. In some application contexts, publishers and subscribers will not have complete distrust of the event based service, and may trust a subset of brokers. This gives the following definitions of information and subscription confidentiality:

Definition 12 (Confidentiality with Partial Trust of Infrastructure (Notification/Filter level))**.** Publisher confidentiality remains the same as in definition 11. Information and Subscription confidentiality are redefined as follows:

- The infrastructure can perform content-based routing, without the publishers trusting part of the infrastructure with the content. **(Information Confidentiality)**
- The subscribers can obtain dynamic, content-based data without revealing their subscription functions to the publishers or untrusted parts of the infrastructure. **(Subscription Confidentiality)**

These modified definitions imply the use of trust domains within the broker network, which can weaken reliability and efficiency of routing. Consequently, the confidentiality definitions can be further modified to allow distrusted brokers to perform content-based routing on attributes of events that are not considered confidential. This gives the following modification:

Definition 13 (Confidentiality with Partial Trust of Infrastructure (Attributes/Attribute Filter level))**.** Publisher confidentiality remains the same as in definition 11. Information and Subscription confidentiality are redefined as follows:

- The infrastructure can perform content-based routing, without the publishers trusting part of the infrastructure with all or some of the content. **(Information Confidentiality)**
- The subscribers can obtain dynamic, content-based data without revealing some or all of the attribute filters of their subscription functions to the publishers or untrusted parts of the infrastructure. **(Subscription Confidentiality)**

2.4.2 Definition of Confidential Content-Based Publish/Subscribe

A formal definition of confidential content-based publish/subscribe is given by Raiciu and Rosenblum (Raiciu and Rosenblum, 2006), which uses definition 11 of confidentiality. An ideal confidential content-based publish/subscribe protocol is defined by the authors as one where a broker has no access to plaintext notifications and subscriptions, only their indices, and so an oracle with access to the plaintext notifications and subscriptions is used to perform content-based matching that given the indices of a notification and subscription, returns true if the two match. Any confidential content-based publish/subscribe scheme must not leak any additional information to the broker. This definition is formalised and shows that the following properties must hold in such a scheme:

- **Notification Security** - notification ciphertext indistinguishability is guaranteed.
- **Subscription Security** - ciphertext representations are distinguishable using covering, however a stricter threat model may assume that they should not be.
- **Notification Unforgeability** - the inability of an attacker to forge arbitrary notifications appearing to be valid.
- **Subscription Unforgeability** - the inability of an attacker to forge arbitrary subscriptions that appear to be authentic.
- **Match Isolation** - the inability to perform any further computation other than testing for matching and covering using the oracle.

Raiciu and Rosenblum determine the following five algorithms that are essential for any confidential content-based publish subscribe implementation as given by their definition:

- `KeyGen(`t`)` - given a security parameter t, return a shared key K between a publisher and associated subscribers.
- `IndexSub(`K, S`)` - executed by a subscriber and given its subscription S and key K, returns an encrypted subscription, S_e.
- `IndexNot(`K, N`)` - executed by a publisher and given its notification N and key K, returns an encrypted notification N_e.
- `Match(`N_e, S_e`)` - returns 1 if $N_e \prec_S^N S_e$, 0 otherwise.
- `Cover(`S_{e_1}, S_{e_2}`)` - returns 1 if $S_{e_2} \prec_S^S S_{e_1}$, 0 otherwise.

Raiciu and Rosenblum go on to present a confidential content-based publish/subscribe scheme that abides by this definition(section 2.6.1.1). Prior to this research, Li et al. proposed a confidential publish/subscribe scheme using prefix-preserving cryptography (Li et al., 2004), but it does not guarantee notification security.

2.5 Availability

A Denial of Service (DoS) attack is performed by a malicious entity to disrupt legitimate access to a network service. If the attack is performed by multiple entities, then it is a Distributed Denial of Service (DDoS) attack. These attacks can vary significantly in their characteristics (Mirkovic and Reiher, 2004) and consequently there are a variety of mitigation techniques.

Mitigating against denial of service attacks that attempt to reduce the availability of publish/subscribe systems has surprisingly received very little research attention. A comprehensive taxonomy of DoS attacks in content-based publish/subscribe systems that categorises their effects and their properties is given by Wun et al. (Wun et al., 2007). Using the event-based middleware, PADRES (Fidler et al., 2005), the authors were able to identify the effects of these attacks, and classify the properties of DoS attacks. While no solutions are proposed, the authors express the hope that their work will lead to others instigating further research, however, this does not appear to have been the case.

2.5.1 CBPS Denial of Service Attack Characteristics

The experiments performed by Wun et al. allow the identification of three effects resulting from DoS attacks: localisation; workload complexity and message state. Localisation describes the behaviour where an attack that induces heavy load at an external broker may inhibit propagation of the attack due to very high input queuing delay, and therefore resulting in internal brokers being unaffected. A publication attack results not only in input queuing delays at the broker serving the attacker, but also increased input and output queuing delays at other external brokers with subscriptions matching the notifications being flooded, in spite of the localisation effect. This property is referred to by the authors as the transmission of the attack, as it disrupts resources beyond the initial broker node under attack.

The expressiveness of the content-based model allows for the crafting of notifications and filters that consist of many attributes or attribute filters respectively, and as these increase in number, so does the message complexity. The matching of complex notifications

and filters requires increased processing and memory requirements when compared to lower ones, so an attacker can take advantage of this by utilising a high complexity attack workload. As is expected, the authors' experimental results show that the effects of such an attack are more significant than ones of low complexity. At an affected broker, the rate of increase in response times is much greater and the rate of decrease after the conclusion of an attack is slower. The latter due to the backlogged messages of the workload that are contained in the input queue of the broker. These messages must be processed by the broker even after the attack has ended and this leads to a continuing increase in response time despite the attack having concluded.

In addition to transmission and workload complexity, attackers can also benefit from the maintenance of state in content-based publish/subscribe systems. Subscription flooding capitalises on the maintenance of filter state at the brokers by injecting malicious subscriptions at a high rate into the infrastructure and consequently increasing the memory consumption at brokers. The results of a low complexity constant rate subscription flooding attack experiment are given and they show that this attack, when compared to a publication flooding attack with similar properties, yields a more severe increase in memory consumption. The impact of the reduction of free memory at the broker leads to an increase in processing time of approximately two orders of magnitude, and consequently exponential growth in the response time of the broker.

The authors draw a number of conclusions having identified these effects from their DoS experiments: mitigating solutions might be able to utilise localisation to prevent attacks; workload complexity can currently be measured by attribute and attribute filter counts; and attacks taking advantage of message state could be mitigated by limiting the lifetime of notifications and subscriptions, and the use of policies to manage state. To this, it should be added that covering can help in limiting the effects of a message state based attack by not requiring the propagation of covered subscription filters, while merging may reduce the memory requirements at a broker. Disallowing blackhole advertisements (an advertisement that covers all subscriptions) and blackhole subscriptions (a subscription that covers all events) helps to prevent an attacker gain an understanding of the interests of clients, potentially limiting the number of brokers affected by an attack.

Having discussed these effects, a taxonomy of denial of service attacks in content-based publish/subscribe is given. The taxonomy proposes six classes of properties for attacks, these are: exploitation; source; target; propagation; statefulness; techniques; and content-dependence. The authors used the taxonomy to classify a number of attacks as to be using

either the stockpiling technique or to be of semantic weakness type. An example attack using stockpiling would be one where the attacker uses a blackhole advertisement to match all subscriptions in an effort to cause maximum disruption by a flooding of notifications. Semantic weakness allows an attacker to utilise theoretical or conceptual design flaws, for example the SIENA content-based model (Carzaniga et al., 2001) allows for unsubscribe and unadvertise operations which can be abused by attackers to remove advertisements and filters that they do not own from the infrastructure.

2.5.2 Solutions to Denial of Service Issues

In addition to attempting to address authentication and confidentiality, Eventguard (Srivatsa and Liu, 2005) implements a number of mechanisms that attempt to address DoS attacks. The authors identify three types of DoS attacks: flooding-based; fake unsubscribe and unadvertise; and selective or random message dropping. The use of signatures to prevent the dissemination of fake notifications and the rejection of duplicate notifications on the input queue are used to mitigate against flooding-based DoS, spam and spoofed messages.

2.6 Review of Existing Solutions

Wang et al. (Wang et al., 2002) suggest the application and in some cases modification of existing techniques to address the various classes of security issues. End-to-end and point-to-point authentication can be achieved by the use of public-key infrastructure. Computing with encrypted data (Abadi et al., 1987), secure circuit evaluation (Abadi and Feigenbaum, 1990) and private information retrieval (Chor et al., 1995) (Di Crescenzo et al., 2000) are proposed areas of interest that may help to provide confidentiality, but no solutions are given. The authors propose a number of techniques to provide accountability, including out-of-band solutions such as providing users with decryption keys with respect to any service level agreements, infrastructure-based solutions such as allowing perimeter broker nodes to perform accounting and trusted brokers to perform auditing tasks, and alternatively by the possible use of verifiable secure computation (Gennaro and Micali, 1995). Although many initial ideas and areas of further related research are proposed, it appears that most of these have not been further explored by the research community. To date, the key contribution of this work has been to identify and raise awareness of security issues, and to define confidentiality in the context of publish/subscribe.

2.6.1 Confidential Content-Based Publish/Subscribe

2.6.1.1 Raiciu and Rosenblum

The definition of a confidential content-based publish/subscribe system has been formalised by Raiciu and Rosenblum (Raiciu and Rosenblum, 2006) (section 2.4.2). Confidential content-based techniques are proposed for equality, substring and numeric matching. An implementation based on SIENA and requiring little modification is presented and additionally the authors claim that their proposed techniques can be implemented in other content-based event-distribution systems. The basis of the techniques is to implement content-based matching on encrypted data.

Equality matches are implemented using a pseudorandom function F. A subscription value S is passed to function F that is keyed with a shared secret key K, i.e. $F_K(S)$. Notifications are encrypted by first passing the notification value to F keyed with S, then using the returned value as the key for encrypting a uniform random nonce, rnd, i.e. $F_{F_K(N)}(rnd)$. An encrypted notification N_e matches an encrypted subscription S_e if $F_{F_K(N)}(rnd) = F_{S_e}(rnd)$, so in addition to N_e and S_e the only information revealed to the infrastructure is rnd. Given two subscriptions S_{e_1} and S_{e_2}, S_{e_1} covers S_{e_2} if $S_{e_1} = S_{e_2}$.

As substring matching is computationally expensive, this motivates the authors to implement keyword matching, as they believe this to be adequate for most applications. The mechanisms proposed are an application of those originally devised by Goh (Goh, 2003). Given r hash functions in a Bloom filter (Bloom, 1970) BF, a master key is $K = (k_1, k_2, \ldots, k_r)$ drawn uniformly at random from $\{0,1\}^{rt}$ where t is some security parameter, a subscription is encrypted as $(F_{k_1}(S), \ldots, F_{k_r}(S))$. A notification N is encrypted by first extracting keywords $w_1, w_2, \ldots, w_n$, then for each keyword w_i is encrypted as if a subscription, $(x_{i,1}, \ldots, x_{i,r}) = (F_{k_1}(w_i), \ldots, F_{k_r}(w_i))$, codification is completed by setting each bit of the Bloom filter, $BF[F_{rnd}(x_{i,r})] = 1$ for each r. An encrypted notification and subscription, $N_e = (rnd, BF)$ and $S_e = (x_1, x_2, \ldots, x_r)$ respectively, are not matched if for some i to r, $BF[F_{rnd}(x_i)] = 0$. The covering test is identical to that of equality matching.

Techniques for inequality and range subscriptions are presented that are both based on the secure index scheme proposed by Chang and Mitzenmacher (Chang and Mitzenmacher, 2005), however they are not without their disadvantages. The inequality operator is not a function on a pair of real values, but instead an approximation that operates on a pair (x, y) where $x \in \mathbb{R}$ and $y \in D$ such that $D \subset \mathbb{R}$ and D must be agreed upon between publisher and subscribers. This reduces expressiveness and introduces additional coupling.

The range matching algorithm is an approximation scheme with a trade-off between the subscription size and matching times against the number of false positives and likelihood of information leakage. Despite these disadvantages, the techniques are proven to be correct implementations of confidential content-based publish/subscribe.

Evaluation of these encrypted matching techniques is performed using a modified version of SIENA. Inequality matching with and without covering is 1.7 and 3 times more expensive, respectively. Range matching times similarly see a benefit from utilising covering, as they are only 3 times more expensive than plaintext compared to 6 times more expensive without the use of covering. Equality matching times are also six times more on average than plaintext. Increases in communication overhead are more significant in scale, with notifications 15 times larger, and subscriptions 10 times larger than their plaintext representations.

The authors identify several limitations applicable to all confidential content-based publish/subscribe schemes as defined by definition 11: the "Attack at Dawn" problem; limited indistinguishability; confidentiality-generality tradeoff; and trust. While the "Attack at Dawn" problem can be mitigated using anonymising techniques, the use of a shared key between publisher and subscriber not only increases the coupling between these entities, but also requires the assumption that they will not leak it. Determining when a leak has occurred, its source, and its recipients is non-trivial.

In addition to the issues that are applicable to all confidential publish/subscribe schemes, there are a number that are specific to the techniques described above. The content-based model is limited in its expressiveness, substring support is replaced by the less granular keyword matching, and the inequality tests can only be preformed against a subset of real numbers, although perfect matching is possible with high space and consequently time complexity. The overhead of the inequality scheme is $2 \cdot l$ where l is the size of the dictionary, i.e. number of reference points.

All forms of confidentiality given in definition 11 are met (it is the only work surveyed that achieves this) and there no longer needs to be any trust between entities with notification and subscription content, however, this is replaced by a trust relationship over the shared key. Unless a mechanism is devised allow for effective monitoring such that key leakage can be detected and the access rights of the source of the key revoked, then a dependence on implicit trust remain.

2.6.1.2 Li et al.

Prior to the scheme of Raiciu and Rosenblum, Li et al. (Li et al., 2004) proposed a confidential publish/subscribe scheme that is based on the premise that interval matching can be transformed to prefix matching, and that using a prefix preserving encryption function, notifications and subscriptions can be encrypted and matched in their non-plaintext form by a broker. As is the case with the Raiciu and Rosenblum scheme, a shared key between publisher and subscribers is required, however, Li et al. propose some justification for this by arguing that their scheme is designed for private publish/subscribe systems with an event infrastructure under the control of a single entity.

An interval $[32, 111]$ using 8-bit binary representation can be transformed to the set of prefixes, $\{001*, 010*, 0110*\}$, an algorithm for which is given by the authors. The encryption and decryption algorithms for prefix-preserving IP address anonymisation (Fan et al., 2004) can then be applied to the set of prefixes. Having agreed upon a shared key K, a pseudorandom prefix-preserving function F_K, is used to encrypt the attributes of a notification, N. A subscriber uses the authors' interval to prefix algorithm for each attribute filter, encrypting each using the same pseudorandom prefix-preserving function. Brokers are able to establish if notifications are covered by subscription filters by matching the encrypted set of prefixes against the encrypted bits of the notification.

The algorithm is shown to be computationally efficient and in the order of n where n is the number of bits in the interval, however, this technique leaks information with the justification given by the authors that it is a necessary trade-off against efficiency. A malicious entity in the publish/subscribe system given a plaintext-ciphertext pair, can attempt to ascertain the prefix bits given another ciphertext. The more pairs an entity is able to acquire, the probability of prefix bits being leaked increases. As a result of this, notification indistinguishability can not be guaranteed. While the authors go on to propose the use of different keys for each attribute and time bounded prefix-preserving pseudorandom function, these only help to mitigate against this security issue.

2.6.1.3 Other work on Confidentiality

The SIENA fast forward implementation[2] contains an implementation of confidential exact matching, but that it is insecure as notification are distinguishable (Raiciu and Rosenblum, 2006). Eventguard (Srivatsa and Liu, 2005) implements confidentiality using per-topic shared keys for topic-based publish/subscribe systems.

[2] http://www.inf.usi.ch/carzaniga/cbn/forwarding/index.html

2.6.2 Access Control

2.6.2.1 Access Control Upper Bound Filters

Miklós describes a method to define access control policies on clients' advertisement and subscription filters (Miklós, 2002). These policies are upper bound filters that use the covering and strict covering semantics in conjunction with client credentials to define positive access rights for subscriptions and notifications. Using the SIENA content-based model, upper bounds access rights for publishers and subscribers can be given by access control filters that are associated with clients' credentials. The access control filter only allow a broker to accept a filter or notification from a client if it is covered by the control filter and if the control filter is associated with the client's credentials.

The approach adopted addresses a number of attack scenarios as defined by the author. Fake advertisements, blackhole advertisements and blackhole subscriptions can not be performed if control filters are appropriately defined by the access control management entity, and this leads to a reduced risk to stockpiling and message state based attacks. Message state attacks can be further limited as an attack can not enact a flooding of arbitrary subscriptions to the infrastructure in an attempt to increase the memory consumption at brokers. Publisher flooding is also restricted and access violations can no longer occur due to the use of credentials.

The techniques proposed do not address any form of confidentiality. Clients must submit their advertisement, notifications and subscriptions to the infrastructure with the assumption that it can be trusted. A screening technique is proposed that attempts to provide some form of publisher confidentiality by allowing a publisher to permit subscribers to only view the notification attributes that are matched by an attribute filter in their subscription.

2.6.2.2 Scopes

Fiege et al. (Fiege et al., 2004) attempt to address the issue of implicit trust between publishers, subscribers and infrastructure by defining the concept of scopes which can be used to map external trust defined by external contracts between entities to the publish/subscribe system. Scopes allow for the grouping of a publisher and its trusted subscribers, thereby limiting the visibility of notifications to these clients, however, there is an inheritance relationship between scopes that allows notifications to be propagated to a super-scope if the scope interface between the two allows this.

An implementation of scopes using REBECA (Mühl, 2002) is described with particular

focus on the changes required to the routing tables, such that these are split between scopes, and as a result grouping brokers within the infrastructure in a given scope in addition to the clients, effectively creating an overlay. A scope join request issued at a broker is processed by the first broker in the scope, with the reply to the requesting broker sent on the return path to allow all brokers on it to a create routing table entries for the scope. All nodes on this path join the scope.

Access control is implemented at the external brokers of the infrastructure. A publisher submits an advertisement to an external broker along with an attribute certificate[3] that is used as the publisher's credential. Given the credential, the advertisement is either discarded if not valid, or propagated within the scope along with an additional certificate, containing the public key of the publisher, which is used to authenticate subscribers at external brokers. A subscriber requesting to join the scope, submits its attribute certificate to the external broker, which verifies the signature of the certificate using the publisher's public key. Border brokers ensure that only authorised clients participate in the scope and ensures through the attributes of the content providers' attribute certificate that there are no access violations on publish/subscribe API calls.

The authors address a number of the security issues of scopes. As mentioned above, the scope can be extended, and the addition of the broker not only adds it to the scope, but also adds brokers on the path to the scope. These nodes may be distrusted and should not be included. The proposed mechanism to prevent this is for each broker on the path, from the joining broker to the broker currently in the scope, to append its attribute certificate to the request. The broker processing the join request then tests the attribute certificates with respect to any trust management policies. For those circumstances where the broker has only distrusted paths to a scope that it wishes to join, tunnelling can be used to create a connection to it.

The work addresses trust explicitly through the creation of trusted overlays upon the publish/subscribe network to create a trusted environment for communication. While the use of scopes does indeed overcome the security issues induced through the implicit trust assumed by content-based publish/subscribe, scopes reduce the fault-tolerance, as there is a trade-off between trust and path redundancy of the infrastructure. Application of the techniques to application contexts where there are no external contracts between publishers and subscribers is not discussed, however, with the addition of monitoring techniques and reputation management, scopes could be extended to these scenarios.

[3] http://www.ietf.org/rfc/rfc3281.txt

2.6.2.3 Role-based Access Control

Belokosztolski et al. (Belokosztolszki et al., 2003) propose the use of Role-based Access Control (RBAC) (Sandhu et al., 1996) in publish/subscribe systems with the research goal of implementing security management in the publish/subscribe middleware with minimal modifications to the publish/subscribe API and its algorithms. The proposed system implements RBAC on the HERMES event-based middleware (Pietzuch and Bacon, 2002) using the Open Architecture for Secure Interworking Services (OASIS). RBAC provides the following three mechanisms (Mühl et al., 2006): restrictions on the interaction of event clients; trust levels for event-brokers; and encryption of event data to control information flow in the system on a fine-grained basis.

In confidential publish/subscribe, the assumption held is that the entire infrastructure is distrusted, however, this is not the case for the proposed RBAC solutions. Instead, the infrastructure is assumed to be trusted to perform content-based matching and forwarding, but certain brokers may not be trusted to access all attributes of notifications. Clients are not implicitly trusted, requiring the use of access control mechanisms at edge brokers. These trust relationships imply that the strict definition of confidentiality given in 11 is not met, but instead the weaker form that is given in definition 13.

The RBAC publish/subscribe model assumes that the publish/subscribe model is defined with respect to type-based publish/subscribe, however, Mühl et al. (Mühl et al., 2006) state that it supports access control policy specification using notification attributes and subscription filters. Each client is assigned a credential that is used to determine its role. Using OASIS, the model allows the event type owner to specify the following role-based access control policies at the edge brokers:

- **Broker-Client Connection Policy** - given a client's credential, it is used to determine if the client should be hosted by the broker.
- **Type Management Policy** - given a client's credential, it is used to determine if the client is allowed to create, modify or remove event types.
- **Advertisement Policy** - allows the event type owner to restrict the roles that are permitted to publish matching notifications.
- **Subscription Policy** - allows the event type owner to define the roles that are allowed to issue subscriptions for the given event type.

These policies are implemented at the edge brokers. Any client performing an action at

an edge broker must provide its credentials so that the relevant policy can be used to permit or disallow the action and ensure that no access violation can occur. For advertisement and subscription policies, restriction techniques are proposed that transforms an advertisement or subscription filter to one that is more restrictive. For content-based publish/subscribe this is defined by an upper bound control filter as described by Miklós (see section 2.6.2.1).

Under certain conditions, it is not possible to assume that the infrastructure can be trusted to access the content of notifications and subscription filters. By assigning brokers to roles, access to a set of event types and their sub-types can be specified by policies defined with respect to these roles. The use of a web of trust of X.509 certificates rooted at the event type owner is proposed to implement this mechanism. The web of trust allows brokers to ensure that for a given event, it can be forwarded to other brokers that have been assigned to a role, which is permitted to access the event's type. It can also be used by a publisher to ensure that its hosting broker is permitted to accept and propagate its advertisement.

Belkosztolski et al. (Belokosztolszki et al., 2003) propose that if required, events may be encrypted to prevent eavesdropping, however, the approach of specifying access control on event types creates trust domains in the infrastructure, resulting in partitioning that reduces redundancy and efficiency (Mühl et al., 2006). Pesonen et al. (Pesonen et al., 2007) propose the implementation of access control using encryption of either event types or attributes. For event type security, types are associated with a key, and for attribute security, each attribute is associated with a key. Attribute encryption allows for a much finer granularity of access control, so brokers can perform content-based routing on attributes that they are permitted to access. Where the broker has no access rights to the event contents, routing is performed using the unencrypted event type identifier. The proposed system uses the Advanced Encryption Standard (AES) (Nechvatal et al., 2001) in EAX mode (Bellare et al., 2004), and consequently symmetric key use. A capability-access model (Pesonen et al., 2006) is used to determine if a broker has been granted access rights to the encryption keys by the event type owner. As expected, experimental results show that event throughput is greater with event type encryption than attribute encryption due to the increased overhead of the latter, but attribute encryption results in lower hop counts, as more information is available to the broker network for routing.

The use of RBAC in conjunction with publish/subscribe addresses a number of security issues. Publisher confidentiality can be guaranteed, but the strictest forms of information and subscription confidentiality can only be met in part. At least some of the infrastruc-

ture must be trusted so that the content-based matching and forwarding algorithms can operate correctly, thereby breaching information and subscription confidentiality, but for many applications this will likely be acceptable. As expected, experimental results show attribute encryption results in lowered throughput in the infrastructure when compared to event-type encryption and no encryption, but reduced hop counts against event-type encryption.

RBAC will not prevent attacks by entities assumed to be trusted, but who have become malicious, and this is complicated further when trusted clients and brokers sporadically do not behave as expected. Policies would have to be manually modified by the event type owners and if encryption is used, keys revoked and reissued to comply with the new policy and to expel the malicious entity from those that are trusted. To implement this more successfully, without the opportunity for malicious reporting of misbehaviour, monitoring mechanisms must be defined whose observations are used to determine the reputation and trustworthiness of entities which are then used to influence policy. Finally, the RBAC publish/subscribe model only considers trust relationships between publisher and infrastructure, and subscriber and infrastructure. Scenarios where brokers do not trust one another are not considered. These issues are also true for the for scope technique described above.

2.6.3 Spam

Publish/Subscribe is a many-to-many form of communications where, as is the case with e-mail, the cost of communication is cheap. This allows attackers to propagate unsolicited and bogus messages, such as spam and bogus messages. It is this that motivates Tarkoma (Tarkoma, 2006) to investigate the problem of content-based spam and leads to the proposal of a system that blacklists the public-key identities of publishers and brokers that disseminate spammers.

Content-based spam can be classified as either inbound or outbound. Inbound describes unwanted messages in the input queue of a broker that are matched by a filter in its routing table. Outbound are malicious messages that a broker attempts to not propagate on matching filters. A number of mitigating techniques such as sender verification and blacklisting are proposed by the author for both types of spam.

With e-mail, endpoints are defined by the unique e-mail address, so an attacker only needs to have a list of addresses to target to perform the attack. In content-based publish/subscribe, the endpoints are described by filters and not addresses. Only if a no-

tification matches a filter will it be forwarded. As a result of the content-based model, an attacker must be aware of the subscription filters in the infrastructure to maximise the subscribers who will receive the spam. Should the publish/subscribe API allow it, blackhole advertisement and subscriptions allow the attacker to learn about the interests of the clients, where the former can be used to determine all subscriptions at a given time, and the latter matching all notifications submitted to the infrastructure. Depending on the structure, an alternative to the blackhole techniques is for an attacker or clique of attackers to infiltrate the structure of the broker network at particular positions to sample filters and notifications. For example, in a publish/subscribe system that makes use of a hierarchical structure the root node would be the optimal position at which to sample communications.

Tarkoma's proposed solution to spam is to allow publishers and brokers to be marked as blacklisted in a blacklist that is implemented using a distributed hash table (DHT). This requires the use of public key identities issued by a certificate authority, so the publish/subscribe system must be augmented with not only a DHT, but also public-key infrastructure. Each advertisement and notification submitted to the infrastructure is signed at each hop and dropped by a broker if it is signed by a blacklisted public-key identity. In order to ensure that a publisher or broker is not maliciously blacklisted, several observations of malicious behaviour are required for blacklisting. The author cites two disadvantages with this solution, the first is that may give rise to network partitioning, and the second is that the additional calls to the DHT and the verification of signatures can introduce overheads such as increased response times and larger messages sizes respectively.

As with scopes, the proposed system could be extended with monitoring and reputation management to provide a more comprehensive solution to ensure trustworthy behaviour in publish/subscribe. Unfortunately, due to an absence of experimental results, it is difficult to assess the extent of the potential for network partitioning and the overheads due to DHT lookups and signature verification.

2.7 Summary

In this chapter, the PST preliminaries have been presented, followed by a survey of security issues in publish/subscribe systems and approaches to address these. The PST preliminaries include a definition of a PST and the overhead metric that is used in later chapters. Disadvantages to the various security approaches surveys are identified when juxtaposed to the proposed creation of a trusted PST, most notably that the systems are

unable to adapt changes in behaviour of nodes.

Chapter 3

Trusted Publish/Subscribe Trees

3.1 Social Choice and Welfare Preliminaries

3.1.1 Introduction

Social choice theory is described as being "concerned with relationships between individuals' preferences and social choice" (Fishburn, 1973) and alternatively by Suzumura as "being concerned with the evaluation of alternative methods of collective decision-making, as well as with the logical foundations of welfare economics" (Arrow et al., 2002). The latter goes on to state that "as soon as multiple individuals are involved in making decisions for their common cause, one or other method of collective decision-making cannot but be invoked". Social choice theory is the study of the specification of preferences, their motivating utilities, and the aggregation mechanisms of individual preferences to a socially acceptable preference, all underpinned by concepts from welfare economics.

The history of social choice theory dates back to the late 1700s. Its theoretical foundations in collective-decision making begin in 1781 when the Académie Royale des Sciences published Borda's positional voting scheme[1] (Borda, 1781) and adopted it for the election of members to the academy from 1780 to 1784 when the voting scheme was attacked by Napoléon and consequently revoked. As a member of the academy and the First Consul of the First French Republic, Napoléon[2], was not the only detractor of the Borda count. A fellow member, the Marquis de Condorcet, opposed Borda's scheme, and so he proposed an alternative using simple majority pairwise comparisons of all candidates' rankings and defined the voting paradox in which social preference may be cyclic despite individual preferences being transitive[3] (Condorcet, 1785). Suzumura cites the Condorcet paradox

[1] http://gallica.bnf.fr/ark:/12148/bpt6k35800/f787
[2] Membership of the academy was not restricted to scientists.
[3] http://gallica.bnf.fr/ark:/12148/bpt6k417181/f4

as signalling the requirement for a formal treatment of collective decision making (Arrow et al., 2002), but it was well over one hundred years later that significant research contributions were made towards this by Arrow with his definition of the social welfare function and his impossibility theorem (Arrow, 1951) (Arrow, 1963). Arrow's axiomatic framework allows for the analysis and comparison of social welfare functions, and proves that none can satisfy all of a set of reasonable conditions.

The origin of welfare economics is widely regarded to be the work of Bentham who proposed that moral actions were those that maximised pleasure as opposed to those that maximised pain and presented the Felicific calculus (Bentham, 1781) to measure utility. One property of this calculus is to take into account the number of individuals who obtain pleasure from the action under consideration. Mill drew upon this to formulate the greatest happiness principle that is widely described in the literature as "greatest happiness of the greatest number"[4] (Mill, 1863), that is the maximisation of the utility of the majority.

Rather than adopt a utilitarian approach to determining the most trusted PST, the trust framework prefers a PST that maximises the trustworthiness of the least well-off node. This method is motivated by Rawls' principles of justice, and more precisely, the difference principle (Rawls, 1971). This principle is part of the second principle of justice and states that social and economic inequalities satisfy the condition that they are to be to the greatest benefit of the least advantaged members of society. Rawls' goes on to state that under these principles of justice, a society would favour a social choice that maximises the least well-off, the maximin principle. In this work, the leximin social welfare functional is used, which is an extension of the maximin principle.

Another aspect of social theory considered in this section is interpersonal comparability of utility. There has been much discussion as to whether utility is individually measurable and interpersonally comparable. Utilitarians supported this view, as it allows for the maximisation of the utility of a society, but it was later rejected by others and ultimately so by Arrow in his definition of the social welfare function that assumes preferences to be ordinal and of incomparable interpersonal utility (Arrow, 1951), and proposing that "interpersonal comparison of utilities has no meaning and, in fact, that there is no meaning relevant to welfare comparisons in the measurability of individual welfare". To justify this, it is claimed that measurability of utility is meaningless as comparing mar-

[4]This quote used to describe the principle is from prior work on morals by Hutcheson in Inquiry concerning Moral Good and Evil, 1725.

ginal utility differences at different levels of well-being is not possible[5]. This implies that Arrow disagreed with Gossen's first law (Gossen, 1854), commonly referred to as the law of diminishing marginal utility, which states that the marginal utility (the utility gain or loss) of the consumption of a unit of service decreases as the supply of units increases (i.e. the utility gain from the first unit of consumption is greater than the second and so on), as this assumes the measurability that he postulates is not possible. Robbins argues that it is questionable that mental states can be measured (Robbins, 1935), as is assumed to be the case for Gossen's laws. He proposes that while it is possible to determine if some individual has a greater marginal utility from one state than another (intrapersonal comparability), as one can ascertain this from the subject's introspection or by observation, it is not possible to compare the magnitude of satisfaction of different individuals, neither by comparing the individuals' introspection nor by observation. This theory contradicts Gossen's first law, as it is not possible to measure and compare the marginal utilities of individuals, and it contradicts interpersonal comparability of utility.

Sen argues that interpersonal comparability of utility is possible using the following as an example to justify his argument: "We may, for example, have no great difficulty in accepting that Emperor Nero's utility gain from the burning of Rome was smaller than the sum-total of the utility loss of all the other Romans who suffered from the fire" (Sen, 1999). Clearly such a statement is not rational given the theory put forward by Robbins, but it is difficult to disagree with it. Sen also urges caution as there is not a one-to-one mapping between individuals' utilities in all scenarios (as may be the case in the example of the burning of Rome), and this leads to his conclusion that utility may be fully comparable, partially comparable or incomparable. As the social welfare function defined by Arrow can not consider interpersonal comparability, Sen proposed the Social Welfare Functional (Sen, 1977). These differ to social welfare functions in that they aggregate individuals' utility functions to a social preference without making any assumptions as to the type of the utility function, and allow for interpersonal comparability.

Social choice theory has a long history, but much of the work covered in this chapter and used to determine the trust metric for publish/subscribe trees is relatively recent. In this chapter, a predominantly mathematical treatment of social choice and welfare theory is given that covers social welfare functions, a discussion of cardinal and ordinal preferences, impossibility theorems and egalitarianism. Social choice theory is a vast domain, so this

[5]Temperature differences are often given in the literature, as an analogy to support this argument. Arrow gives the example of a 1 °C from 0 °C to 1 °C having a different impact than of 100 °C to 101 °C. We add that the significance is dependent upon the individual.

review is not complete, and covers only the necessary topics that are required to understand the formulation of the trust metric for PSTs.

3.1.2 Social Welfare Function

Definition 14 (Preference Ordering)**.** A preference ordering is a binary relation R over a set of alternatives C that satisfies the following properties:

1. R is complete. $\forall x, y \in C. xRy \vee yRx$.
2. R is transitive. $\forall x, y, z \in C. xRy \wedge yRz \implies xRz$.

The binary relation, R, is a weak order. In order to fully capture the properties of preference, both preference, P, and indifference, I, are defined in terms of R.

1. xRy **(Weak Preference)**
2. $xPy \implies \neg yRx$ **(Strict Preference)**
3. $xIy \implies xRy \wedge yRx$ **(Indifference)**

Definition 15 (Society)**.** A society is a set of individuals $N = \{1, 2, \ldots, n\}$.

Definition 16 (Profile)**.** Given a society of n individuals, a profile is an n-tuple of preference ordering, $(\pi_1, \pi_2, \ldots, \pi_n)$ where π_i is the preference ordering of the i^{th} individual of the society.

Definition 17 (Social Welfare Function)**.** A social welfare function is a mapping $F : \mathcal{O} \rightarrow \Pi$ where the domain, $\mathcal{O}$, is the set of all profiles for a society's n individuals and over the set of C alternatives, and the co-domain is the set of all preference orderings, Π over the set C of alternatives.

3.1.2.1 General Possibility Theorem

Arrow originally proposed five reasonable conditions for social welfare functions (Arrow, 1951), however, non-imposition (the social welfare function is a surjective function) and monotonicity (if an alternative rises in one or more individual preferences, the alternative will not fall in the social preference ordering) were later replaced by pareto-efficiency (Arrow, 1963), as the latter can be derived from the two replaced axioms and independence of irrelevant alternatives. Given the axioms of unrestricted domain, pareto-efficiency, independence of irrelevant alternatives and non-dictatorship, Arrow presented the General

Possibility Theorem, which proved that no social welfare function can satisfy all these reasonable conditions.

Axiom 1 (Unrestricted Domain)**.** Arrow defines this axiom as "All logically possible orderings of the alternative social states are admissible". The domain $\mathcal{O}$ of the social welfare function F is the set of all logically possible n-tuples over the set of alternatives C.

Axiom 2 (Pareto-Efficiency)**.** For all $x, y \in C$ and for each individual i of a society, if xP_iy then xPy.

Axiom 3 (Independence of Irrelevant Alternatives)**.** Let $(\pi_1, \pi_2, \ldots, \pi_n)$ and $(\pi'_1, \pi'_2, \ldots, \pi'_n)$ be two profiles for a society of n-individuals and where each π_i and π'_i is a preference ordering over the set of alternatives C. For all individuals i and all $x, y \in C$, $xR_iy \iff xR'_iy \implies xRy \iff xR'y$. That is the social preference ordering of x and y are solely dependent on their orderings in the individual preference orderings.

Axiom 4 (Non-Dictatorship)**.** A social welfare function is dictatorial if there is an individual i in a society such that for all $x, y \in C$, $xP_iy \implies xPy$.

Theorem 2 (Arrow's General Possibility Theorem)**.** If $|C| \geq 3$ then there is no social welfare function F that satisfies the axioms of unrestricted domain, pareto-efficiency, independence of irrelevant alternatives and non-dictatorship. The above axioms are proved to be inconsistent.

3.1.3 Social Welfare Functionals

Arrow's proof shows that the only social welfare function to be of unrestricted domain, pareto-efficient and independent of irrelevant alternatives is dictatorship. Sen identified that the absence of interpersonal comparability of preferences as the reason for this, and as a result, proposed the social welfare functional that mapped profiles of individual utility functions to a social preference ordering (Sen, 1970, 1977). By allowing ordinal and cardinal comparability, the General Possibility Theorem result no longer applied, however it is important to note that utility can not always be considered to be comparable.

Definition 18 (Individual Evaluation Function)**.** An individual utility function is a real-valued function to the set of alternatives C for an individual i, $u_i : C \to \mathbb{R}$.

Definition 19 (Representable)**.** A preference ordering, R is said to be representable by an evaluation function u, if and only if $\forall x, y \in C. xRy \iff u(x) \geq u(y)$.

Definition 20 (Profile)**.** A profile U is an n-tuple of utility functions, $(u_1, u_2, \ldots, u_n)$. It can also be defined as a real-valued function, $U : C \times N \to \mathbb{R}^{|N|}$ where $U(\cdot, i) = u_i$ and $U(c, \cdot)$ is a restriction of U to c, the evaluation vector of alternative $c \in C$.

Definition 21 (Social Welfare Functional)**.** A social welfare functional is a function $F : \mathcal{U} \to \mathcal{R}$ where $\mathcal{U}$ is the set of all permissible profiles and $\mathcal{R}$ is the set of all preference orderings.

By allowing interpersonal comparability, social welfare functionals need not be dictatorial and hold the other desirable properties given by Arrow, thereby avoiding the general possibility theorem. The leximin social welfare functional used in the trust metric for PSTs utilises the cardinal full comparability (definition 29). All of the following definitions in the remainder of this subsection can be found in (Sen, 1970, 1977). Definitions 25 to 29 define the various types of ordinal and cardinal comparability for individual utility functions. Definitions for partial comparability (Sen, 1970) are omitted, as they are not utilised.

Definition 22 (Set of Individual Real-Valued SWFLs)**.** Let L_i be the set of real-valued social welfare functionals over the alternatives C for individual i.

Definition 23 (Set of all profiles)**.** Let L be the cartesian product, $\prod_{i=1}^{n} L_i$ ($L = \mathcal{U}$ in definition 21).

Definition 24 (Comparability Set/Invariance Requirement)**.** Let $\bar{L} \subset L$ be a comparability set if and only if $\forall U, U' \in \bar{L} : F(U) = F(U')$ where F is a social welfare functional.

Definition 25 (Ordinal Non-Comparability)**.** For all $(u_1, u_2, \ldots, u_n), (u'_1, u'_2, \ldots, u'_n) \in \bar{L}$, there is a n-tuple of positive monotonic functions, $\Psi = (\psi_1, \psi_2, \ldots, \psi_n)$, such that $\forall i \in N . u'_i = \psi_i(u_i)$.

Definition 26 (Ordinal Level Comparability)**.** For all $(u_1, u_2, \ldots, u_n), (u'_1, u'_2, \ldots, u'_n) \in \bar{L}$, there is a positive monotonic function, ψ, such that $\forall i \in N . u'_i = \psi(u_i)$.

Definition 27 (Cardinal Non-Comparability)**.** For all $(u_1, u_2, \ldots, u_n), (u'_1, u'_2, \ldots, u'_n) \in \bar{L}$, there is a n-tuple of positive affine functions, $\Psi = (\psi_1, \psi_2, \ldots, \psi_n)$, such that $\forall i \in N . u'_i = \psi_i(u_i)$.

Definition 28 (Cardinal Unit Comparability)**.** For all $(u_1, u_2, \ldots, u_n), (u'_1, u'_2, \ldots, u'_n) \in \bar{L}$, there is a positive real n-vector $\mathbf{a}$ and a positive real number b, such that $\forall i \in N . u'_i = a_i + b \cdot u_i$.

Definition 29 (Cardinal Full Comparability)**.** For all $(u_1, u_2, \ldots, u_n), (u'_1, u'_2, \ldots, u'_n) \in \bar{\mathcal{U}}$, there is a positive affine function, ψ, such that $\forall i \in N. u'_i = \psi(u_i)$.

For cardinal full comparability (definition 29), there is a one-to-one mapping between the individuals' utility functions, so not only are the units of welfare comparable, but also their origins, implying that absolute values and differences of utility are inter-personally comparable. This is not the case for cardinal unit comparability (definition 28), where the transformation shifts the origin with respect to the individual specific parameter, a_i, so origins are not comparable across individuals. This rules out the inter-personal comparison of absolute values of utility, but differences of utility are still comparable. Ordinal full comparability (definition 26) requires that a monotonic function that preserves the ordering across individuals' utility functions.

Below, axioms 1 - 4 are redefined for social welfare functionals and the impossibility theorems with respect to these axioms are presented. Sen proves that no ordinal and cardinal non-comparable SWFLs can satisfy all the axioms, theorems 3 (identical to theorem 2) and 4. By weakening the axioms, that is by removing one of them as a requirement, or by weakening theorem 3 so that the SWFL is ordinal level comparable, a possibility arises.

Axiom 5 (Unrestricted Domain (SWFL))**.** Identical to Axiom 1.

Axiom 6 (Pareto-Efficiency (SWFL))**.** For all $x, y \in C$ and for each individual i of a society, if $u_i(x) > u_i(y)$ then xPy.

Axiom 7 (Independence of Irrelevant Alternatives (SWFL))**.** Let $(\pi_1, \pi_2, \ldots, \pi_n)$ and $(\pi'_1, \pi'_2, \ldots, \pi'_n)$ be two profiles for a society of n-individuals and where each π_i and π'_i is a preference ordering over the set of alternatives C. For each individual i, $xR_iy \iff xR'_iy \implies xRy \iff xR'y$.

Axiom 8 (Non-Dictatorship (SWFL))**.** A social welfare function is dictatorial if there is an individual i in a society such that for all $x, y \in C$, $u_i(x) > u_i(y) \implies xPy$.

Theorem 3 (Ordinal Non-comparable SWFL Impossibility)**.** There is no social welfare function that is ordinally non-comparable and for which the axioms 5, 6, 7, and 8 hold true.

Theorem 4 (Cardinal Non-comparable SWFL Impossibility)**.** There is no social welfare function that is cardinally non-comparable and for which the axioms 5, 6, 7, and 8 hold true.

3.1.4 Leximin Social Welfare Functional

The leximin social welfare functional captures the definition of the difference principle by maximising the welfare of the least well-off individual. Unlike the maximin approach, leximin allows for ties to be broken by maximising the welfare of the $n-1, n-2, \ldots$ least-well off until the tie is broken. In the following definition of the leximin social welfare functional, pareto-efficiency and non-dictatorship are substituted by the stricter strong pareto and anonymity axioms, respectively. Either ordinal level comparability or cardinal full comparability with Hammond's equity axiom (Hammond, 1976) must be satisfied by the leximin SWFL. In this work, the latter is used, as the individual utility functions to be aggregated are cardinal trust functions.

Axiom 9 (Strong Pareto Efficiency)**.** For all $x, y \in C$ and for each individual i of a society, if $u_i(x) \geq u_i(y)$ then xRy and if there is an individual i such that $u_i(x) > u_i(y)$ then xRy.

Axiom 10 (Anonymity)**.** If U' is a reordering of U, then $F(U') = F(U)$.

Axiom 11 (Hammond's Equity Axiom)**.** If there is a profile U over a set of alternatives C, any two individuals $m, n \in N$ and any two alternatives, $x, y \in C$, such that $u_m(y) < u_m(x) < u_n(x) < u_n(y)$ and $\forall i \in N \setminus \{m, n\} : u_i(y) = u_i(x)$ then xRy.

Definition 30 (Leximin Social Welfare Functional)**.** Let $i(x)$ be the i^{th} worst-off individual under the alternative x, that is there is a subset $M \subset N$ where $|M| = i - 1$ individuals such that for all $m \in M$, $u_i(x) \geq u_m(x)$. For any given pair of alternatives $x, y \in C$, xPy if and only if there is an $i \in N$ such that:

1. $u_{i(x)}(x) > u_{i(y)}(y)$; and
2. $u_{m(x)}(x) = u_{m(y)}(y)$ where $m \in M$.

If $\forall i \in N : u_{i(x)}(x) = u_{i(x)}(x)$ then xIy.

Theorem 5 (Leximin Properties)**.** Any social welfare functional satisfying axioms 5, 9, 7, 10, 11 and cardinal inter-personally comparable (definition 29) is a leximin social welfare functional.

3.1.5 Analytical Formulation of Leximin

In this subsection, an analytical approach to the leximin social welfare functional (Yager, 1997) is presented. Rather than implementing the leximin social welfare function as one of pairwise comparisons, Yager's defines an ordered weighted average (OWA), $F_{leximin}$:

such that xRy if and only if $F_{leximin}(x) \geq F_{leximin}(y)$, xPy if and only if $F_{leximin}(x) > F_{leximin}(y)$, and xIy if and only if $F_{leximin}(x) = F_{leximin}(y)$. The analytical leximin OWA allows for the specification of a trusted PST maximisation problem where the PST that maximises $F_{leximin}$ is the most trustworthy.

Definition 31 (Ordered Weighted Average (OWA))**.** An ordered weighted average operator F of dimension n is a mapping $F : \mathbb{R}^n \to \mathbb{R}$ that has an associated vector of weights $W = [w_1, w_2, \ldots, w_n]$ such that $\sum_{i=1}^{n} w_i = 1$ and each $w_i \in [0, 1]$ and where $F(y_1, y_2, \ldots, y_n) = \sum_{j=1}^{n} w_j \cdot z_j$ where z_j is the j-largest y_i.

Definition 32. (Analytical Leximin Aggregation) The analytical leximin aggregation operator, $F_{leximin}$, is an ordered weighted average where the weight vector $W = [w_1, \ldots, w_{n-2}, w_{n-1}, w_n]$ is defined as follows:

$$w_1 = \frac{\Delta^{n-1}}{(1+\Delta)^{n-1}},$$
$$w_j = \frac{\Delta^{n-j}}{(1+\Delta)^{n+1-j}} \text{ for all } 2 \leq j \leq n.$$

If $|a - b| < \Delta$ then $a = b$. If $a > b$ then $|a - b| > \Delta$.

Theorem 6 (The Analytical Leximin Aggregation is a Leximin SWFL)**.** For all $x, y \in C$, $xP_{leximin}y \iff F_{leximin}(u_1(x), u_2(x), \ldots, u_n(x)) > F_{leximin}(u_1(y), u_2(y), \ldots, u_n(y))$ and $xI_{leximin}y \iff F_{leximin}(u_1(x), u_2(x), \ldots, u_n(x)) = F_{leximin}(u_1(y), u_2(y), \ldots, u_n(y))$ where $u_i(x) \in \mathbb{R}$, $u_i(x)$ is the utility of individual i for the alternative x, $xP_{leximin}y$ implies that x is preferred to y by a leximin social welfare functional and $xI_{leximin}y$ implies that x is indifferent to y by the same leximin social welfare functional.

Yager does not explicitly discuss inter-personal comparability, but it is clear that it applies to the analytical leximin aggregation. Definition 24 implies that $\forall U, U' \in \bar{L} : F(U) = F(U')$, that is $\forall x, y \in C : xRy \iff \forall (u_1, u_2, \ldots, u_n) \in \bar{L} : F((u_1(x), u_2(x), \ldots, u_n(x))) \geq F((u_1(y), u_2(y), \ldots, u_n(y)))$ and from this xPy and xIy can be defined as in definition 14 (Sen, 1970). For Yager's analytical leximin function, the following invariance must hold true given theorems 6 and 24, $xR_{leximin}y \iff \forall (u_1, u_2, \ldots, u_n) \in \bar{L} : F_{leximin}((u_1(x), u_2(x), \ldots, u_n(x))) \geq F_{leximin}((u_1(y), u_2(y), \ldots, u_n(y)))$. However, all elements of $\bar{L}$ can not be cardinal non-comparable or cardinal unit-comparable given theorem 5 that states that leximin SWFLs must be cardinal fully comparable, as it would be a contradiction to the invariance requirement.

3.2 Trust and Publish/Subscribe Trees

3.2.1 Definition of Trust

Trust is "the firm belief in the competence of an entity to act dependably, securely and reliably within a specified context" and distrust is "the lack of firm belief in the competence of an entity to act dependably, securely and reliably within a specified context" (Grandison and Sloman, 2000). Typically, the terms trust and reputation have been considered to be interchangeable, particularly in the areas of peer-to-peer and mobile ad hoc network routing. For some applications, this may be a justifiable assumption, as the only trust source is past behaviour, however other trust sources such as a priori knowledge and knowledge external to the system may be of value in making a trust judgement. In the context of publish/subscribe, behavioural history is of too coarse a granularity to be evaluated as a single property due to an entity's ability to assume multiple roles within the system. For example, a node may be a publisher in one tree and a subscriber in another with differing competences of each role. A trust evaluation function may benefit by considering the role under which past behaviours occurred.

The sources to determine trust are varied, with their importance to trust evaluation dependent on the trustor, who may consider some properties to be of more significance than others depending on the context. Vector-based trust models have been proposed to represent the properties that determine the competence of an entity within a given context (Ray and Chakraborty, 2004), a generalisation of which is provided below. Note that no assumptions are made as to either the properties to be considered or the context, as it is not of any significance to the PST problems addressed in this thesis and their solutions.

Definition 33 (Trust Vector)**.** A trust vector is a d-dimensional real-valued vector $\Lambda^{\eta}_{i,j} = [\lambda^{\eta}_{i,j_1}, \lambda^{\eta}_{i,j_2}, \ldots, \lambda^{\eta}_{i,j_d}]$ such that for each λ^{η}_{i,j_n} is a real value, each representing a different property of trust, such as reputation, within some context η. $\Lambda^{\eta}_{i,j}$ is the trust vector representing i's trust opinion of j within some context η.

Definition 34 (Individual Trust Function)**.** For each individual $i \in N$, i has a trust function $\tau_i : \mathbb{R}^d \to \mathbb{R}$ which is a mapping of trust vectors to trust values. Given a pair of individuals i and j, a trust vector $\Lambda^{\eta}_{i,j}$, $\tau_i(\Lambda^{\eta}_{i,j})$ is a real value representing i's trust in j within the context η.

Axiom 12 (Consistency of Individual Trust Functions)**.** For each $i, j, k \in N$, if i has trust vectors $\Lambda^{\eta}_{i,j}$ and $\Lambda^{\eta}_{i,k}$ such that $\Lambda^{\eta}_{i,j} = \Lambda^{\eta}_{i,k}$, then $\tau_i(\Lambda^{\eta}_{i,j}) = \tau_i(\Lambda^{\eta}_{i,k})$. If this is untrue

for some node i, then it is bias and can be determined to be malicious. No external information other than that from the trust vector should be considered.

Definition 35 (Trust Ordering)**.** For each $i, j, k \in N$, a trust preference ordering can be defined as follows:

1. $\tau_i(\Lambda^{\eta}_{i,j}) \geq \tau_i(\Lambda^{\eta}_{i,k}) \implies jR_ik$;
2. $\tau_i(\Lambda^{\eta}_{i,j}) > \tau_i(\Lambda^{\eta}_{i,k}) \implies jP_ik$;
3. $\tau_i(\Lambda^{\eta}_{i,j}) = \tau_i(\Lambda^{\eta}_{i,k}) \implies jI_ik$.

As per Arrow's definition of preference given in definition 14, R is complete and transitive.

3.2.2 Semiring-based Trust

Theodorakopoulos and Baras propose a semiring-based trust framework that allows users to determine the trustworthiness of others, even in the absence of first-hand trust information (Theodorakopoulos and Baras, 2006). Given two nodes, i and j, the i's trust in j is determined by the aggregation of the trustworthiness of the nodes on some path or set of paths between the two nodes. Their proposed model can be used to express the a variety of trust models, such as Eigentrust (Kamvar et al., 2003) and the PGP web of trust (Zimmermann, 1995).

Semirings can not only be used as a framework for trust models, but can also be used to determine the trustworthiness of paths in some graph, in this case a connectivity graph. The path trust semiring presented in definition 37 is a generalisation of the path trust rating used in SPROUT (Marti et al., 2005) and allows an individual i to determine the trust of the path to some individual j. This differs to the Theodorakopoulos and Baras framework that gives the trust of the individual.

Definition 36 (Semiring)**.** A semiring $(S, \oplus, \otimes)$ is a set, S, with two binary operators, $\oplus$ and $\otimes$ that meets the following axioms:

- $(S, \oplus)$ is commutative semigroup with neutral element $\mathbf{0}$:

$$a \oplus b = b \oplus a$$

$$(a \oplus b) \oplus c = a \oplus (b \oplus c)$$

$$a \oplus \mathbf{0} = a$$

- $(S, \otimes)$ is a semigroup with a neutral element $\mathbf{1}$ and an absorbing element $\mathbf{0}$:

$$(a \otimes b) \otimes c = a \otimes (b \otimes c)$$

$$a \otimes \mathbf{1} = \mathbf{1} \otimes a = a$$

$$a \otimes \mathbf{0} = \mathbf{0} \otimes a = \mathbf{0}$$

- $\otimes$ is distributive $\oplus$:

$$(a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c)$$

$$a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$$

A semiring $(S, \oplus, \otimes)$ with a partial order that is monotone with respect to both operators is called an ordered semiring $(S, \oplus, \otimes, \leq)$:

$$a \leq a' \wedge b \leq b' \Rightarrow a \oplus b \leq a' \oplus b' \wedge a \otimes b \leq a' \otimes b'$$

An ordered semiring $(S, \oplus, \otimes, \leq)$ is ordered by the difference relation, or naturally ordered, if,

$$\forall a, b \in S \,.\, (a \leq b) \Leftrightarrow \exists z \in S . a \oplus z = b$$

A semiring $(S, \oplus, \otimes)$ is idempotent if the idempotent law holds for $\oplus$:

$$a \oplus a = a$$

For idempotent semirings, the relation defined by:

$$a \leq b \Leftrightarrow a \oplus b = b$$

is a partial order.

Definition 37 (Path Trust Semiring)**.** The trusted path semiring is a semiring, $(S, \oplus, \otimes)$ where $S = [0, 1]$ and $\oplus$ and $\otimes$ are defined as:

$$\text{for all } s_1, s_2 \in S,\ s_1 \oplus s_2 = max(s_1, s_2)$$

$$\text{for all } s_1, s_2 \in S,\ s_1 \otimes s_2 = s_1 s_2$$

Example 1 (Example Use of Trusted Path Semiring)**.** Assume the presence of a simple path $\sigma_1 = (e_1, e_2, ..., e_n)$ where n is the number of edges, and the vertices on the path are given by $\delta(e_1) = \{v_1, v_2\}, \delta(e_2) = \{v_2, v_3\}, ..., \delta(e_n) = \{v_n, v_{n+1}\}$. Let vertex v_1 and v_{n+1} be the initial and final vertex respectively. Let the trustworthiness of the path σ_1 be τ_{σ_1} and is determined by the initial vertex to be $\tau(v_1, v_2) \otimes \tau(v_1, v_3) \otimes ... \otimes \tau(v_1, v_{n+1})$ where $\tau : V \times V \to S$ and gives the trust that one vertex has in another, represented by values from the set S of the semiring. Additionally, given p alternative simple paths from v_0 to v_{n+1}, the most trusted one is given by $\tau_{\sigma_1} \oplus \tau_{\sigma_2} \oplus ... \oplus \tau_{\sigma_p} = max(\tau_{\sigma_1}, \tau_{\sigma_2}, ..., \tau_{\sigma_p})$.

Example 1 demonstrates how the path trust semiring given in definition 37 can be used to rate the trustworthiness of a path. As stated above, it is a semiring representation of the reliability path rating metric defined by Marti et al. (Marti et al., 2005). The metric can be interpreted as the likelihood that a given message is correctly routed to the destination, as the trust of each node is the probability that the node will execute the routing protocol as defined. In the context of publish/subscribe, this translates to the path trust rating being the probability that the nodes on the path correctly execute the content-based routing and matching algorithms.

3.2.3 Trust Relationships in Publish/Subscribe Trees

The trust definitions provided allow for an entity to evaluate the trustworthiness of others, but it does not define how an entity can determine the trustworthiness of a publish/subscribe tree. To do this, first it is important to understand the trust relationships between nodes in a PST. These relationships can then be evaluated using the individual trust functions and aggregated to give a trust value for a PST. Using the confidentiality and role definitions, the following trust relationships can be identified:

1. Does the publisher trust the subscribers to receive its publications? (from Modified Publisher Confidentiality)

2. Does the publisher trust the internal nodes (be it subscribers or routers) with its publications? (from Modified Information Confidentiality)

3. Do the subscribers trust the internal nodes with their subscription functions so that content-based matching and routing can take place (from Modified Subscription Confidentiality)

4. Do the subscribers trust the publisher to publish correct and timely publications?

5. Do the subscribers, both leaf and internal, trust upstream nodes to route events to it correctly?

6. Do the internal nodes, excluding subscribers, trust the publisher, so as to want to route its events?

7. Do the internal nodes trust the downstream internal nodes on the PST to correctly route events that they propagate?

8. Do the internal nodes trust the upstream internal nodes to route events correctly from the publisher?

9. Do the internal nodes trust the downstream subscribers so as to want to route events to them?

3.2.4 Trust Evaluation Functions for PSTs

From the list of relationships given in section 3.2.3, trust evaluation is dependent on the role of the node and its depth in the PST. The definitions below show how the trust of a PST is evaluated for the publisher, the internal subscribers and the terminal subscribers.

The publisher has a contract to deliver events to each subscriber in the publish/ subscribe tree, so it is logical to conclude that it wishes to maximise the trustworthiness of each path to the subscribers. The publisher may reject a subscriber's attempt to join the tree, that is implement some kind of trust-based access control. The publisher trust function gives the publisher's trust value of a PST by aggregating the trust of each path to each subscriber in the PST. Note that paths to subscribers may share sub-sequences of vertex and edge sequences, but irrespective of this, each path is evaluated. Additionally, where the publisher is neighbouring the subscriber, the trust value of the path is 1, as the subscriber has been trusted to receive events and there is no intermediate path that must be considered in the evaluation of the path.

Definition 38 (Publisher PST Trust)**.** Let $T = (V, E)$ be a PST, where $V = S \cup R \cup \{p\}$ for a publisher p, set of subscribers S and set of routers R, and let α be some aggregation function, $\alpha : \mathbb{R}^{|S|} \longrightarrow \mathbb{R}$. For each $s \in S$, there is a path $\sigma_{p,s} = \{p, \ldots, s\}$, a vertex sequence with initial vertex p, final vertex s and if $|\sigma_{p,s}| > 2$, it has intermediate vertices $\{v_1, v_2, \ldots, v_{|\sigma_{p,s}|-2}\}$, and whose trustworthiness is given by:

$$\tau_p(\sigma_{p,s}) = \begin{cases} 1 & \text{if } |\sigma_{p,s}| = 2 \\ \tau_p(\Lambda^{\eta}_{p,v_1}) \otimes \tau_p(\Lambda^{\eta}_{p,v_2}) \otimes \cdots \otimes \tau_p(\Lambda^{\eta}_{p,v_{|\sigma|-2}}) \otimes \tau_p(\Lambda^{\eta}_{p,v_{|\sigma|-1}}) & \text{if } |\sigma_{p,s}| > 2 \end{cases}$$

The trust of T for p is a function of the trust of the paths to each subscriber and is given by $\tau_p(T) = \alpha(\tau_p(\sigma_{p,s_1}), \tau_p(\sigma_{p,s_2}), \ldots, \tau_p(\sigma_{p,s_{|S|}}))$.

Terminal nodes, always subscribers given the PST definition, receive events along a path sourced at the publisher and with at least one node on the path being aware of its subscription. Their trust in the PST is determined exclusively by the trust of this path.

Again, as with the publisher, if the subscriber is adjacent to the publisher, its trust value of the path to the root is given as 1 because it has expressed a willingness to receive notifications from the publisher by joining the PST.

Definition 39 (Terminal Subscriber PST Trust)**.** Let $T = (V, E)$ be a PST, where $V = S \cup R \cup \{p\}$ for a publisher p, set of subscribers S and set of routers R. For each subscriber $s \in S$ such that s is a terminal of T and $\sigma_{s,p} = \{s, \ldots, p\}$ is a path in T with initial vertex s to terminal vertex p and if $|\sigma_{s,p}| > 2$ with intermediate vertices $\{v_1, v_2, \ldots, v_{|\sigma_{s,p}|-2}\}$, then the trust of s in T is given by:

$$\tau_s(T) = \begin{cases} 1 & \text{if } |\sigma_{s,p}| = 2 \\ \tau_s(\Lambda^{\eta}_{s,v_1}) \otimes \tau_s(\Lambda^{\eta}_{s,v_2}) \otimes \cdots \otimes \tau_s(\Lambda^{\eta}_{s,v_{|\sigma_{s,p}|-2}}) \otimes \tau_s(\Lambda^{\eta}_{s,v_{|\sigma_{s,p}|-1}}) & \text{if } |\sigma_{s,p}| > 2 \end{cases}$$

For each internal subscriber node, not only does it receive notifications on the path to the publisher, but it also propagates the notification on the subtree of which it is the root, but only if the notification matches its proxied subscription. An internal subscriber can be considered as simultaneously sharing the roles of terminal subscriber and publisher, so it also shares their associated trust relationships. An internal subscriber's trust of a PST is a function of the trust in the path to the publisher and the trustworthiness of the paths to the subscribers in its subtree.

Definition 40 (Internal Subscriber PST Trust)**.** Let $T = (V, E)$ be a PST, where $V = S \cup R \cup \{p\}$ for a publisher p, set of subscribers S and set of routers R. For each subscriber $s \in S$ such that s is an internal node, there is a path $\sigma_{s,p} = s, \ldots, p$ where s is the initial vertex, p is the final vertex and with intermediate vertices $\{v_1, v_2, \ldots, v_{|\sigma_{s,p}|-2}\}$ if $|\sigma_{s,p}| > 2$. The trust of the $\sigma_{s,p}$ is given by:

$$\tau_s(\sigma_{s,p}) = \begin{cases} 1 & \text{if } |\sigma_{s,p}| = 2 \\ \tau_s(\Lambda^{\eta}_{s,v_1}) \otimes \tau_s(\Lambda^{\eta}_{s,v_2}) \otimes \cdots \otimes \tau_s(\Lambda^{\eta}_{s,v_{|\sigma|-2}}) \otimes \tau_s(\Lambda^{\eta}_{s,v_{|\sigma|-1}}) & \text{if } |\sigma_{s,p}| > 2 \end{cases}$$

Additionally, for each $s \in S$ such that s is an internal node, let $T_s = (V_s, E_s)$ be the subtree rooted at s. For each $s' \in (S \setminus s) \cap V_s$, there is a path $\sigma_{s,s'} = \{s, \ldots, s'\}$ that has initial vertex s, final vertex s', and intermediate vertices $\{v_1, v_2, \ldots, v_{|\sigma_{s,s'}|-2}\}$. The trust of the path $\sigma_{s,s'}$ is given by:

$$\tau_s(\sigma_{s,s'}) = \begin{cases} \tau_s(\Lambda^{\eta}_{s,s'}) & \text{if } |\sigma_{s,s'}| = 2 \\ \tau_s(\Lambda^{\eta}_{s,v_1}) \otimes \tau_s(\Lambda^{\eta}_{s,v_2}) \otimes \cdots \otimes \tau_s(\Lambda^{\eta}_{s,v_{|\sigma|-1}}) \otimes \tau_s(\Lambda^{\eta}_{s,s'}) & \text{if } |\sigma_{s,s'}| > 2 \end{cases}$$

For each internal subscribe node s in a PST T, the trust of s in T is given by $\tau_s(T) = \beta(\tau_s(\sigma_{s,p}), \tau_s(\sigma_{s,s'_1}), \ldots, \tau_s(\sigma_{s,s'_{d-1}}))$ where $\beta : \mathbb{R}^d \longrightarrow \mathbb{R}$ is some aggregation function of trust values, and $d = |V_s \cap S| + 1$.

In the implementation of this model, the aggregation functions α and β in definitions 38 and 40 are the leximin aggregation function given in definition 32 and the minimum aggregation function. If d were not variable across PSTs, then β would also be given by definition 32. Although other aggregation functions such as arithmetic mean could have been used, this would not have been fitting with Rawls' difference principle.

Consider two PSTs $t_1 = (V_{t_1}, E_{t_1})$ and $t_2 = (V_{t_2}, E_{t_2})$, and an internal subscriber $s \in V_{t_1}, V_{t_2}$, such that $\tau_s(T_1) = \beta(0.7, 0.8, 0.9, 0.1)$ and $\tau_s(T_2) = \beta(0.5, 0.5, 0.5, 0.5)$. If β is the arithmetic mean then $t_1 P t_2$, however, if β is the minimum aggregation function then $t_2 P t_1$. It can be argued that $t_2 P t_1$ is a more appropriate trust preference for s because of two reasons, the first fairness and the second, the increased likelihood of damage to the reputation of s if the path with trust value 0.1 has a subscriber as its final vertex.

Assuming that in t_1, $\tau_s(\sigma_{s,s'}) = 0.1$ and in t_2, $\tau_s(\sigma_{s,s'}) = 0.5$, then it is evident that s' is better off under in t_2, as the path to it from s is of a higher trust value. t_2 is a fairer choice and benefits the least well-off, but this comes at a cost to the trust of other paths. Additionally, in this scenario, choosing t_1 over t_2 is not wise, as even through the other paths are of very high trust value, the path of trust value 0.1 is liable to loss or corruption of events that are re-broadcast to s' and consequently resulting in lower reputation feedback of s from s'.

Having defined the individual trust functions for PSTs, the only remaining issue is how to aggregate these to give a trust value for PSTs. As discussed in section 3.1.1, an egalitarian approach is rejected and the reasons for this have been illustrated above. The social ordering of PSTs must improve the well-being of the least well-off with respect to trust, so it follows that the leximin social welfare functional is used, since it is assumed that all nodes are to be treated equally and there is consequently an absence of policies defining a subset of preferred nodes. However, a quantitative measure is required because the problem to be solved is one of finding the PST that maximises the trustworthiness of the PST used for a given advertisement, so Yager's analytical leximin function given

in definition 32 must be used instead. Unfortunately, its use is not without issue, as it requires cardinal full comparability and it is, at the least, questionable if trust functions comply with this property (see section 3.2.5).

Definition 41 (Socially Trusted PST Aggregation)**.** Let $t = (V_t, E_t)$ be a PST where $V_t = S \cup R \cup \{p\}$. For each $i \in S \cup \{p\}$, there is a real-value $\tau_i(T)$ representing i's trust value of t. The social trust value of t is given by $F_{leximin}(\tau_{i_1}(T), \tau_{i_2}(T), \ldots, \tau_{i_{|S\cup\{p\}|}}(T))$.

3.2.5 Trust and Interpersonal Comparability

There has been much discussion in fields of social sciences and computer science regarding trust, with a variety of trust definitions, trust evaluation functions and trust metrics proposed. In section 3.1, a brief overview of social choice theory is given and includes a discussion of interpersonal comparability. Here, this theory is used to show that individual trust evaluation functions are interpersonally non-comparable when used to determine preference over a set of possible PSTs, and can only be interpersonally comparable under strict conditions.

Assuming that trust is measured as a real value, for trust evaluation functions to meet cardinal full comparability, as required by the leximin social welfare functional, there must be a one-to-one correspondence between individuals' trust evaluation functions. To assume this is not logical due to the heterogeneous nature of the individuals whose understanding of trust and the manner in which trust sources (e.g. reputation from behavioural history) are evaluated can not only differ given the context, but also given the personal characteristics of the individual, which can evolve with interactions over time.

Many different trust metrics have been proposed, with some trust management systems using discrete trust values, and others using real-valued representations of trust. Metrics classified as the latter are typically represented by the unit interval or [-1, 1] with 0 as the origin indicating indifference and/or uncertainty, -1 representing absolute distrust and 1 representing absolute trust. It is assumed that trust can be expressed using real values, but irrespective of the metric chosen, if two individuals i and j have trust opinions 0.5 and 0.75 of two PSTs t_1 and t_2 respectively, is t_2 more trustworthy than t_1? It seems obvious to assume $0.75 > 0.5 \implies t_2 P t_1$ for i and j, but this ignores how the two trust values have been derived. This is a problem of interpersonal comparability. There is also the issue of interpersonal comparability of differences or comparison of marginal utility. If $\tau_i(T_2) - \tau_i(T_1) > \tau_j(T_1) - \tau_j(T_2)$, is the gain in trust for i when the PST T_2 is chosen instead of T_1 greater than the loss (or gain) for j? Due to the complex nature of trust,

there are arguments against this.

Definition 41 gives the socially trusted PST aggregation, which uses the analytical leximin aggregation function (definition 32) to define an ordering over a set of PSTs with respect to each individual's trust evaluation of the PSTs. As the aggregation function implements a leximin social welfare functional using cardinal individual PST trust functions, these must be inter-personally comparable given theorem 5. The inter-personal comparability of trust requires that a unit of trust exists and that its definition is shared by all individuals, and that the origin of the trust continuum used to measure trust at each individual is identical. Under these criteria, absolute trust values between individuals can be compared, however it is unrealistic to assume these to be true for trust. An individual may determine that the values -1, 0 and 1 represent absolute distrust, indifference or ignorance, and absolute trust, while another may assume that trust is represented by the unit interval where 0 represents absolute trust and ignorance and 1 represents absolute trust. Even if some continuum can be defined such that the definitions of the extremes and the origin are identical and it can be imposed upon all individuals, there is no notion of a unit of trust.

Trust can be formed using a number of trust sources, but typically in the literature, it is a function of reputation, which is in turn a function of past interactions. In a given application context, such as a P2P file sharing system, users may have different perceptions of identical behaviour. Some may tolerate corrupted file downloads more than others, and in this scenario may rate identical transactions differently. For example, in Eigentrust (Kamvar et al., 2003), a given user i downloads a corrupted file from a user k and rates the transaction as -1, but a user j may download the same corrupted file from k and rate the transaction 0, perhaps due to having a higher tolerance of such malicious behaviour or citing communication errors as the reason for the transaction failure. In this scenario, i and j interpret the trust sources differently, j is more tolerant of malicious behaviour, perhaps because the file has less value to it or perhaps because of its characteristics. Assume that τ_i and τ_j are identical and that some origin on the trust continuum exists, if i and j both hold a trust value of 0.7 in some entity k, the meaning of that value differs between the two, as their interpretation of the trust sources that is their individual trust functions are different. Given two PSTs T_1 and T_2, it is not possible to state that $\tau_i(T_1) - \tau_i(T_2) > \tau_j(T_2) - \tau_j(T_1)$ is true or false, that is the differences in the trust of two PSTs are not comparable across individuals. It is postulated that trust is inter-personally incomparable.

The trust mechanisms described above provide a social preference ordering of the feasible PSTs such that the socially preferred choice maximises the trust of the node with the lowest trust evaluation of the trees, that is the least well off as implied by the leximin social welfare functional. Mapping individuals' cardinal trust evaluation functions to a social preference ordering using the leximin social welfare function requires that the trust evaluation functions are interpersonally comparable, but arguments have been presented that contradict this. The proposed solution to this issue is to assume that each individual's trust evaluation function is identical, along with all other auxiliary functions required, such as reputation functions (definition 42). This assumption appears to be overly strict and unrepresentative of the nature of trust. Nodes are considered to homogeneous with respect to trust evaluation, valuing properties and behaviours of others identically. Trust values between pairs of nodes would only differ because of different instances of trust sources used as input to the trust function. Some may consider the following assumptions to be unreasonable, however they are no worse than the ignoring the issue of interpersonal comparability of trust, which is currently the case in trust-related literature. A greater understanding of how trust is derived from the perspective of social sciences and a formalisation of this are both beyond the scope of this thesis, but certainly warrants further academic research.

Definition 42 (Inter-personal Comparable Trust Assumption)**.** $\forall i, j \in S \cup \{p\} : \forall T \in \mathcal{T} : \tau_i(T) = \tau_j(T)$ where $\mathcal{T}$ is the set of PSTs for some connectivity graph G. It must also be assumed that for all $i, j \in S \cup \{p\}$ and for all $k \in V$, if $\Lambda^{\eta}_{i,k} = \Lambda^{\eta}_{j,k}$, then $\tau_i(\Lambda^{\eta}_{i,j}) = \tau_j(\Lambda^{\eta}_{i,k})$.

3.3 Summary

The first part of this chapter provides an introduction to social welfare and choice theory, with the emphasis on the following areas: social welfare functions; social welfare functionals; the leximin social welfare functional; and inter-personal comparability. The introduction provides only the necessary prerequisites for an understanding of the motivations of the trust functions that are given later in the chapter.

In the remainder of the chapter, the trust functions for PSTs are presented and the issue of inter-personal comparability of trust is addressed. The flow of communications in the PST is used to define PST trust evaluation functions for the publisher, internal subscribers and terminal subscribers. The individual trust functions give the trustworthiness of the PST as function of some subset of paths in the PST, which is dependent on its role and height in the tree. Using the analytical leximin function, a leximin ordering over a set of

PSTs can be derived by aggregating each individual's evaluation. As the approach uses the leximin social welfare function, cardinal full inter-personal comparability is required, but it is argued that such an assumption for trust is unrealistic, so simplifying assumptions are proposed to address this. This is an issue that is widely disregarded in the trust literature.

Chapter 4

Minimum Overhead-Maximum Trust PST Problem

In this chapter, the PST trust maximisation problem with overhead budget is defined and algorithms are presented to solve the problem. As the problem is shown to be in NP-Complete, two algorithms are proposed to solve it. An exhaustive search algorithm enumerates all feasible PSTs and selects the one that maximises trust within some overhead budget. As the problem is in NP-Complete, evidently the algorithm can not be expected to scale to large problem instances, so a tabu search algorithm is presented that approximates the solution. Both algorithms are dependent upon a modification to Char's spanning tree enumeration algorithm to find all PSTs in a graph, which is also described in this chapter.

4.1 Problem Definition

Problem 2 (The PST Trust Maximisation Problem with Overhead Budget (MTPSTO))**.** Given an overhead budget $B > 0$, an event distribution E, an undirected connectivity graph $G_c = (V_c, E_c)$, a publisher p that holds an advertisement A_p, a set of subscribers $S = \{s \mid sf_s(A_p) = true\}$ where sf_s is the subscription function of s, a set of routers $R = V_c \setminus C$ where $C = \{p\} \cup S$, find a PST T that is rooted at p, spans C and maximises the trust value $\tau(T) = F_{leximin}(\tau_{c_1}(T), \ldots, \tau_{c_{|C|}}(T))$ where $\tau_{c_i}(T)$ is the trust evaluation of i^{th} node in C, subject to $O_T(Ev) \leq B$.

Problem 3 (The PST Trust Maximisation Problem with Overhead Budget (MTPSTO) - Decision Problem)**.** Given an overhead budget $B > 0$, $q \in [0, 1]$ is a trust quota in the unit interval, an event distribution E, an undirected connectivity graph $G = (V, E)$, a publisher p that holds an advertisement A_p, a set of subscribers $S = \{s \mid sf_s(A_p) = true\}$

where sf_s is the subscription function of s, a set of routers $R = V_c \setminus C$ where $C = \{p\} \cup S$, is there a PST T that is rooted at p and spans C such that $\tau(T) \geq q \wedge O_T(E) \leq B$.

Problem 2 is a PST trust maximisation problem within a given overhead budget, B. This approach is preferred to a multi-objective problem as this allows for the budget to be tunable to suit the application requirements. For example, if performance is of great importance then a strict overhead budget is required. This will restrict the search space of the feasible PSTs to those that provide low communication overheads, but possibly at the expense of the trustworthiness of the PST that solves the problem. Applications that are suited to this model are real-time data distribution applications, such as radar data distribution, and media distribution where the QoS required is dependent upon the properties of the media.

Theorem 7 (MTPSTO is NP-complete)**.** The PST Trust Maximisation Problem with Overhead Budget is NP-complete.

Proof of Theorem 7. To show that MTPSTO $\in$ NP, for a given PST $T_{PST} = (V_{PST}, E_{PST})$, the overhead of the tree $O_{T_{PST}}(E)$ with respect to some event distribution E and the trust value $\tau(T_{PST})$ must be established in polynomial time. The algorithm to calculate $O_{T_{PST}}(E)$ for T_{PST} must start at the publisher vertex of the PST and perform a post-order tree traversal of the tree, so that the $O_{t_i}(E) = (r+f)\cdot\Phi E(\neg s_i \wedge s_i^{'}) + f\cdot\Phi E(s_i \wedge s_i^{'})$ is calculated for each node $i \in V_{PST}$. At each i, every event $e \in E$ must be tested to establish if $s_i(e) = true$, so this algorithm has $\mathcal{O}(|V_{PST}||E|)$ complexity. This part of the proof is similar to the MOPST $\in$ NP proof by (Cao and Shen, 2009).

To complete this first part of the proof, it must be shown that the trust value of the PST can be calculated in polynomial time. The analytical leximin aggregation function $F_{leximin}$, given in definition 32, which is used to aggregate the individual trust evaluations of a PST must sort the set of trust values to be aggregated, which can be performed by a tuned quicksort algorithm of $\mathcal{O}(c \cdot log(c))$ (Bentley and McIlroy, 1993) and then calculate the OWA which has a total cost of $2c$ as there are two constant time operations, the multiplication of a trust value with its weight and the summation, so the running time of $F_{leximin}$ is $\mathcal{O}(c+c\cdot log(c))$ where $c = |C|$. Next, the complexity of establishing the individual trust evaluations that are the inputs to $F_{leximin}$ must be determined. Assume that a data structure containing for all $a, b \in V_c$, $\tau_a(\Lambda^{\eta}_{a,b})$. Given a pair of nodes $a, b \in V_{PST}$, the lookup time for $\tau_a(\Lambda^{\eta}_{a,b})$ is dependent on the data structure used and is assumed to be polynomial time, α. Each of the following roles, the publisher, the internal subscribers and the terminal subscribers are considered. For each terminal subscriber s, the total cost of

calculating $\tau_s(\sigma_{s,p})$ where $\sigma_{s,p}$ is the path from s to p in T_{PST} is $y \cdot \alpha$ where $y = |\sigma_{s,p}| - 2$. For the publisher p in T_{PST}, a pre-order tree traversal algorithm is used to find all the paths to all $s \in V_{PST} \cap S$ where at every s the trust value $\tau_p(\sigma_{p,s})$ is calculated, so the time complexity of this algorithm is $\mathcal{O}(n + \alpha \cdot w)$ where $n = |V_{PST}|$ and $w = |V_{PST} \cap S|$. The internal subscriber must calculate the trust to the publisher and all subscribers in the subtree rooted at the internal subscriber, so it must execute both algorithms, giving $\mathcal{O}(n + \alpha(w + y))$. A PST can be verified in polynomial time, so MTPSTO $\in$ NP.

To prove that MTPSTO is NP-hard, it must be shown that there is a polynomial time reduction from MOPST, that is MOPST $\leq_P$ MTPSTO. The MOPST problem is given in problem 1. A polynomial-time reduction from any instance of MOPST to MTPSTO can be performed, as follows:

1. $\forall a, b \in V_{PST}$, $\tau_a(\Lambda^{\eta}_{a,b}) = 1 \iff \tau(T) = 1$.
2. The decision version of the MOPST problem returns true if $O_t(E) \leq k$, so let $B = k$ in MTPSTO.
3. Let $q = 1$ in the decision problem version of MTPSTO.

Given an instance of MOPST such that $O_{T_{PST}}(E) \leq k$, after reduction of this instance to an instance of MTPSTO using the rules presented above, assume that MTPSTO returns false, that is either $O_{T_{PST}}(E) > B$ or $\tau(T_{PST}) < q$. $\tau(T_{PST}) = 1$ is true for every PST found by MTPSTO given the definition of the analytical leximin function in definition 32 and the fire rule of the reduction, so $O_{T_{PST}}(E) > B$ must hold. However, $B = k$, so it follows that $O_{T_{PST}}(E) > B$ and $O_{T_{PST}}(E) \leq k$ is a contradiction.

Given an instance of MTPSTO such that there is a PST T_{PST} where $\tau(T_{PST}) \geq q \wedge O_{T_{PST}}(E) \leq B$, and assume that in MOPST $O_{T_{PST}}(Ev) > k$. Let $k = B$ as defined the polynomial-time reduction, substituting B for k, $O_{T_{PST}}(E) > k \wedge O_{T_{PST}}(E) \leq k$ is a contradiction. □

4.2 An Exhaustive Search Algorithm for MTPSTO

To solve the MTPSTO problem, an exhaustive search algorithm of all possible PSTs is presented. The algorithm must calculate the trust value and the overhead value of every PST in the connectivity graph $G_c = (V_c, E_c)$ that is rooted at the publisher p and spans all subscribers S for a given advertisement A_p. The set of all PSTs for A_p is a subset of the set of all Steiner trees in G_c. Using this property and the fact that the set of all Steiner

trees is given by the enumeration of all spanning trees for G_c and all its subgraphs, an exhaustive search algorithm must find all the spanning trees of G_c and all its subgraphs that are eligible PSTs, and calculate the trust and overhead values of each. As the set $C = \{p\} \cup S$ must be present in every eligible PST for the advertisement A_p, the considered subgraphs of G must also contain these vertices. Also note that subgraphs with router vertices with only one adjacent edge are not examined as all spanning trees found will not be PSTs as this router vertex will be a terminal router vertex in every PST and this contradicts the definition of a PST.

Algorithm 1 presents an exhaustive search to find the PST that solves the MTPSTO problem. At line 2, all subgraphs containing the publisher and the set of subscribers are found. At line 6, for each subgraph, the PSTEnumeration algorithm is executed, which finds all spanning trees in the subgraph that are also PSTs. The overhead values for each PST is calculated at line 9 and if the overhead is less than or equal to the assigned budget, the trust of the PST is evaluated at line 11. Should the PST have the same trust value as the best PST found so far by the algorithm, ties are broken by selecting the PST with the least overhead.

4.3 Spanning Tree Enumeration

A number of algorithms have been proposed to solve the problem of enumerating all spanning trees of a graph. Backtracking-based techniques proposed by (Minty, 1965) and (Gabow and Myers, 1978) have $\mathcal{O}(m + n + mt)$ and $\mathcal{O}(m + n + nt)$ complexity for undirected graphs respectively, where $m = |E|$, $n = |V|$, and t is the number of spanning trees. Prior to the publication of these algorithms, an alternative was proposed (Char, 1968), and although a complexity analysis was not given, it was later shown to be of $\mathcal{O}(m + n + n(t + t_0))$ running time where t_0 is the number of subgraphs found by the algorithm that are not spanning trees (Jayakumar et al., 1984). Char's approach differs to the backtracking-based techniques as it lexicographically tests subgraphs to determine if each is a spanning tree. It is shown to be suitable for enumerating PSTs, as the spanning tree test of a subgraph can be modified to determine if the subgraph is a PST.

4.3.1 Char's Spanning Tree Enumeration Algorithm

Char's algorithm, presented in algorithm 2, begins with the initialising of T_{init}, the initial spanning tree at line 1. The original algorithm, as described by (Jayakumar et al., 1984),

Algorithm 1: MTPSTO-Exhaustive(G_c, p, S, B)

Input : $G_c = (V_c, E_c)$ /* Connectivity Graph */
Input : p /* Publisher */
Input : S /* Subscribers */
Input : B /* Overhead Budget */
Input : E /* Event Set */
Output: T_{A_p} /* PST */

```
1  begin
2      G_c = {(V_x, E_x) | V_x ⊂ V_c ∧ C ⊆ V_x ∧ ∀(a,b) ∈ E_c, (a,b) ∈ E_x ⟹ a,b ∈ V_x}
           /* G_c is the set of all subgraphs of G_c including G_c */
3      T_best ← null
4      O_best ← ∞
5      τ_best ← −∞
6      for (V_x, E_x) ∈ G_c do
7          A ← PSTEnumeration(G)
8          for T_Ap ∈ A do
9              O_TAp ← EvaluateTreeOverhead(T_Ap, E, p)
10             if O_TAp ≤ B then
11                 τ_TAp ← EvaluateTrust(T_Ap, p, S, )
12                 if τ_TAp > τ_best then
13                     τ_best ← τ_TAp
14                     O_best ← O_TAp
15                     T_best ← T_Ap
16                 else if τ_TAp = τ_best ∧ O_TAp < O_best then
17                     τ_best ← τ_TAp
18                     O_best ← O_TAp
19                     T_best ← T_Ap
```

utilises Breadth-first Search (BFS) to determine the initial spanning tree and label the vertices from n to 1 in the order that they are visited. In the algorithm below and in the implementation used in this work, Depth-first Search (DFS) is chosen instead, as it shown to reduce the value of t_0 (Jayakumar et al., 1984).

Having found the initial tree and labeled the vertices, let $\mathsf{REF}(i)$ give the index of the parent of the vertex i. Line 4 initialises the first tree sequence λ_0 to be that found by the BFS or DFS. The sequence DIGIT holds a subgraph of the graph, and is at first set to the initial spanning tree. $\mathsf{DIGIT}(k)$ gives the index of a node that is adjacent to k. The length of DIGIT is $n-1$ and there must be a $\mathsf{DIGIT}(i) = n$ where $1 \leq i \leq n-1$ for the subgraph defined by DIGIT to be a connected graph. Let SUCC be a function such that $\mathtt{SUCC}(\mathsf{DIGIT}(k))$ gives the next vertex in adjacency list of k.

A DIGIT sequence is tested for tree compatibility using the IsTreeSeq procedure. A sequence gives a tree if and only if for each $\mathsf{DIGIT}(i)$ such that $1 \leq i \leq n-1$, there is

Algorithm 2: CharEnumeration(G)

```
   Input  : G = (V, E)                                  /* Connectivity Graph */
   Output: T_span                                /* Set of all spanning trees in G */
 1 begin
 2     T_init = DFS(G)        /* Find initial tree and label vertices */
 3     T_span ← {T_init}
 4     λ_0 ← (REF(1), REF(2), ..., REF(n-1))    /* T_init seq. representation */
 5     DIGIT(i) ← REF(i), 1 ≤ i ≤ n − 1
 6     k ← n − 1
 7     while k ≠ 0 do
 8         if SUCC(DIGIT(k)) ≠ null then
 9             DIGIT(k) ← SUCC(DIGIT(k))
10             if IsTreeSeq(DIGIT) then
11                 T_span ← T_span ∪ {Translate(DIGIT)}
12                 k ← n − 1
13         else
14             DIGIT(k) ← REF(k)
15             k ← k − 1
```

a vertex sequence $(i, \mathsf{DIGIT}(i), \mathsf{DIGIT}(\mathsf{DIGIT}(i)), \ldots, j)$ giving a path from i to j where $j > i$. It is, however, not necessary to test each i for tree compatibility. Given a tree compatible sequence, $(\mathsf{DIGIT}(1), \mathsf{DIGIT}(2), \ldots, \mathsf{DIGIT}(n\text{-}1))$, the next sequence is given by changing $\mathsf{DIGIT}(k)$ to $\mathtt{SUCC}(\mathsf{DIGIT}(k))$, resulting in a sequence $(\mathsf{DIGIT}(1), \mathsf{DIGIT}(2), \ldots, \mathsf{DIGIT}(k\text{-}1), \mathtt{SUCC}(\mathsf{DIGIT}(k)), \mathsf{REF}(k\text{+}1), \ldots, \mathsf{REF}(n\text{-}1))$, so the test for tree compatibility need only take place at position k in the sequence, as all other positions will pass the test.

Lines 7 to 15 present the process of enumerating the remaining spanning trees. The algorithm, starting with the initial spanning tree sequence, generates a series of subgraph sequences. Rather than test each of the sequences from the set of all possible sequences of $n-1$ edges, if for some k, $\mathsf{DIGIT}(k)$ is set to the index of a vertex at line 9 such that there is no path $k, \mathsf{DIGIT}(k), \ldots, n$ then sequences containing the subsequence $(\mathsf{DIGIT}(1), \mathsf{DIGIT}(2), \ldots, \mathsf{DIGIT}(k))$ at positions $1, 2, \ldots, k$ respectively, can not be a tree so these sequences are ignored by the algorithm, reducing the search space.

The running time of Char's algorithm is dominated by $n(t + t_0)$. The number of non-tree compatible subgraphs, t_0 is shown to be dependent on the the initial spanning tree. Jayakumar et al. (Jayakumar et al., 1984) propose DFS search must start at the vertex in the graph with the maximum degree, as $t + t_0$ is shown to be a function of the degree of all $n-1$ vertices. Additionally, the following three properties are identified that may further reduce t_0 (Jayakumar et al., 1984):

1. Maximising the number of leaf nodes of the initial spanning tree found by DFS, as for each leaf node s, there will be no sequence generated by changing $\mathsf{DIGIT}(s)$ that is a non-tree compatible sequence.

2. Maximising the number of ancestors of each vertex in the DFS, as this reduces $t+t_0$.

3. Minimising the number of descendants of each vertex k, as this is the upper bound the number of times the block from lines 8 - 12 is executed for k.

These properties lead to two DFS techniques being proposed by Jayakumar et al. (Jayakumar et al., 1984), both beginning the search at the vertex with the highest degree. The first heuristic selects the next node to visit that maximises the number of ancestors in the tree, and in the event of a tie, chooses the vertex that minimises the degree. The second heuristic visits vertices of minimum degree, deciding ties by choosing the vertex with the most ancestors in the tree. Experimental results show there is little difference between the two techniques with respect to the value of t_0 and both techniques greatly reduce t_0 when compared to BFS. The second heuristic is used for the implementation of DFS in this work.

The second proposed optimisation is path compression (Jayakumar et al., 1984). Assume a tree sequence is found by the algorithm when $k = i$, this gives a tree sequence $\lambda_1 = (\mathsf{DIGIT}(1), \ldots, \mathsf{DIGIT}(i\text{-}1), \mathsf{DIGIT}(i), \mathsf{REF}(i\text{+}1), \ldots, \mathsf{REF}(n\text{-}1))$. Let the next tree λ_2 be found for some $k > i$ such that position λ_2 differs to λ_1 at position k only implying that the first i positions of DIGIT remain the same, as those in λ_1. The tree compatability test for λ_1 tests for a path $i, \mathsf{DIGIT}(i), \ldots, j$ where $j > i$ and this must be true for λ_1 to be a tree. For λ_2, this path is also present in the tree. Assume that the path found by the tree compatability test for λ_2 is $k, \mathsf{DIGIT}(k), \ldots, i, \ldots, j, \ldots, m$ such that $k < m$. It is known that the path from i to j is unique and $j > i$ from the tree compatibility test for λ_1, so if the pair (i, j) is available to tree compatability test for λ_2 the path can be "compressed" thus reducing the complexity of the procedure. The test can terminate successfully if $j \geq m$ or proceed from position j. Algorithm 3 shows the tree compatibility test with path compression, where the $\mathsf{NEXTVERTEX}$ sequence is used to store the compressed path and to perform the test.

Theorem 8 (Leaf Node Property)**.** Let $\lambda = (\mathsf{DIGIT}(1), \mathsf{DIGIT}(2), \ldots, \mathsf{DIGIT}(n\text{-}1))$ be a DIGIT sequence representation of a spanning tree of some graph G and let k be the index of a vertex where $1 \leq k \leq n-1$ such that there is no $\mathsf{DIGIT}(i) = k$ where $1 \leq i \leq n-1$.

Algorithm 3: isTreeSeq(DIGIT, NEXTVERTEX, k)

```
   Input  : DIGIT                                   /* Tree Sequence */
   Input  : NEXTVERTEX                  /* Path Compressed Sequence */
   Input  : k                          /* Position of changed DIGIT(k) */
 1 begin
 2     i ← k
 3     j ← DIGIT(k)
 4     while i ≥ j do
 5         if i = j then
 6             return false
 7         if j < i then
 8             j ← NEXTVERTEX(j)
 9     NEXTVERTEX(k) ← j
10     for m ← k + 1 to Length(DIGIT) do
11         NEXTVERTEX(m) ← REF(m)
12     return true
```

The vertex k is a leaf node in the tree represented by the sequence λ. If $k = n$ and there is only one $\mathsf{DIGIT}(i) = k$ then the vertex n is a leaf node in the tree represented by sequence λ.

Proof of Theorem 8. First, the case where $1 \leq k \leq n-1$ is considered. Let $\lambda = (\mathsf{DIGIT}(1), \mathsf{DIGIT}(2), \ldots, \mathsf{DIGIT}(n\text{-}1))$ be a DIGIT sequence representation of a spanning tree of some graph G and let k be the index of a vertex where $1 \leq k \leq n-1$.

Assume that there is some set I of vertex indices, where:

1. $\forall i \in I . 1 \leq i \leq n-1 \wedge i \neq k$;
2. $\forall i, j \in I . i \neq j$;
3. $|I| \geq 1$;
4. $\forall i \in I . \mathsf{DIGIT}(i) = k$

There must also be a $\mathsf{DIGIT}(k) = m$ where $1 \leq m \leq n$ and $\forall i \in I . m \neq k \neq i$ (i.e. no loop). The values at $\mathsf{DIGIT}(k)$ and $\mathsf{DIGIT}(i)$ for all $i \in I$, give the following edges in the tree represented by λ, $(k, \mathsf{DIGIT}(k))$ and all edges (i, k) for all $i \in I$. The presence of these edges contradicts k being a leaf node, as if this were the case then there would only be one edge adjacent to k, but there are instead $|I| + 1$ adjacent edges. As $1 \leq k \leq n-1$ is true, there will always be one edge $(k, \mathsf{DIGIT}(k))$, so k is a leaf vertex in λ if and only if $|I| = 0$.

Finally, the case where $k = n$ is considered. Again, k is a leaf vertex if and only if k is adjacent to one edge. As $k > n - 1$ there is no value $\mathsf{DIGIT}(k)$ in the sequence λ and consequently no edge $(k, \mathsf{DIGIT}(k))$ in the spanning tree represented by λ, so if k is a leaf vertex, there must only be one edge (i, k) where $1 \leq i \leq n - 1$. Therefore, when $k = n$, k is a leaf vertex in the tree given by λ if and only if $|I| = 1$. □

Algorithm 4: isPST(DIGIT)

```
   Input: DIGIT                                          /* Tree Sequence */
 1 begin
 2     for k ← n to 1 do
 3         if isRouter(k) then
 4             l ← 1
 5             if k = n then
 6                 l ← 2
 7             for i ← n − 1 to 1 do
 8                 if DIGIT(i) = k then
 9                     c ← c + 1
10                     if c = l then
11                         return true
12     return false
```

Theorem 9. Let λ be a DIGIT sequence that is a spanning tree and also a PST of some graph G. For each k, $1 \leq k \leq n - 1$, if k is the index of a router then there is at least one $\mathsf{DIGIT}(i) = k$ and if $k = n$ and k is the index of a router there are at least two $\mathsf{DIGIT}(i) = k$ where $1 \leq i \leq n - 1$ and $i \neq k$.

Proof of Theorem 9. Given the definition of a PST (definition 10, no router node can be a leaf vertex and given theorem 8, for a given leaf vertex k, a tree sequence DIGIT has either no $\mathsf{DIGIT}(i) = k$ if $1 \leq k \leq n-1$ or one $\mathsf{DIGIT}(i) = k$ if $k = n$ where $1 \leq i \leq n-1 \wedge i \neq k$. Therefore, if k is the index of a router and $1 \leq k \leq n - 1$, then there is at least one $\mathsf{DIGIT}(i) = k$ and if $k = n$ then there are at least two $\mathsf{DIGIT}(i) = k$. □

Theorem 9 implies that the algorithm to test if a tree sequence is a PST must check if each router index is not a leaf vertex in the tree represented by the sequence. Algorithm 4 makes use of this property to implement the PST test that must be executed immediately before line 11 in algorithm 2. This gives algorithm 5 to find all spanning trees in a graph that are also PSTs.

Algorithm 5: PSTEnumeration(G)

```
Input  : G = (V, E)                                  /* Connectivity Graph */
Output: T_span                                       /* Set of all spanning trees in G */
1  begin
2      T_init = DFS(G)          /* Find initial tree and label vertices */
3      T_span ← {T_init}
4      λ_0 ← (REF(1), REF(2), ..., REF(n-1))    /* T_init seq. representation */
5      DIGIT(i) ← REF(i), 1 ≤ i ≤ n − 1
6      NEXTVERTEX(i) ← REF(i), 1 ≤ i ≤ n − 1
7      k ← n − 1
8      while k ≠ 0 do
9          if SUCC(DIGIT(k)) ≠ null then
10             DIGIT(k) ← SUCC(DIGIT(k))
11             if IsTreeSeq(DIGIT, NEXTVERTEX, k) ∧ IsPST(DIGIT) then
12                 T_span ← T_span ∪ {Translate(DIGIT)}
13                 k ← n − 1
14         else
15             DIGIT(k) ← REF(k)
16             k ← k − 1
```

4.4 Tabu Search Algorithm for MTPSTO Problem

4.4.1 Tabu Search Preliminaries

All problems in NP-complete can be solved by exhaustive search, but as the size of the problem instance increases, the running times become impractical. Unless $P = NP$ holds true, it is unlikely that there exists a polynomial-time algorithm to solve these and NP-hard problems. For these problems, approximate solutions that are at least close to the optimal can be found within reasonable time bounds by using approximation or metaheuristics algorithms. Approximation algorithms differ to metaheuristics, in that the former guarantees that the solution found is within a factor of the optimal solution for all problem instances and has provable running time bounds (Talbi, 2009). Despite this, metaheuristics have been shown to find good solutions for a variety of optimisation problems, including graph theory problems. One of these metaheuristics for combinatorial optimisation problems is tabu search (Glover, 1989, 1990), which extends local search by marking recent moves as tabu and not to be remade for some number of iterations. Tabu search reduces the likelihood of cycling, by allowing for the search to progress beyond localised areas of the search space. This allows solutions that may be better than the local optimum to be found.

A number of approaches to the Steiner problem in graphs that use the tabu search

metaheuristic have been proposed (Ribeiro and De Souza, 2000) (Gendreau et al., 1999). The proposed algorithm by Ribeiro and De Souza, finds solutions that are better than the Takahashi-Matsuyama heuristic (Takahashi and Matsuyama, 1980) and F-tabu (Gendreau et al., 1999). As this metaheuristic is shown to be successful for finding the minimum Steiner tree in graphs, the use of tabu search for finding the PST that is close to the optimal for a MTMOPST problem is explored in the remainder of this section.

For combinatorial optimisation problems (definition 43), the search space can be explored by defining a move that when applied to an existing current solution gives a new solution. In the case of the Steiner tree problem, these moves are the addition and removal of Steiner nodes from an existing Steiner tree. Local search heuristics, such as hill climbing, find the local optimum when no improving moves are feasible (the stopping criterion), but this may not be the global optimum. The tabu search heuristic overcomes the local optimum problem by allowing moves that do not yield an improvement in the solution and through the use of a short-term memory structure, the tabu list, which stores recent moves that can not be reapplied to solutions for some given number of iterations.

The tabu search algorithm for a minimisation combinatorial problem is described in algorithm listing 6, however it is only a simple approach and may be extended and modified in a number of ways, as described below. Let $S(x)$ be the set of moves that are feasible to apply to solution x and $s(x)$ be the solution given when move s is applied to x. At line 1, the initial solution x, is initialised, and next it is set to x_{best}. To complete the initialisation phase, the iteration counter k begins at zero and the tabu list $\mathcal{T}$ is the empty set. The tabu search executes until the stopping criteria at line 6 is true, that is either the maximum number of iterations, k_{max} is met, or there are no non-tabu moves available for the search to follow. At line 7, of all the legitimate moves available at a given iteration, the move that yields the best neighbouring solution to the current solution x, is set as s_k and the new current solution becomes $s_k(x)$. Regardless of whether the current solution x, is better than than the best solution found up to iteration k (lines 8 and 9), the move s_k becomes tabu at line 12 and is followed by the removal of an existing tabu move.

Definition 43 (Combinatorial Optimisation Problem)**.** Given a set of feasible solutions $\mathcal{F}$ and a function $F : \mathcal{F} \to \mathbb{R}$, find the optimal solution $x \in \mathcal{F}$ for a minimisation problem such that $F(x) \leq F(y)$ for all $y \in \mathcal{F}$, or $F(x) \geq F(y)$ for a maximisation problem.

The choice of the move structure is dependent upon the problem. For the Steiner tree problem in graphs, a move is defined as the addition to or removal of a Steiner node from the solution. When a move is applied to a solution, it is marked as tabu by storing it

Algorithm 6: Tabu Search

Output: x_{best} `/* Tabu search best solution */`

```
1  begin
2      Let x ∈ F be an initial, feasible solution.
3      x_best = x
4      k = 0
5      T = ∅
6      while k < k_max ∨ S(x) − T ≠ ∅ do
7          x = s_k(x) such that for all s ∈ S(x) − T, F(s_k(x)) < F(s(x))
8          if F(x) < F(x_best) then
9              x_best = x
10         if |T| > t_limit then
11             T = T − {s | s ∈ S ∧ s ≠ s_k}
12         T = T ∪ {s_k}
13         k = k + 1
14     return x_best
```

in the tabu list for some number of iterations. A number of strategies for the tabu list have been proposed, that can generally be classified as recency-based or frequency-based. Recency-based tabu lists are commonly used and examples include lists that: store the moves of the last k-iterations; store the inverse moves of the moves for the last k-iterations; store the moves and their inverses of the last k-iterations. For recency-based tabu lists, k is typically a small value, and it is therefore short-term memory, but it is shown to yield good results. The alternative is frequency-based tabu lists that keep track of the frequency of moves (long-term memory), allowing the search to tend to less frequent moves and consequently allowing it to diversify the exploration of the search space, however this is not strictly a tabu list, as moves are not marked as tabu.

While the use of a tabu list of moves is effective at preventing cycles, they may prevent good moves from being executed where there is no risk of cycling. Aspiration functions are used to allow certain tabu moves to be chosen by the tabu search in lieu of the best non-tabu move. The most commonly used form allows a tabu move to applied to the current solution if it yields a solution that is better than the best solution, x_{best}, that is if there is a $t \in \mathcal{T}$ such that $F(t(x)) < F(x_{best}) \wedge F(t(x)) < F(s_k(x)) \wedge \forall t' \in \mathcal{T} \setminus \{t\} : F(t(x)) < F(t'(x))$, then $x_{best} = t(x)$.

The use of diversification strategies allow the tabu search to continue from a solution at some point in the search space other than the existing current solution. After some number of iterations, the diversification strategy is executed and the search restarts from the solution found by the strategy. The motivation for the use of diversification is to allow

the search space to explore other regions of the search space, as local search heuristics explore a localised search subspace. As described by Gendreau (Gendreau, 2003), the use of diversification gives the search breadth and is the most critical issue in the design of tabu search algorithms. Diversification also breaks any cycling if the diversified solution is not part of the cycle. For the tabu search algorithm to solve the Steiner tree problem in (Ribeiro and De Souza, 2000), the diversification strategy uses the Takahashi-Matsuyama heuristic to find a Steiner tree, which is rooted at a different vertex each time it is executed. This is a form of restart diversification, which determines the point in the search space where the search restarts. An informed restart diversification strategy could make use of a frequency-based tabu list to diversify to lesser explored regions of the search space. The other principal diversification strategy is continuous diversification, in which the objective function is modified to penalise moves to solutions with respect to their frequency in the tabu list. The penalty is increased for moves that feature either entirely or in part (given the moves' components) more frequently in the tabu list.

At each iteration of the tabu search, the solutions given by each move must be evaluated using the objective function, but this may be a time consuming operation. Rather than evaluate each candidate solution for the exact objective value, a surrogate objective function can be used instead. A good surrogate function is representative of the objective function, that is the ordering defined by the function over the set of feasible solutions is identical or the inverse to that of the objective function. After evaluation using the surrogate objective function, a subset of potentially favourable moves can then be evaluated using the objective function with the best being selected. An alternative of this candidate list strategy is the probabilistic tabu search, which select moves with probability proportional to its surrogate objective value.

In algorithm 6, the termination criteria is $k < k_{max} \vee S(x) - \mathcal{T} \neq \emptyset$, that is either some maximum number of iterations is met or there are no permissible non-tabu moves to explore. Other termination criteria include (and as listed in (Gendreau, 2003)): reaching a limit on CPU time; the number of iterations without an improvement in $F(x_{best})$ being equal to some limit; and when the objective value reaches some threshold.

4.4.2 Algorithm

To solve the MTPSTO problem, the use of the tabu search metaheuristic is proposed. The presentation of the tabu search algorithm begins with algorithm listings 7 and 8, which describe the procedure to find insertion and removal candidate moves, respectively.

Similar to the move structure defined in (Ribeiro and De Souza, 2000), a tabu search move is defined as the addition or removal of a router node from the PST. As is the case with Steiner trees, there is a subset of nodes that must always be included in the vertex set of the tree, these are the publisher node and the subscriber nodes. It follows that only the combination of router nodes is variable, hence the choice of move structure.

Definition 44 (MTPSTO Tabu Search Insertion Move)**.** The function m^+ is the insertion move function $m^+ : \mathcal{T}_c \times R \rightarrow \mathcal{H}_c$ where $\mathcal{T}_c$ is set of all PSTs in G_c, $\mathcal{H}_c$ is the set of all subgraphs in G_c. Let $G_c = (V_c, E_c)$ be a connectivity graph, $T_{PST} = (V_{PST}, E_{PST})$ be a PST in G_c, r be a node in the set $R \setminus V_{PST}$ (i.e. not in the PST) to be added to T_{PST}. $m^+(T_{PST}, r) = (V_{mod}, E_{mod})$ where $V_{mod} = V_{PST} \cup \{r\}$, $E_{mod} = \{(a, b) \mid (a, b) \in E_c \wedge a = r \wedge b \in V_{PST}\} \cup E_{PST}$, and $(V_{mod}, E_{mod}) \in \mathcal{H}_c$.

Definition 45 (MTPSTO Tabu Search Deletion Move)**.** The function m^- is the insertion move function $m^- : \mathcal{T}_c \times R \rightarrow \mathcal{H}_c$ where $\mathcal{T}_c$ is set of all PSTs in G_c, $\mathcal{H}_c$ is the set of all subgraphs in G_c. Let $G_c = (V_c, E_c)$ be a connectivity graph, $T_{PST} = (V_{PST}, E_{PST})$ be a PST in G_c, r be a node in the set $R \cap V_{PST}$ to be removed from T_{PST}. $m^-(T_{PST}, r) = (V_{mod}, E_{mod})$ where $V_{mod} = \{v \mid v \in V_{PST} \wedge v \neq r\}$, $E_{mod} = \{(a, b) \mid (a, b) \in E_{PST} \wedge a \neq r\}$, and $(V_{mod}, E_{mod}) \in \mathcal{H}_c$.

4.4.2.1 Determining Moves

Algorithm 7 returns a set M^+ of routers that can be added to a PST, T_{PST}. The algorithm guarantees that for all $r \in M^+$, there is at least one PST that has a vertex set $\{p\} \cup S \cup \{r\} \cup \{x \mid x \in R \wedge x \in V_{PST}\}$ in the connectivity graph G_c. The algorithm iterates over the set of routers that are in the connectivity graph, but not in T_{PST} (line 2). The insertion move of the router r is a potential candidate move, but only if its addition to the T_{PST} will result in a subgraph of G_c that contains a PST. At line 3, E_r is an edge set that contains all adjacent edges to r in G_c that are also adjacent to V_{PST}. If the cardinality of E_r is 1, then r will be a vertex of degree 1 in the subgraph induced by the algorithm, so r would be terminal vertex of any PST. This is a contradiction of the definition of a PST, so a test for this property is performed at line 4. The addition to T_{PST} of r and its edges adjacent to T_{PST}, need only contain one PST for the move to be feasible. For every pair of edges that are adjacent to r and two distinct nodes, x and y in V_{PST} (line 5), a cycle in is formed in T_{PST}. At line 6, $p_{T^{xy}_{PST}}$ is the set of edges between x and y in T_{PST}. Line 7 is an optimisation that ignores the pairs of edges under consideration if the path that it encloses is of length 1, as an edge can not be removed from $p_{T^{xy}_{PST}}$ to break the

cycle. A node can not be removed from the $p_{T_{PST}^{xy}}$, as it may not contain a router node and even if it did, it would result in a more complex move structure. It must be possible to remove an edge e from $p_{T_{PST}^{xy}}$ such that the two vertices which are endpoints of e are not router nodes (lines 8 to 10). To ensure that this is the case, one of the following predicates must be true: $\exists(u,v) \in p_{T_{PST}^{xy}} : u,v \in S$ (terminal vertices are subscribers); $\exists(u,v) \in p_{T_{PST}^{xy}} : u \in S \wedge v \in R \wedge |(u,x) \mid (v,x) \in E_{PST}|$ (there is a PST where u is a terminal subscriber and v is an internal router); or $\exists(u,v) \in p_{T_{PST}^{xy}} : u \in R \wedge v \in R \wedge |(u,x) \mid (u,x) \in E_{PST}| \geq 2 \wedge |(u,x) \mid (u,x) \in E_{PST}| \geq 2$ (two routers with at least one descendant in T_{PST}, which is reconnected by the router r and a pair of edges adjacent to T_{PST}). For each r whose addition to T_{PST} results in one of these predicates being true, it is added to M^+.

Algorithm 7: FindInsertionMoves(G_c, T_{PST}, R, S)

Input : $G_c = (V_c, E_c)$ `/* Connectivity graph */`
Input : $T_{PST} = (V_{PST}, E_{PST})$ `/* Current PST */`
Input : R `/* Set of routers` $R = V \setminus C$ `where` $C = \{p\} \cup S$ `*/`
Input : S `/* Set of subscribers */`
Output: M^+ `/* Set of routers that can be added to` T_{PST} `*/`

```
1  begin
2      foreach r ∈ V_PST \ R do
3          E_r ← {(a,b) | (a,b) ∈ E_c ∧ (a = r ∧ b ∈ V_PST) ∨ (a ∈ V_PST ∧ b = r)}
4          if |E_r| > 1 then
5              foreach ((r,x),(r,y)) ∈ E_r^2 do
6                  p_{T_PST^xy} ← FindPath(T_PST, x, y)
7                  if |p_{T_PST^xy}| > 1 then
8                      foreach (u,v) ∈ p_{T_PST^xy} do
9                          if u,v ∈ S ∨ (u ∈ S ∧ v ∈ R ∧ deg(v) > 2) ∨ (u ∈ R ∧ v ∈
                           S ∧ deg(u) > 2) ∨ (u ∈ R ∧ v ∈ R ∧ deg(u) > 2 ∧ deg(v) > 2)
                           then
10                             M_+ ← M_+ ∪ {r}
11                             break outer;
12     return M^+
```

Algorithm 8 describes the procedure by which removal moves are found. The algorithm determines if the graph induced by the removal of a router node from a PST and the addition of all edges between all pairs of nodes remaining in the PST will result in a graph that has at least one PST. The for loop at line 2 iterates over each router node in the PST, T_{PST}, where r is the current route node under consideration. Lines 3 to 5 create a graph G_{mod} that is a copy of the PST without the router r and its adjacent edges. In the

block that begins at line 6, for each edge in G_c, if its endpoints are in G_{mod}, then it is added to G_{mod}'s edge set. At line 5, G_{mod} may be a disconnected graph, so the addition of edges from G_c that have both endpoints in G_{mod} may reconnect the graph. If G_{mod} is not connected, then it will not contain a PST, so the algorithm tests for this at line 9. If G_{mod} is connected, the next condition for the graph to contain a PST is that for all routers in $(R \cap V_{mod}) \setminus \{r\})$, no router must have only one adjacent edge. If a router has one adjacent edge, it will be a terminal node of a PST for G_{mod}, and this is a contradiction to the definition of a PST. If G_{mod} contains no dangling router nodes, then it is added to the set of feasible moves M^-.

Algorithm 8: FindRemovalMoves(G_c, T_{PST}, R)

Input : $G_c = (V_c, E_c)$ `/* Connectivity Graph */`
Input : $T_{PST} = (V_{PST}, E_{PST})$ `/* Current PST */`
Input : R `/* Set of routers` $R = V \setminus C$ `where` $C = \{p\} \cup S$ `*/`
Output: M^- `/* Set of routers that can be removed from` T_{PST} `*/`

```
1  begin
2      foreach r ∈ V_PST \ ({p} ∪ S) do
3          V_mod ← {v | v ∈ V_PST ∧ v ≠ r}
4          E_mod ← {(a, b) | (a, b) ∈ E_PST ∧ (a ≠ r ∧ b ≠ r)}
5          G_mod ← (V_mod, E_mod)
6          foreach (x, y) ∈ E_c do
7              if x, y ∈ V_mod then
8                  E_mod ← E_mod ∪ {(x, y)}
9          if IsConnectedGraph(G_mod) then
10             foreach r ∈ R ∩ V_mod do
11                 if |{(a, b) | (a, b) ∈ E_mod ∧ (a = r ∨ b = r)}| < 2 then
12                     Continue at line 2
13             M⁻ ← M⁻ ∪ {r}
14     return M⁻
```

4.4.2.2 Move Evaluation

The application of a move to a PST gives a subgraph of the connectivity graph (definition 44 and definition 45). Algorithms 7 and 8 ensure that the subgraph will contain at least one PST. Algorithm 9 finds the PST in a graph that maximises the trustworthiness within the overhead budget constraint. If no PST is found within the overhead budget constraint, the PST that maximises the trustworthiness is found. Although the algorithm describes the evaluation of an insertion move, the evaluation of a deletion move only differs at line 4. The for loop at line 5 iterates over each PST found by the PST

enumeration algorithm (algorithm 5). The PST is evaluated and if found to be a better PST than those previously evaluated, then T_{max}, τ_{max} and O_{max} are set appropriately. The algorithm terminates at line 22 by returning the best PST T_{max} that has been found in the subgraph G_{mod}, along with its trust value (τ_{max}) and its overhead value (O_{max}).

Algorithm 9: EvaluateMove(G_c, T_{PST}, r, B)

Input : $G_c = (V_c, E_c)$ /* Connectivity graph */
Input : $T_{PST} = (V_{PST}, E_{PST})$ /* Current PST */
Input : r /* Router to be added */
Input : B /* Overhead Budget */
Output: $(T_{max}, \tau_{max}, O_{max})$

```
1  begin
2      τ_max ← −∞
3      O_max ← ∞
4      G_mod = m⁺(T_PST, r)    /* If removal move, function m⁻ is used */
5      foreach T_next ∈ PSTEnumeration(G_mod) do
6          τ_{T_next} ← EvaluateTreeTrust(T_next)
7          O_{T_next} ← EvaluateTreeOverhead(T_next)
8          if O_max > B ∧ O_{T_next} > B then
9              if τ_{T_next} > τ_{T_max} then
10                 T_max ← T_next
11                 τ_max ← τ_{T_next}
12                 O_max ← O_{T_next}
13         else if O_max < B ∧ O_{T_next} < B then
14             if τ_{T_next} > τ_{T_max} then
15                 T_max ← T_next
16                 τ_max ← τ_{T_next}
17                 O_max ← O_{T_next}
18         else if O_max > B ∧ O_{T_next} < B then
19             T_max ← T_next
20             τ_max ← τ_{T_next}
21             O_max ← O_{T_next}
22     return (T_max, τ_max, O_max)
```

4.4.2.3 Surrogate Objective Function

Given a set of moves $M^+ \cup M^-$, each move must be evaluated to determine which gives the best PST tree with respect to the objectives of maximising trust within some overhead budget. As mentioned in section 4.4.1, evaluation of each move to determine the exact objective value of the solution it yields can be time consuming. To address this issue, the use of surrogate evaluation function is proposed that can be used to reduce the cardinality of the set $M^+ \cup M^-$. The smaller set of moves is then evaluated fully for exact objective

values of the solutions that result from their application.

The PST solution to a MTPSTO problem maximises the trust held in the PST by the least trusting node (through the use of the leximin social welfare functional) within some overhead budget. A greedy approach is adopted for the surrogate objective function, which seeks to maximise the improvement to the least well-off node. Given the subgraph G_{mod} that is induced by the application of a move m to the current PST solution, T_{PST}, the surrogate objective value is given by the most trusted path between the node with the least trust in T_{PST} and the publisher, p. Due to the fact that a semiring-based trust model for path trust is used (section 3.2.2), it is possible to to use the generic shortest distance algorithm algorithm defined in (Theodorakopoulos and Baras, 2006) to find the most trusted path between two nodes.

Definition 46 (Surrogate Objective Function for MTPSTO Tabu Search)**.** The surrogate objective function is defined as, $sobj : \mathcal{H}_c \rightarrow \mathbb{R}_{[0,1]}$ where $\mathcal{H}_c$ is the set of all subgraphs of G_c and $\mathbb{R}_{[0,1]}$ is the unit interval. Given a PST T_{PST} and a router insertion move $r \in V_{PST} \setminus (\{p\} \cup S)$, let $G_{mod} = (V_{mod}, E_{mod})$ be the graph induced by the addition of r to T_{PST}, where $V_{mod} = V_{PST} \cup r$ and $E_{mod} = E_{PST} \cup \{(a, b) \mid (a, b) \in E_c \wedge (a = r \wedge b \in V_{PST})\}$, and is a subgraph of the connectivity graph, G_c. $sobj(G_{mod}) = max_{\forall \sigma_{x,p} \in \mathcal{P}_{x,p}}(\tau_x(\sigma_{x,p}))$ where $\forall y \in V_{PST} \setminus \{x\} : \tau_x(T_{PST}) < \tau_y(T_{PST})$, $\mathcal{P}_{x,p}$ is the set of paths in G_{mod} between x and p and τ_x is the trust function of x.

4.4.2.4 Penalty Function

Tabu search is designed for combinatorial problems in the form given by definition 43. The MTPSTO problem differs to this in that not only does it maximise the trust objective, but it does so within the overhead budget constraint. There are a number of approaches to modifying tabu search to handle these types of problems, these are: a static penalty on solutions that do not respect the constraint; or an adaptive penalty function where the penalisation value applied to solutions that breach the problem constraint is dependent upon some other variable.

Two techniques are proposed for the MTPSTO problem. The first is a simple static penalty function that penalises any over-budget solution by decreasing the trust value by 50%. The second is the Near Feasibility Threshold (NFT) technique for tabu search (Kulturel-Konak et al., 2004), which determines the penalty to be applied to an over-budget solution with respect to the short-term and long-term memory structures of the tabu search. The method uses the properties of moves in the tabu list to determine the

NFT. Solutions that are infeasible are mildly punished if within the NFT region, and more significantly so if beyond it. The NFT approach is compared to alternative methods by Nonobe and Ibaraki (Nonobe and Ibaraki, 1998) and Gendreau et al. (Gendreau et al., 1994). While it outperforms the Nonobe and Ibaraki method with respect to best solution quality for a number of problems, results in comparison to the method devised by Gendreau et al. are mixed. For the orienteering problem, which is a maximisation problem with a single constraint, both the NFT approach and the Gendreau et al. approach are shown to be effective.

Definition 47 (NFT for Tabu Search (Kulturel-Konak et al., 2004))**.** The feasibility ratio, R_j at iteration j of an instance of a tabu search is given by equation 4.1 where F_j is the number of moves in the tabu list that have yielded feasible solution at iteration j and T_j is the size of the tabu list at iteration j.

$$R_j = \frac{F_j}{T_j} \tag{4.1}$$

The NFT for a constraint i is determined by equation 4.2. If the current move gives a feasible solution, the NFT increases to encourage searching in infeasible region. If the move is infeasible, then the NFT decreases. This increases the region beyond the NFT, which is subject to greater penalisation than that within the NFT.

$$NFT_{i,j+1} = \begin{cases} NFT_{i,j}\left(1 + \frac{R_j}{2}\right), & \text{if move is feasible} \\ NFT_{i,j}\left(\frac{1+R_j}{2}\right), & \text{if move is infeasible.} \end{cases} \tag{4.2}$$

Should a solution x be beyond the NFT_i, its objective value $F(x)$ is penalised to give $F_p(x)$ as defined in equation 4.3 where F_{all} is the unpenalised objective value of the best solution found, F_{feas} is the objective value of the best feasible solution found, n is the number of constraints, $d_i(x, B)$ is the value by which x is over the i's constraint value B, NFT_i is the NFT of constraint i, and k_i is an exponent for the constraint i that amplifies the penalty when x is beyond NFT_i.

$$F_p(x) = F(x) + (F_{all} - F_{feas}) \times \sum_{i=1}^{n} \left(\frac{d_i(x, B)}{NFT_i}\right)^{k_i} \tag{4.3}$$

4.4.2.5 Diversification

Two restart diversification techniques are proposed: Takahashi-Matsuyama diversification, which uses the Steiner tree heuristic by Takahashi and Matsuyama (Takahashi and Matsuyama, 1980) to form a PST (algorithm 10); and SPT diversification that creates PSTs that are also shortest path trees to some arbitrary node. After every n iterations of the tabu search algorithm or when there are no moves for the tabu search to exploit, the diversification method is invoked.

Takahashi and Matsuyama present a Steiner tree heuristic in (Takahashi and Matsuyama, 1980) that can easily be modified to find a Steiner tree that is a PST. The only modification required is to stipulate that the initial vertex of the algorithm is either a publisher or a subscriber, and that the remaining subscribers and, if not the initial vertex, the publisher are the Steiner nodes. Algorithm 10 describes the Takahashi-Matsuyama diversification method. Lines 3 to 6 initialise the diversified PST to contain a randomly chosen node from the set of C containing the publisher and subscribers. At each iteration of the while loop at line 7, the node $y \in C \setminus V_{PST}$ with the shortest path to any node $z \in V_{PST}$ is added to T_{PST} (line 16 and 17). At line 12, the ShortestPath function is a single-pair shortest path algorithm, which can be implemented using Dijkstra's algorithm (Dijkstra, 1959). The Endpoint function at line 16 returns the vertices for a given edge set.

Algorithm 10: Takahashi-Diversification(G_c, p, S)

```
1  begin
2      C = S ∪ {p}
3      Let x be a randomly chosen node from the set C
4      V_PST = {x}
5      E_PST = ∅
6      T_PST = (V_PST, E_PST)
7      while |V_PST ∩ C| ≠ C do
8          length ← ∞
9          δ_min ← null
10         foreach y ∈ C \ V_PST do
11             foreach z ∈ V_PST do
12                 δ_y,z ← ShortestPath(G_c, y, z)
13                 if |σ_y,z| < length then
14                     δ_min ← δ_y,z
15                     length ← |δ_y,z|
16         V_PST = V_PST ∪ Endpoints(δ_min)
17         E_PST = E_PST ∪ δ_min
18     return T_PST
```

The SPT diversification method chooses a node x at random from the set $S \cup \{p\}$. The initial PST tree $T_{PST} = (V_{PST}, E_{PST})$ is created, where $V_{PST} = \{x\}$ and $E_{PST} = \emptyset$. For each $y \in (S \cup \{p\}) \setminus V_{PST}$, $V_{PST} = V_{PST} \cup \sigma_{y,x}$ where $\sigma_{y,x}$ is the set of vertices of the shortest path between x and y in G_c, and $E_{PST} = E_{PST} \cup \delta_{x,y}$ where $\delta_{x,y}$ is the set of edges of the shortest path between x and y in G_c.

4.4.2.6 Tabu Search Algorithm for MTPSTO

The tabu search algorithm for the MTPSTO problem is given in algorithm 11 and is described by a flowchart in figure 4.1. At line 4, the initial solution is found using the diversification algorithm, either the Takahashi-Matsuyama diversification (as listed) or the SPT diversification. The initial solution is stored as the current PST solution, T_{PST} and evaluated for its trust value at line 5. Should the initial solution be over budget, it is penalised using the penalty function of choice (lines 6 and 7). The initial solution is made the best solution at line 8.

The tabu search executes until i_{max} iterations without any improvement in the best solution have occurred. From line 11 to 14, the set of feasible insertion and removal moves for the current PST, T_{PST} are found. Rather than evaluate every moves found, an aggressive optimisation is performed where only the best insertion and best deletion moves with respect to the surrogate objective function are considered (lines 13 and 14), however if both of these moves are tabu, diversification is invoked (lines 15 and 18). These moves are evaluated and if over budget, they are penalised using the penalty function (lines 19 to 20). Of the two moves under consideration during an iteration of a tabu search, the one leading to the more trustworthy PST (lines 25 to 28) is chosen as the next current solution of the search. Two approaches are considered for the selection of the most trusted PST given the chosen moves. The adaptive PST policy chooses the PST that maximises trust and has not been visited previously, while the best PST policy considers previously visited PSTs. The preferred move is placed in the tabu list and the oldest move in the list removed if the tabu list has reached some maximum (line 29). Should the PST found during the current iteration be the most trustworthy one found so far, then it is set as the best solution and the value i, which counts the number of iterations since an improvement in the best solution found, is set to zero. If this is not the case, i is incremented. The number of iterations since the last diversification has not reached the maximum, the current solution (upon which moves are applied in the next iteration) is set to the best solution of those given by the moves m^+ and m^-.

4.5 Summary

This chapter begins with the definition of the Maximum Trusted PST with Overhead Budget (MTPSTO) problem and a proof that shows that the problem is in NP-complete. The algorithms presented in this chapter make use of Char's spanning tree enumeration algorithm (Char, 1968), which is modified to provide an enumeration of PSTs. An exhaustive search algorithm for the problem is presented, which is expected to be of little use for anything but the smaller problem instances due to the problem complexity. This provides the motivation for the use of the tabu search metaheuristic. A brief background on tabu search is provided, followed by the tabu search algorithm for the MTPSTO problem.

Algorithm 11: TabuSearch(G, p, R, S, B, i_{max}, Δ_{max})

```
Input  : G = (V, E)                                  /* Connectivity Graph */
Input  : R                                           /* Set of routers in V (R ⊂ V) */
Input  : S                                           /* Set of subscribers in V (S ⊂ V) */
Input  : B                                           /* Overhead budget */
Input  : i_max                                       /* Maximum iterations without improvement */
Input  : Δ_max                                       /* Maximum iterations before diversification */
1  begin
2      i = 0; Δ = 0
3      L ← ∅                                         /* Tabu List */
4      T_PST ← Takahashi(G)
5      τ_{T_PST} ← EvaluateTreeTrust(T_PST)
6      if EvaluateTreeOverhead(T_PST) > B then
7          τ_{T_PST} ← PenaltyFunction(τ_{T_PST})
8      τ_best ← τ_{T_PST}; T_best ← T_PST
9      while i < i_max do
10         O_next ← ∞; τ_next ← −∞
11         M+ ← FindInsertionMoves(G, T_PST, R, S)
12         M− ← FindRemovalMoves(G_c, T_PST, R)
13         m+ ← max_{∀m∈M+}(sobj(m))
14         m− → max_{∀m∈M−}(sobj(m))
15         if m+ ∈ L ∧ m− ∈ L then
16             T_PST ← Takahashi(G)
17             Δ ← 0
18             Continue at line 9
19         (T_{m+}, τ_{m+}, O_{m+}) ← EvaluateMove(G, T_c, m+)
20         (T_{m−}, τ_{m−}, O_{m−}) ← EvaluateMove(G_c, T_PST, m−)
21         if O_{m+} > B then
22             τ_{m+} ← PenaltyFunction(τ_{m+})
23         if O_{m−} > B then
24             τ_{m−} ← PenaltyFunction(τ_{m−})
25         if τ_{m+} > τ_{m−} then
26             m_tabu ← m+; T_next ← T_{m+}; τ_next ← τ_{m+}
27         else
28             m_tabu ← m−; T_next ← T_{m−}; τ_next ← τ_{m−}
29         L ← L ∪ {m_tabu}
30         if τ_next > τ_best then
31             T_best ← T_next;
32             τ_best ← τ_next
33             i ← 0; Δ ← 0
34         else i ← i + 1
35         if Δ < Δ_max then
36             T_PST ← T_next; Δ ← Δ + 1
37         else
38             T_PST ← Takahashi(G)
39             Δ ← 0
```

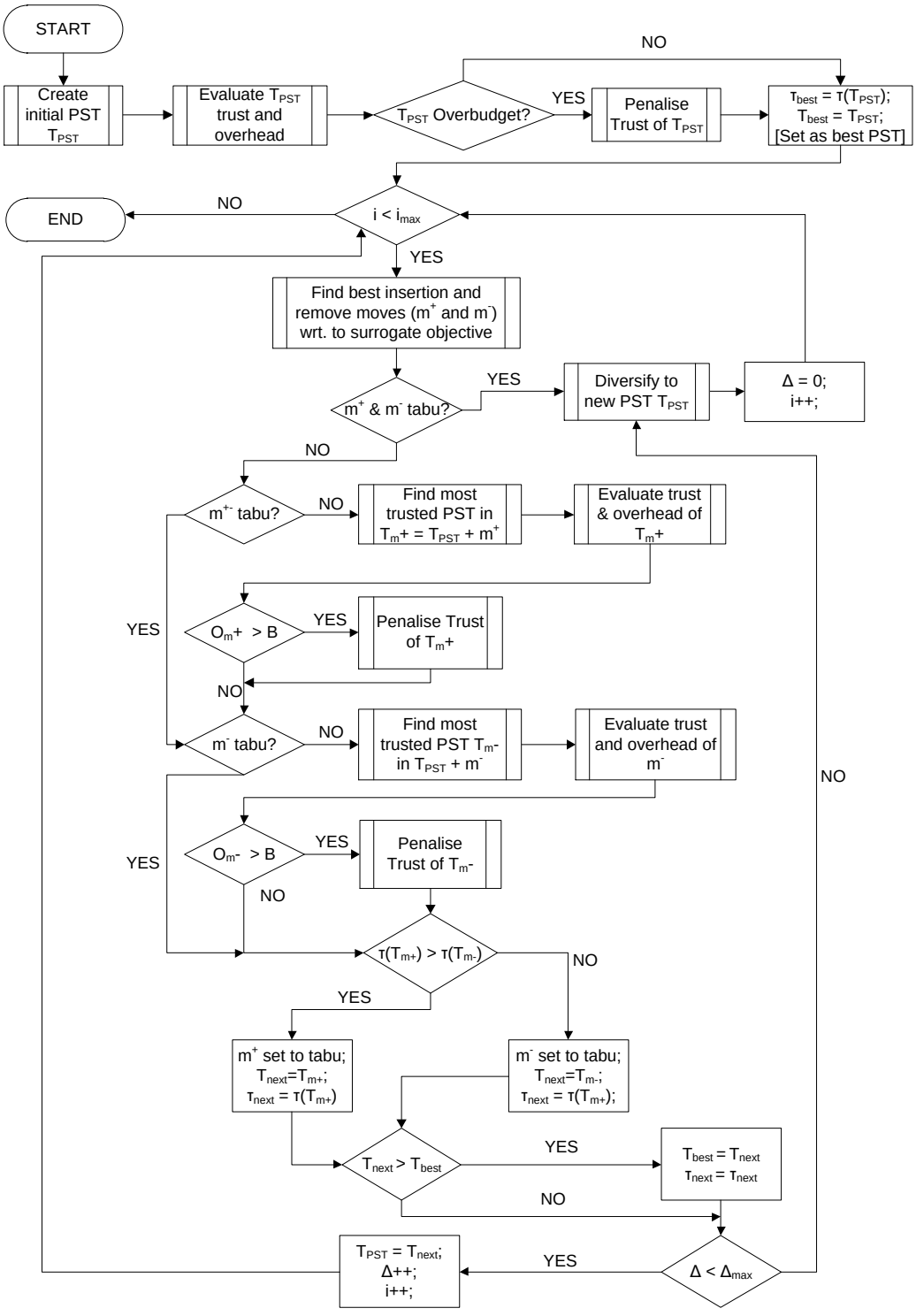

Figure 4.1: Flowchart Describing Algorithm 11

Chapter 5

Evaluation and Computational Results

5.1 Evaluation Overview

The mathematical analysis of tabu search is an open area of research (Glover et al., 1993), so to analyse the performance of the tabu search algorithms that are proposed in chapter 4, an evaluation based on experimental results is presented in this chapter.

The evaluation is concerned with two properties, the quality of the solutions found and the running times of the algorithm. The former is given by the relative error of the trust and overhead values with respect to the optimal solution, and the latter is measured by subtracting the values returned from the `System.nanoTime()` method of the Java class library, which is called immediately prior to and after the experiment. Note that the running times exclude any operations required to initialise the experiment, such as the instantiation of the connectivity graph from its GraphML[1] representation. The aim of the evaluation is to draw conclusions on the suitability of tabu search for the MTPSTO problem by:

- comparing the quality of the solutions found by the tabu search algorithms to the optimal solutions;
- comparing the running times of the tabu search algorithms to the exhaustive search algorithm;
- assessing the difference in the quality of solutions found by the algorithms using the static and NFT penalty functions;

[1] http://graphml.graphdrawing.org/

- assessing the difference in running times between algorithms using the static and NFT penalty functions;
- determining if either the Takahashi-Matsuyama or the Shortest Path Tree diversification strategies yield better solutions with respect to running time and/or solution quality;
- examining the difference in solution quality and running times between the best and adaptive PST selection strategies;
- comparing the shortest path tree and the Steiner tree (found by the Takahashi-Matsuyama heuristic), both rooted at the publisher and spanning the set of subscriber nodes, to the optimal solutions and those found by the tabu search algorithms;
- comparing the average running times, average relative errors in trust and overhead values, and the number of exact and over budget solutions found by each algorithm.

The algorithms described in chapter 4 have been implemented using the Java programming language. The implementations are dependent upon two third-party libraries, the Java Universal Network/Graph Framework (JUNG)[2] (ver. 2.01) and the OpenTS library[3] (ver. 1.0-exp10), a tabu search framework. The JUNG library provides a framework that allows for the modelling, analysis and manipulation of graphs. The OpenTS library provides a tabu search framework that is used as the basis of the implementations of the tabu search algorithms.

Each experiment was executed five times unless stated otherwise. The running times given in the results tables and in the discussion throughout the remainder of this chapter are averages over these five execution runs unless stated otherwise. The solution quality is measured by the relative errors in the trust and overhead values between the approximation and the exact solution. Equation 5.1 defines relative error where v is a value greater than 0 and v_a is an approximation of v.

$$\eta = \left| \frac{v - v_a}{v} \right| \tag{5.1}$$

[2] http://jung.sourceforge.net/
[3] http://www.coin-or.org/Ots/index.html

5.2 Evaluation Environment

All experiments were performed on the Amazon Elastic Compute Cloud (Amazon EC2) service[4] using a High-Memory Extra Large instance (m2.xlarge). The instance has a specification of 17.1 Gb of RAM, two virtual cores with 3.25 EC2 Compute Units (one EC2 compute unit is equivalent to a 1.0-1.2 GHz 2007 Opteron or 2007 Xeon)[5] reported as two 2.67 GHz Intel Xeon X5550 CPUs by the command `cat /proc/cpuinfo`, and 420 Gb of instance storage. Amazon Linux AMI 64-bit with Linux kernel 2.6.35.11 was the chosen operating system image. All implementations were executed using the Java runtime environment available in this image, IcedTea6 1.9.1[6]. For each experiment, the only option passed to the Java virtual machine was to set the maximum heap size to 16 Gb, `-Xmx16G`.

The choice of this evaluation environment was motivated by the high memory requirement of the exhaustive search algorithm. To ensure fair comparability of the running times of the proposed tabu search algorithms with those of the exhaustive search, the same instance type was used, despite the tabu search algorithms having a smaller memory footprint.

5.3 Evaluation Data Sets

Motivated by the need of Operations Research (OR) researchers to compare algorithms for identical problems, a number of test data sets was devised for a variety of OR problems (Beasley, 1990). The OR library[7] includes test data sets for Steiner problems, such as the Steiner tree in graphs and the prize collecting Steiner tree. As the problem addressed in this thesis is novel one, it was not possible to make use of this library, so a series of test data sets was generated for the evaluation of the proposed algorithms. The design choices for the evaluation test data sets are given in the remainder of this section.

The three problem sets, A, B, and C, used in the evaluation are described in tables 5.1, 5.3, and 5.4 respectively. For each problem in the test data set, the problem input with reference to the problem definition given in problem definition 2 consists of three types of variable:

- publish/subscribe inputs,

[4] http://aws.amazon.com/ec2/
[5] http://aws.amazon.com/ec2/instance-types/
[6] http://icedtea.classpath.org/
[7] http://people.brunel.ac.uk/ mastjjb/jeb/info.html

  - E (set of events randomly chosen from the event distribution Ev)
  - B (overhead budget)
  - A_p (publisher's advertisement)
  - S (set of subscribers)
  - $SF_S = \{sf_s \mid s \in S\}$ (set of subscription functions for each subscriber where sf_s is the subscription function of subscriber s)
  - R (set of routers);

- connectivity graph, $G_c = (V, E_c)$;
- the trust graph used as input to the individuals' trust functions, $G_\tau = (V, E_\tau)$;

The test data sets are comprised of problem instances with varying $|R|$, as the primary objective is to analyse the proposed algorithms with respect to connectivity graphs of increasing sizes in both V and E_c, the latter achieved by maintaining the graph density within a test data set to be approximately equal across the problem instances as $|R|$ increases. By increasing the number of routers in each problem, the test data sets allow for the evaluation of algorithms with respect to problems of increasing complexity, as both the number of possible moves at each iteration of the tabu search and the dominant factor of the PST enumeration algorithm $n(t + t_1)$ increase. For all problems, the cardinality of the set of subscribers, S, is 5.

For a given test data set, each problem is identified by an identifier in the following format, `<Problem Data set><Subset Number>-<Problem Number>` where `<Problem Data set>` is the data set identifier (A, B, or C), `<Subset Number>` indicates the subset within the problem data set and is equal to value of $|R|$ for each problem in the subset, and `<Problem Number>` is the problem identifier where $1 \implies B = 2000$, $2 \implies B = 3000$, $3 \implies B = 4000$, $4 \implies B = 5000$ and $5 \implies B = 2^{31} - 1$ (Java's largest maximum integer). Test data sets are made of subsets of five problems, each problem sharing identical parameters other than the value of the overhead budget, B. The values chosen for B exclude 1000 as there is no optimal PST solution with an overhead value that is less than or equal to 1000 for all problems where the optimal solution is known. No budgets are considered where $5000 < B < 2^{31} - 1$, as all optimal solutions found where $B = 2^{31} - 1$ are identical to those where $B = 5000$, implying that there is no PST, T, with higher trust values where $5000 < O_T < 2^{31} - 1$ and O_T is the overhead value of T. The choice of $B = 2^{31} - 1$ is so that the algorithms can find the most trusted PST with the largest permitted integer overhead budget.

Problem set A (table 5.1) consists of problems where $1 \leq |R| \leq 9$. Set A is the only problem set where optimal solutions are available for comparison to those found by the tabu search algorithms, as for larger problems, the running times of the exhaustive search are excessive. Table 5.2 shows the execution times of the exhaustive search for each subset of problems in problem set A. The average times given are those of the five algorithm runs for each subset of problems, except for A9 where this was impractical. Each experiment run finds the solutions where the overhead budget is 2000, 3000, 4000, 5000, and $2^{31} - 1$. For problem subsets A0 to A4, the exhaustive search executes quickly, however, there is an order of magnitude difference in the execution time with the addition of an additional router to problems of subsets A5 and A8. The timings exhibit non-linear growth, which is to be expected, as the problem under consideration is in NP-Complete. Given the execution time of the exhaustive search for problem A9, attempts to solve larger problems were not attempted.

The remaining problem sets B (table 5.3) and C (table 5.4), allow for the evaluation of the algorithms with respect to larger cardinalities of the set R. For problem set B, problems with between 10 to 19 routers are defined, and for problem set C, from 20 to 100 routers in increments of 10. Problems C100-1, C100-2, C100-3, C100-4 and C100-5 are the largest problems by $|V|$, $|E_c|$ and $|R|$ that are considered, where the connectivity graph has 106 vertices and 2782 edges, and the trust graph of the same number of vertices and 1020 edges.

5.3.1 Publish/Subscribe Properties

To generate the advertisement A_p for the publisher p, the set of events E drawn from the event distribution Ev and the subscriptions sf_s for all $s \in S$, the Numbers Interval (NI) model is used (Huang and Garcia-Molina, 2003). The model defines a subscription as an interval on the real line, with events represented as real values. Given a subscription of interval $[x, y]$, event i matches the subscription if and only if $x \leq i \leq y$ is true, where $x, y, i \in \mathbb{R}$. Huang and Molina define three variants of the NI model to generate subscriptions: the NI-R (random center) model where a real value is chosen at random to be the centre of the subscription interval; the NI-X model where the location of the subscriber (x-coordinate) is used as the centre of the interval; and the NI-Xmod model where some random offset is added to the x-coordinate of the subscriber to give the centre of the interval. The NI-X and NI-Xmod models generate localised subscriptions, which is of interest when evaluating the overhead metric, but is irrelevant with respect to the

	Properties								Solution	
Pr	$\|V_c\|$	$\|E_c\|$	$\|S\|$	$\|R\|$	B	G_c Density	$\|V_{trust}\|$	$\|E_{trust}\|$	$\tau_{T_{best}}$	$O_{T_{best}}$
A0-1	6	7	5	0	2000	0.467	6	20	-	-
A0-2	6	7	5	0	3000	0.467	6	20	0.0322	2179
A0-3	6	7	5	0	4000	0.467	6	20	0.0322	2179
A0-4	6	7	5	0	5000	0.467	6	20	0.0322	2179
A0-5	6	7	5	0	$2^{31}-1$	0.467	6	20	0.0322	2179
A1-1	7	10	5	1	2000	0.476	7	30	-	-
A1-2	7	10	5	1	3000	0.476	7	30	0.0181	2398
A1-3	7	10	5	1	4000	0.476	7	30	0.0181	2398
A1-4	7	10	5	1	5000	0.476	7	30	0.0181	2398
A1-5	7	10	5	1	$2^{31}-1$	0.476	7	30	0.0181	2398
A2-1	8	14	5	2	2000	0.5	8	40	0.0931	1850
A2-2	8	14	5	2	3000	0.5	8	40	0.0931	1850
A2-3	8	14	5	2	4000	0.5	8	40	0.0931	1850
A2-4	8	14	5	2	5000	0.5	8	40	0.0931	1850
A2-5	8	14	5	2	$2^{31}-1$	0.5	8	40	0.0931	1850
A3-1	9	18	5	3	2000	0.5	9	50	-	-
A3-2	9	18	5	3	3000	0.5	9	50	0.0224	2917
A3-3	9	18	5	3	4000	0.5	9	50	0.0224	2917
A3-4	9	18	5	3	5000	0.5	9	50	0.0224	2917
A3-5	9	18	5	3	$2^{31}-1$	0.5	9	50	0.0224	2917
A4-1	10	22	5	4	2000	0.489	10	60	-	-
A4-2	10	22	5	4	3000	0.489	10	60	0.1855	2224
A4-3	10	22	5	4	4000	0.489	10	60	0.1855	2224
A4-4	10	22	5	4	5000	0.489	10	60	0.1855	2224
A4-5	10	22	5	4	$2^{31}-1$	0.489	10	60	0.1855	2224
A5-1	11	27	5	5	2000	0.491	11	70	-	-
A5-2	11	27	5	5	3000	0.491	11	70	0.0542	2262
A5-3	11	27	5	5	4000	0.491	11	70	0.0812	3196
A5-4	11	27	5	5	5000	0.491	11	70	0.0812	3196
A5-5	11	27	5	5	$2^{31}-1$	0.491	11	70	0.0812	3196
A6-1	12	33	5	6	2000	0.5	12	80	-	-
A6-2	12	33	5	6	3000	0.5	12	80	-	-
A6-3	12	33	5	6	4000	0.5	12	80	0.0360	3846
A6-4	12	33	5	6	5000	0.5	12	80	0.0360	4414
A6-5	12	33	5	6	$2^{31}-1$	0.5	12	80	0.0360	4414
A7-1	13	39	5	7	2000	0.5	13	90	-	-
A7-2	13	39	5	7	3000	0.5	13	90	-	-
A7-3	13	39	5	7	4000	0.5	13	90	0.0692	3570
A7-4	13	39	5	7	5000	0.5	13	90	0.0692	3570
A7-5	13	39	5	7	$2^{31}-1$	0.5	13	90	0.0692	3570
A8-1	14	45	5	8	2000	0.495	14	100	-	-
A8-2	14	45	5	8	3000	0.495	14	100	-	-
A8-3	14	45	5	8	4000	0.495	14	100	0.0031	3657
A8-4	14	45	5	8	5000	0.495	14	100	0.0031	4031
A8-5	14	45	5	8	$2^{31}-1$	0.495	14	100	0.0031	4031
A9-1	15	52	5	9	2000	0.495	15	110	0.2184	1885
A9-2	15	52	5	9	3000	0.495	15	110	0.2184	1885
A9-3	15	52	5	9	4000	0.495	15	110	0.2184	1885
A9-4	15	52	5	9	5000	0.495	15	110	0.2184	1885
A9-5	15	52	5	9	$2^{31}-1$	0.495	15	110	0.2184	1885

Table 5.1: Problem Set A

Pr.	Min. (s)	Max. (s)	Avg. (s)
A0	0.0153	0.0871	0.0339
A1	0.0239	0.1522	0.058
A2	0.1238	0.3774	0.1852
A3	0.8051	1.2791	0.9304
A4	1.7682	2.4166	1.9041
A5	19.5833	20.212	19.7224
A6	285.8669	287.4492	286.3381
A7	945.8277	949.9657	947.4963
A8	6149.868	6164.197	6158.712
A9	97672.93	97672.93	-

Table 5.2: Execution Times of Exhaustive Search Results for Problem Set A

evaluation objectives described above. In addition, as the generated connectivity graphs do not encapsulate any location information, the NI-R model is used.

Advertisements are not specified in the NI-R model, so it is extended such that A_p is defined as an interval $[a, b]$ on the real line. For each $e \in E$, $a \leq e \leq b$ and is chosen at random from the event distribution Ev, a normal distribution $\mathcal{N}(\mu, \sigma^2)$ where $\mu = (b - a)/2$ (mean) and $\sigma = (\mu - a)/3$ (standard deviation). Where $e > b \wedge e < a$, the value is discarded. The set of events is identical for all problems within a problem set and consists of one thousand events. For each subscriber $s \in S$, the subscription $sf_s = [x, y]$ such that $a \leq x \leq b \wedge a \leq y \leq b \wedge x \leq y$. The values x and y are drawn from the same normal distribution as the events. In all experiments, the chosen values for a and b are 0.333 and 0.666 respectively.

5.3.2 Connectivity Graph

Many networks topologies exhibit the power-law property in the number of adjacent edges at vertices such that a small set of vertices have a much higher degree than others (i.e. there exists a small set of hub vertices). These topologies are referred to as scale-free networks, examples of which include the World Wide Web (Barabási et al., 2000) and social networks, such as the e-mail network where e-mail addresses are vertices and e-mails are edges (Ebel et al., 2002). For each problem instance, the connectivity graph $G_c = (V, E_c)$ was generated using the `EppsteinPowerLawGenerator` class from the JUNG library that generates a graph with power law properties (Eppstein and Wang, 2002). The algorithm begins by generating a random graph with m-edges and then running for a number of iterations, during which an edge is removed from the graph and a new edge added that connects two nodes where one is chosen with probability that is proportional to its degree, subject to the edge not being a member of the edge set and not being a

Pr	$\|V_c\|$	$\|E_c\|$	$\|S\|$	$\|R\|$	B (Budget)	G_c Density	$\|V_{trust}\|$	$\|E_{trust}\|$
B10-1	16	60	5	10	2000	0.5	16	120
B10-2	16	60	5	10	3000	0.5	16	120
B10-3	16	60	5	10	4000	0.5	16	120
B10-4	16	60	5	10	5000	0.5	16	120
B10-5	16	60	5	10	$2^{31}-1$	0.5	16	120
B11-1	17	68	5	11	2000	0.5	17	130
B11-2	17	68	5	11	3000	0.5	17	130
B11-3	17	68	5	11	4000	0.5	17	130
B11-4	17	68	5	11	5000	0.5	17	130
B11-5	17	68	5	11	$2^{31}-1$	0.5	17	130
B12-1	18	76	5	12	2000	0.496	18	140
B12-2	18	76	5	12	3000	0.496	18	140
B12-3	18	76	5	12	4000	0.496	18	140
B12-4	18	76	5	12	5000	0.496	18	140
B12-5	18	76	5	12	$2^{31}-1$	0.496	18	140
B13-1	19	85	5	13	2000	0.497	19	150
B13-2	19	85	5	13	3000	0.497	19	150
B13-3	19	85	5	13	4000	0.497	19	150
B13-4	19	85	5	13	5000	0.497	19	150
B13-5	19	85	5	13	$2^{31}-1$	0.497	19	150
B14-1	20	95	5	14	2000	0.5	20	160
B14-2	20	95	5	14	3000	0.5	20	160
B14-3	20	95	5	14	4000	0.5	20	160
B14-4	20	95	5	14	5000	0.5	20	160
B14-5	20	95	5	14	$2^{31}-1$	0.5	20	160
B15-1	21	105	5	15	2000	0.5	21	170
B15-2	21	105	5	15	3000	0.5	21	170
B15-3	21	105	5	15	4000	0.5	21	170
B15-4	21	105	5	15	5000	0.5	21	170
B15-5	21	105	5	15	$2^{31}-1$	0.5	21	170
B16-1	22	115	5	16	2000	0.498	22	180
B16-2	22	115	5	16	3000	0.498	22	180
B16-3	22	115	5	16	4000	0.498	22	180
B16-4	22	115	5	16	5000	0.498	22	180
B16-5	22	115	5	16	$2^{31}-1$	0.498	22	180
B17-1	23	126	5	17	2000	0.498	23	190
B17-2	23	126	5	17	3000	0.498	23	190
B17-3	23	126	5	17	4000	0.498	23	190
B17-4	23	126	5	17	5000	0.498	23	190
B17-5	23	126	5	17	$2^{31}-1$	0.498	23	190
B18-1	24	138	5	18	2000	0.5	24	200
B18-2	24	138	5	18	3000	0.5	24	200
B18-3	24	138	5	18	4000	0.5	24	200
B18-4	24	138	5	18	5000	0.5	24	200
B18-5	24	138	5	18	$2^{31}-1$	0.5	24	200
B19-1	25	150	5	19	2000	0.5	25	210
B19-2	25	150	5	19	3000	0.5	25	210
B19-3	25	150	5	19	4000	0.5	25	210
B19-4	25	150	5	19	5000	0.5	25	210
B19-5	25	150	5	19	$2^{31}-1$	0.5	25	210

Table 5.3: Problem Set B

Pr	$\|V_c\|$	$\|E_c\|$	$\|S\|$	$\|R\|$	B (Budget)	G_c Density	$\|V_{trust}\|$	$\|E_{trust}\|$
C20-1	26	162	5	20	2000	0.498	26	220
C20-2	26	162	5	20	3000	0.498	26	220
C20-3	26	162	5	20	4000	0.498	26	220
C20-4	26	162	5	20	5000	0.498	26	220
C20-5	26	162	5	20	$2^{31}-1$	0.498	26	220
C30-1	36	315	5	30	2000	0.5	36	320
C30-2	36	315	5	30	3000	0.5	36	320
C30-3	36	315	5	30	4000	0.5	36	320
C30-4	36	315	5	30	5000	0.5	36	320
C30-5	36	315	5	30	$2^{31}-1$	0.5	36	320
C40-1	46	517	5	40	2000	0.4995169	46	420
C40-2	46	517	5	40	3000	0.4995169	46	420
C40-3	46	517	5	40	4000	0.4995169	46	420
C40-4	46	517	5	40	5000	0.4995169	46	420
C40-5	46	517	5	40	$2^{31}-1$	0.4995169	46	420
C50-1	56	770	5	50	2000	0.5	56	520
C50-2	56	770	5	50	3000	0.5	56	520
C50-3	56	770	5	50	4000	0.5	56	520
C50-4	56	770	5	50	5000	0.5	56	520
C50-5	56	770	5	50	$2^{31}-1$	0.5	56	520
C60-1	66	1072	5	60	2000	0.4997669	66	620
C60-2	66	1072	5	60	3000	0.4997669	66	620
C60-3	66	1072	5	60	4000	0.4997669	66	620
C60-4	66	1072	5	60	5000	0.4997669	66	620
C60-5	66	1072	5	60	$2^{31}-1$	0.4997669	66	620
C70-1	76	1425	5	70	2000	0.5	76	720
C70-2	76	1425	5	70	3000	0.5	76	720
C70-3	76	1425	5	70	4000	0.5	76	720
C70-4	76	1425	5	70	5000	0.5	76	720
C70-5	76	1425	5	70	$2^{31}-1$	0.5	76	720
C80-1	86	1827	5	80	2000	0.4998632	86	820
C80-2	86	1827	5	80	3000	0.4998632	86	820
C80-3	86	1827	5	80	4000	0.4998632	86	820
C80-4	86	1827	5	80	5000	0.4998632	86	820
C80-5	86	1827	5	80	$2^{31}-1$	0.4998632	86	820
C90-1	96	2280	5	90	2000	0.5	96	920
C90-2	96	2280	5	90	3000	0.5	96	920
C90-3	96	2280	5	90	4000	0.5	96	920
C90-4	96	2280	5	90	5000	0.5	96	920
C90-5	96	2280	5	90	$2^{31}-1$	0.5	96	920
C100-1	106	2782	5	100	2000	0.49991015	106	1020
C100-2	106	2782	5	100	3000	0.49991015	106	1020
C100-3	106	2782	5	100	4000	0.49991015	106	1020
C100-4	106	2782	5	100	5000	0.49991015	106	1020
C100-5	106	2782	5	100	$2^{31}-1$	0.49991015	106	1020

Table 5.4: Problem Set C

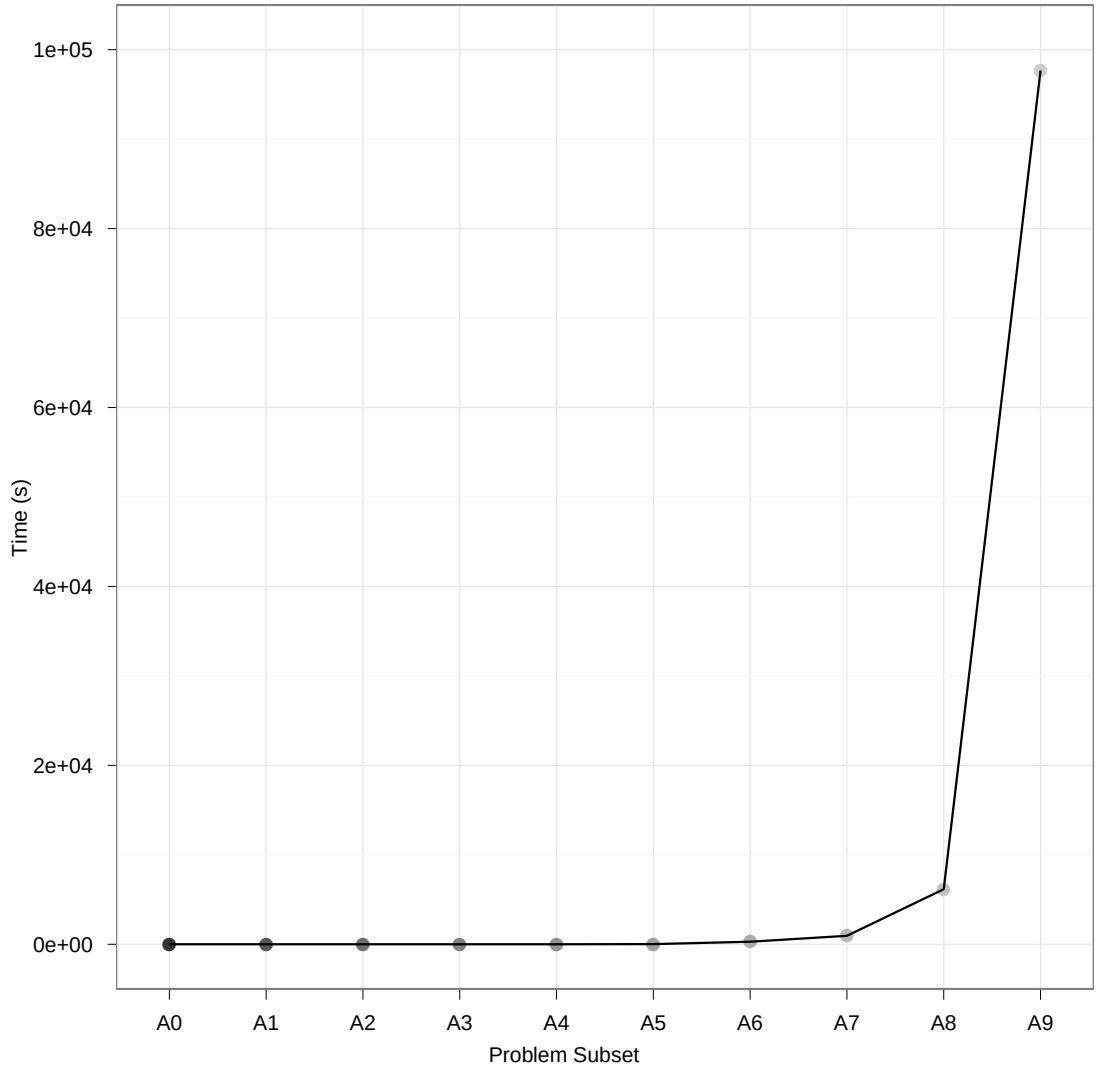

Figure 5.1: Average Execution Times of Exhaustive Search Results for Problem Set A

self-loop. The algorithm ran for one thousand iterations and the number of required edges was set such that the generated graphs had graph density approximately equal to 0.5. The connectivity graphs are simple graphs where no router vertex has only one adjacent edge. This ensures that all routers in R can be considered as a candidate router of the solution PST, as any router with only one adjacent edge can only be a leaf of a PST, which is contradictory to the definition of a PST.

5.3.3 Trust Graph and Trust Functions

Analysis of social networks shows that they have both scale-free and small-word properties (Guha et al., 2004), however, there is some initial research to suggest that the Pareto-lognormal distribution may provide a better fit, as the power-law model over estimates the number of nodes with high degree (Sala et al., 2010). Experiments conducted on the

Advogato trust data set[8] support the theory of an over-estimation of high degree nodes by the power-law distribution. Figure 5.2 shows the complementary cumulative distribution function (CCDF) of the out-degree of vertices in the Advogato trust graph. As the graph is a log-log plot, the CCDF would fit a Pareto distribution only if it were a straight line, however this is not the case. The Pareto distribution where $\alpha = 1.621746$ is shown on the figure and is the best-fit for the empirical data as given by the R library, igraph[9]. Further research on the suitability of the Pareto-lognormal distribution is required, so in this work it is assumed that the trust graph has a high clustering property (small-world) and fits the power-law model.

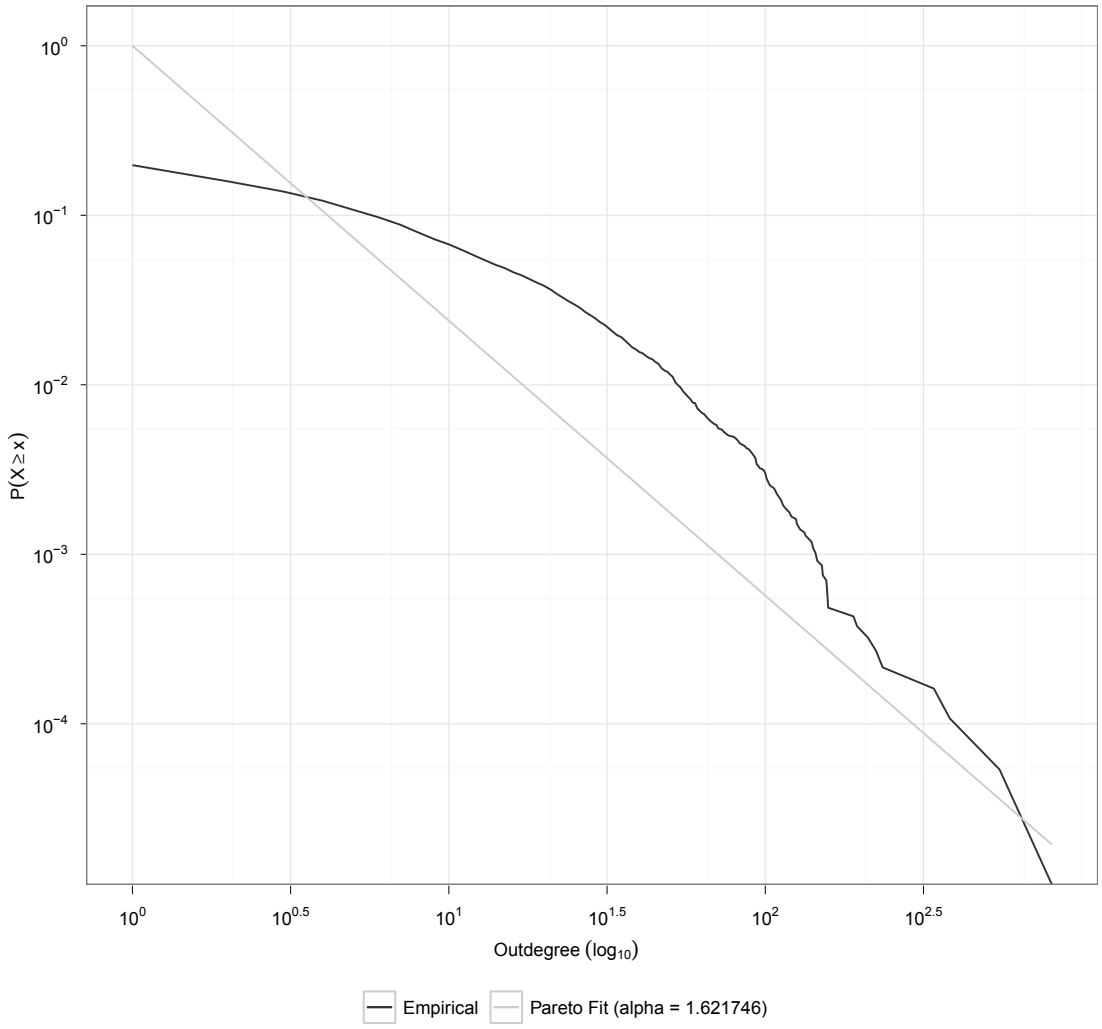

Figure 5.2: Power-law fit to Empirical CCDF of Advogato Out-degree

To generate trust graphs with both power-law and high clustering properties, the model described by Klemm and Eguíluz is used (Klemm and Eguiluz, 2002). It requires

[8] http://www.trustlet.org/datasets/advogato/
[9] http://igraph.sourceforge.net/

two parameters, μ and m, where $\mu \in [0, 1]$ is a tunable parameter such that $\mu = 0$ gives a high clustering model and $\mu = 1$ is the Barabasi-Albert model (scale-free) (Barabási et al., 2000), and m is the number of active nodes at any given iteration. The initial graph has m vertices and is complete with each node in an active state. At any given iteration only m vertices can be active, and it is to these vertices that a new vertex, n, is connected to at each iteration. The algorithm continues by determining with random probability, proportional to the vertex degree (i.e. preferential attachment), if each newly added edge (n, y), where y is an active vertex, is replaced by an edge (n, x) where x is a randomly chosen vertex that is a member of the graph's vertex set and maybe in either the active or inactivate state. An iteration of the algorithm ends with r being set to the active state and one of the existing active vertices being deactivated with probability that is inversely proportional to the ratio of a given active vertex's degree to the sum of the degrees at all active vertices.

To generate the trust graph for a given problem instance, $\mu = 0.1$ and $m = 5$ were selected. $\mu = 0.1$ is chosen for two reasons: the experimental results by Klemm and Eguíluz show that $\mu = 0.1$ gives short average path lengths, which is a property of scale-free networks, and that they grow logarithmically with the graph size, a property of small-world networks; the clustering coefficient is shown to be near to constant where $0 < \mu \ll 1$. Each edge represents a trust relationship, but given two adjacent vertices, a and b, the trust value of their relationship may not be identical, that is $\tau(a, b) \neq \tau(b, a)$, so the undirected edge is replaced by two directed and weighted edges. Due to the absence of analytical research on directed trust graphs with real-valued edge weights representing trust, the trust values of the relationships encapsulated by the trust graph were chosen at random from a normal distribution with mean $\mu = 0.5$ and variance $\sigma^2 = (0.5/3)^2$ where any random values greater than 1 and less than 0 were discarded. Note that trust values must be between 0 and 1 inclusive, as this is a requirement inherited from the analytical leximin function. The user-set parameter of this function is $\Delta = 0.001$.

5.4 Evaluation of Tabu Search Algorithms

5.4.1 Problem Set A

This test data set comprises of problems where $1 \leq |R| \leq 9$ and $|S| = 5$. For all solutions found by the tabu search algorithms, the relative errors to the optimal solutions found by the exhaustive search algorithm (table 5.1) are given. The following subsections provide

an analysis of the results for the tabu search algorithm with respect to each combination of penalty function, PST selection policy, and diversification technique. The section ends with a summary of the conclusions drawn from the results tables and the averages for the properties under evaluation (table 5.9).

5.4.1.1 Static Penalty Function and Best PST Selection

The results for the tabu search algorithm with static penalty function and best PST selection policy are given in table 5.5 with problems where no optimal solution exists omitted. They show that the optimal solution is found for all problems in problem subsets A1, A2, A3, A4, A7, and A9, regardless of the diversification strategy used. In addition, optimal solutions are found for subset A8 when the SPT diversification policy is used. Of the 35 problems in the test data set, 30 optimal solutions are found by the algorithm when SPT diversification is used and 28 are found with Takahashi-Matsuyama diversification.

The problem subsets A1, A2, A3, A4, A7, and A9, each contain problems whose optimal solutions have identical trust and overhead values, however, this is not the case for A5, A6, and A8, where the algorithm finds non-optimal solutions. The algorithm, using either diversification strategy, finds the optimal solution for A5-2, A6-3, and A8-3, the problems with the strictest budget in their subsets where an optimal solution exists. For A5-3, A5-4, and A5-5, the diversification policies yield the same solutions with no relative error in the trust values and 0.1202 relative error in the overhead values. For A6-4 and A6-5, the solutions are identical for both diversification strategies with a relative error of 5×10^{-7} in the trust value and 0.1287 in the overhead values. Problems A8-4 and A8-5 are the only ones where there is a difference in the solutions between the two diversification strategies. The use of SPT diversification leads to the algorithm finding the optimal solutions for A8-3, A8-4 and A8-5, but the Takahashi-Diversification does not, although there is only a small relative error of 1×10^{-6} in the trust value and 0.00928 in the overhead value. With reference to table 5.22, the initial solutions found by the Takahashi-Matsuyama and SPT heuristics are different for the problems A8-4 and A8-5, where the SPT solution is the less favourable. In practise, the solutions found for A8-4 and A8-5 with Takahashi-Matsuyama diversification could be considered preferable, as the relative trust error is negligible and the solutions are of a lower overhead value, however given the problem definition, they are not optimal. No solution found by the algorithm is over the given budget, B, for all problems.

There is little difference in the running times between the diversification policies until

problem A8-4. The differences in the average running times for A8-4 and A8-5 are 10.74 and 10.82 seconds slower with SPT diversification respectively. For problems in A9, the difference increases to approximately 21 seconds. For the problem subsets A1 to A4, the running times of the exhaustive search outperforms those of this tabu search algorithm regardless of the diversification strategy used. The exhaustive search running time for problem subset A1 is approximately 205 times faster, but this declines to 4.5 times faster for problem subset A4.

	Takahashi					SPT				
	PST		Rel. Error			PST		Rel. Error		
Pr	τ_T	O_T	η_τ	η_O	Sec	τ_T	O_T	η_τ	η_O	Sec
A1-2	0.0181	2398	-	-	2.90	0.0181	2398	-	-	2.92
A1-3	0.0181	2398	-	-	2.90	0.0181	2398	-	-	2.92
A1-4	0.0181	2398	-	-	2.89	0.0181	2398	-	-	2.92
A1-5	0.0181	2398	-	-	2.89	0.0181	2398	-	-	2.90
A2-1	0.0931	1850	-	-	11.77	0.0931	1850	-	-	11.93
A2-2	0.0931	1850	-	-	11.78	0.0931	1850	-	-	11.93
A2-3	0.0931	1850	-	-	11.77	0.0931	1850	-	-	11.94
A2-4	0.0931	1850	-	-	11.75	0.0931	1850	-	-	11.93
A2-5	0.0931	1850	-	-	11.79	0.0931	1850	-	-	11.90
A3-2	0.0224	2917	-	-	9.39	0.0224	2917	-	-	9.35
A3-3	0.0224	2917	-	-	9.40	0.0224	2917	-	-	9.34
A3-4	0.0224	2917	-	-	9.42	0.0224	2917	-	-	9.36
A3-5	0.0224	2917	-	-	9.39	0.0224	2917	-	-	9.33
A4-2	0.1855	2224	-	-	9.06	0.1855	2224	-	-	9.03
A4-3	0.1855	2224	-	-	9.06	0.1855	2224	-	-	9.05
A4-4	0.1855	2224	-	-	9.04	0.1855	2224	-	-	9.02
A4-5	0.1855	2224	-	-	9.01	0.1855	2224	-	-	8.99
A5-2	0.0542	2262	-	-	9.13	0.0542	2262	-	-	9.09
A5-3	0.0812	3580	-	0.1202	5.84	0.0812	3580	-	0.1202	5.81
A5-4	0.0812	3580	-	0.1202	5.82	0.0812	3580	-	0.1202	5.80
A5-5	0.0812	3580	-	0.1202	5.80	0.0812	3580	-	0.1202	5.79
A6-3	0.0360	3846	-	-	39.50	0.0360	3846	-	-	39.54
A6-4	0.0360	3846	5×10^{-7}	0.1287	41.45	0.0360	3846	5×10^{-7}	0.1287	41.41
A6-5	0.0360	3846	5×10^{-7}	0.1287	42.91	0.0360	3846	5×10^{-7}	0.1287	42.86
A7-3	0.0692	3570	-	-	88.41	0.0692	3570	-	-	88.17
A7-4	0.0692	3570	-	-	88.74	0.0692	3570	-	-	88.61
A7-5	0.0692	3570	-	-	88.77	0.0692	3570	-	-	88.52
A8-3	0.0031	3657	-	-	20.66	0.0031	3657	-	-	24.12
A8-4	0.0031	3657	1×10^{-6}	0.0928	20.71	0.0031	4031	-	-	31.45
A8-5	0.0031	3657	1×10^{-6}	0.0928	20.70	0.0031	4031	-	-	31.52
A9-1	0.2184	1885	-	-	48.98	0.2184	1885	-	-	70.19
A9-2	0.2184	1885	-	-	48.95	0.2184	1885	-	-	70.05
A9-3	0.2184	1885	-	-	48.94	0.2184	1885	-	-	70.23
A9-4	0.2184	1885	-	-	48.90	0.2184	1885	-	-	70.33
A9-5	0.2184	1885	-	-	48.92	0.2184	1885	-	-	70.33

Table 5.5: Solutions for Problem Set A using Tabu Search with Static Penalty and Best PST Selection

5.4.1.2 Static Penalty Function and Adaptive PST Selection

The results for problem set A (table 5.6) are identical to those of the tabu search with static penalty and best PST selection, for both diversification strategies. It is likely that the absence of any improvement is due to: the initial solution being optimal for some problem subsets (see table 5.22); and the small problem sizes with respect to $|R|$ and consequently the size of the connectivity graph and the number of PSTs (i.e. elements of the search space).

The running times for problem subsets A1 to A4 are similar to those of the tabu search with static penalty and best PST selection policy for both diversification strategies and they are similarly inferior to those of the exhaustive search. For problem subset A6, the use of the adaptive PST selection policy results in slower running times (a little over four times slower), but for problem subsets A7, A8, and A9, the opposite is true.

5.4.1.3 NFT Penalty Function and Best PST Selection

The static penalty function penalises PSTs found by the tabu search that are over budget by decreasing their objective value to 50% of the solution's trust value. Such an approach may prove to be disadvantageous, as a move to a worse current solution may eventually lead to finding a better overall solution, but the static penalty discourages this by favouring PSTs that are within budget. The NFT penalty function attempts to encourage exploration of the search space in a manner that is dependent upon the previously found solutions. It penalises solutions that are over budget to a lesser extent (i.e. within some feasible threshold) if the solutions in the algorithm's memory structures have been under budget, and more so if the opposite is true. According to its authors, such a technique should be expected to yield improvements to the results in comparison to a static penalty function (Kulturel-Konak et al., 2004), but the results for this particular problem set (table 5.7) show otherwise. The algorithm with Takahashi-Matsuyama diversification finds 27 optimal solutions and with SPT diversification, 29 optimal solutions are found, one less in each case than the algorithm using the static penalty function with identical diversification strategy.

Unlike the tabu search utilising the static penalty function, the tabu search with NFT penalty function, best PST selection and Takahashi-Matsuyama diversification finds a non-optimal solution for problem A5-2, which has high relative errors, 0.4976 in the trust value and 0.4129 in the overhead value, and is also over-budget. These are the highest errors of any non-optimal solution found for data set A, but this solution is the optimal

	Takahashi					SPT				
	PST		Rel. Error			PST		Rel. Error		
Pr	τ_T	O_T	η_τ	η_O	Sec	τ_T	O_T	η_τ	η_O	Sec
A1-2	0.0181	2398	-	-	3.01	0.0181	2398	-	-	2.89
A1-3	0.0181	2398	-	-	3.02	0.0181	2398	-	-	2.88
A1-4	0.0181	2398	-	-	3.01	0.0181	2398	-	-	2.89
A1-5	0.0181	2398	-	-	3.00	0.0181	2398	-	-	2.87
A2-1	0.0931	1850	-	-	8.44	0.0931	1850	-	-	8.42
A2-2	0.0931	1850	-	-	8.49	0.0931	1850	-	-	8.41
A2-3	0.0931	1850	-	-	8.40	0.0931	1850	-	-	8.40
A2-4	0.0931	1850	-	-	8.37	0.0931	1850	-	-	8.35
A2-5	0.0931	1850	-	-	8.36	0.0931	1850	-	-	8.36
A3-2	0.0224	2917	-	-	11.12	0.0224	2917	-	-	8.91
A3-3	0.0224	2917	-	-	11.12	0.0224	2917	-	-	8.93
A3-4	0.0224	2917	-	-	11.03	0.0224	2917	-	-	8.88
A3-5	0.0224	2917	-	-	11.06	0.0224	2917	-	-	8.89
A4-2	0.1855	2224	-	-	7.28	0.1855	2224	-	-	7.08
A4-3	0.1855	2224	-	-	7.21	0.1855	2224	-	-	7.01
A4-4	0.1855	2224	-	-	7.20	0.1855	2224	-	-	7.02
A4-5	0.1855	2224	-	-	7.21	0.1855	2224	-	-	7.02
A5-2	0.0542	2262	-	-	13.63	0.0542	2262	-	-	13.67
A5-3	0.0812	3580	-	0.1202	8.26	0.0812	3580	-	0.1202	8.26
A5-4	0.0812	3580	-	0.1202	8.24	0.0812	3580	-	0.1202	8.24
A5-5	0.0812	3580	-	0.1202	8.22	0.0812	3580	-	0.1202	8.25
A6-3	0.0360	3846	-	-	139.19	0.0360	3846	-	-	139.87
A6-4	0.0360	3846	5×10^{-7}	0.1287	138.96	0.0360	3846	5×10^{-7}	0.1287	139.87
A6-5	0.0360	3846	5×10^{-7}	0.1287	127.22	0.0360	3846	5×10^{-7}	0.1287	127.98
A7-2	0.0692	3570	-	-	70.95	0.0692	3570	-	-	71.02
A7-3	0.0692	3570	-	-	72.92	0.0692	3570	-	-	72.75
A7-4	0.0692	3570	-	-	78.38	0.0692	3570	-	-	78.33
A8-3	0.0031	3657	-	-	9.77	0.0031	3657	-	-	9.30
A8-4	0.0031	3657	1×10^{-6}	0.0928	9.77	0.0031	4031	-	-	12.11
A8-5	0.0031	3657	1×10^{-6}	0.0928	9.82	0.0031	4031	-	-	12.13
A9-1	0.2184	1885	-	-	20.39	0.2184	1885	-	-	36.06
A9-2	0.2184	1885	-	-	14.69	0.2184	1885	-	-	29.70
A9-3	0.2184	1885	-	-	20.55	0.2184	1885	-	-	37.15
A9-4	0.2184	1885	-	-	20.49	0.2184	1885	-	-	37.19
A9-5	0.2184	1885	-	-	20.51	0.2184	1885	-	-	37.21

Table 5.6: Solutions for Problem Set A using Tabu Search with Static Penalty and Adaptive PST Selection

one for A5-3, A5-4 and A5-5, and peculiarly it is not found for these problems. All other optimal and non-optimal solutions are identical to those found by the static penalty tabu search algorithms. In comparison to the tabu search algorithm using the static penalty function, the best PST selection policy and the Takahashi-Matsuyama diversification, the running times are similar except for the problems in A8 which are on average 2.55, 2.56, and 2.55 times faster for problems A8-3, A8-4, and A8-5 respectively. As is the case for the tabu search with static penalty algorithms, the exhaustive search runs faster for problem subsets A1-A4.

With the SPT diversification strategy, better performance is observed than with the Takahashi-Matsuyama, as more optimal solutions are found and for the problem A5-2, the relative trust error is eliminated, the relative overhead error is reduced to 0.1698, and the solution found is not over budget. In problem subsets A6 and A8, the algorithm finds the optimal solutions where the budget B is 5000 and $2^{31} - 1$, but not for $B = 4000$. This is the opposite behaviour to that of the Takahashi-Matsuyama diversification strategy, where the solution is found for the strictest budget problems, but is not found for problems with a larger budget. Two solutions found by the algorithm are over budget, problem A6-3 (14% error) and A8-3 (10% error). Compared to the tabu search with static penalty function, there is an improvement in problem subset A6 (two optimal solutions found rather one) and a deterioration in problem subset A8 (two optimal solution rather than all three). The running times, for the most part, are comparable to those of the tabu search algorithm with static penalty, best PST selection and SPT diversification, however there is an improvement when using the NFT penalty function for the problems in the subset A8 and the problem A5-2, and a deterioration for those in the subset A9. For A5-2, A8-3, A8-4, and A8-5 the running times are 3, 1.96, 2.56 and 2.54 times faster respectively. For A6-3, A6-4 and A6-5, the running times are 1.58, 2.13, and 2.05 times slower respectively. In comparison to the exhaustive search running times, the use of SPT diversification results in slower running times for problem subsets A1-A4 and problem A5-2. The latter is not true of with the use of the Takahashi-Matsuyama strategy.

5.4.1.4 NFT Penalty Function and Adaptive PST Selection

The tabu search with NFT penalty and adaptive PST selection policy finds identical solutions to those found by the tabu search with NFT penalty and best PST selection policy where both use the Takahashi-Matsuyama diversification strategy. Although the solutions found are identical, there are notable differences in the running times for the

	Takahashi					SPT				
	PST		Rel. Error			PST		Rel. Error		
Pr	τ_T	O_T	η_τ	η_O	Sec	τ_T	O_T	η_τ	η_O	Sec
A1-2	0.0181	2398	-	-	3.21	0.0181	2398	-	-	3.63
A1-3	0.0181	2398	-	-	3.05	0.0181	2398	-	-	3.62
A1-4	0.0181	2398	-	-	3.01	0.0181	2398	-	-	3.63
A1-5	0.0181	2398	-	-	3.03	0.0181	2398	-	-	3.61
A2-1	0.0931	1850	-	-	11.74	0.0931	1850	-	-	12.76
A2-2	0.0931	1850	-	-	11.84	0.0931	1850	-	-	12.79
A2-3	0.0931	1850	-	-	11.93	0.0931	1850	-	-	12.73
A2-4	0.0931	1850	-	-	11.87	0.0931	1850	-	-	12.66
A2-5	0.0931	1850	-	-	11.97	0.0931	1850	-	-	12.68
A3-2	0.0224	2917	-	-	7.76	0.0224	2917	-	-	9.65
A3-3	0.0224	2917	-	-	7.76	0.0224	2917	-	-	9.62
A3-4	0.0224	2917	-	-	7.73	0.0224	2917	-	-	9.66
A3-5	0.0224	2917	-	-	7.73	0.0224	2917	-	-	9.68
A4-2	0.1855	2224	-	-	8.90	0.1855	2224	-	-	11.64
A4-3	0.1855	2224	-	-	8.84	0.1855	2224	-	-	11.76
A4-4	0.1855	2224	-	-	8.84	0.1855	2224	-	-	11.80
A4-5	0.1855	2224	-	-	8.84	0.1855	2224	-	-	11.80
A5-2	0.0812	3196	0.4976	0.4129	9.87	0.0542	2646	-	0.1698	27.27
A5-3	0.0812	3580	-	0.1202	5.84	0.0812	3580	-	0.1202	6.11
A5-4	0.0812	3580	-	0.1202	5.81	0.0812	3580	-	0.1202	6.09
A5-5	0.0812	3580	-	0.1202	5.85	0.0812	3580	-	0.1202	6.07
A6-3	0.0360	3846	-	-	39.46	0.0360	4414	5×10^{-7}	0.1477	62.62
A6-4	0.0360	3846	5×10^{-7}	0.1287	41.22	0.0360	4414	-	-	88.34
A6-5	0.0360	3846	5×10^{-7}	0.1287	42.56	0.0360	4414	-	-	87.68
A7-3	0.0692	3570	-	-	87.22	0.0692	3570	-	-	94.41
A7-4	0.0692	3570	-	-	86.90	0.0692	3570	-	-	94.62
A7-5	0.0692	3570	-	-	87.48	0.0692	3570	-	-	94.77
A8-3	0.0031	3657	-	-	8.09	0.0031	4031	1×10^{-6}	0.1023	12.30
A8-4	0.0031	3657	1×10^{-6}	0.0928	8.08	0.0031	4031	-	-	12.30
A8-5	0.0031	3657	1×10^{-6}	0.0928	8.09	0.0031	4031	-	-	12.40
A9-1	0.2184	1885	-	-	50.32	0.2184	1885	-	-	69.85
A9-2	0.2184	1885	-	-	50.36	0.2184	1885	-	-	69.54
A9-3	0.2184	1885	-	-	50.69	0.2184	1885	-	-	69.95
A9-4	0.2184	1885	-	-	50.60	0.2184	1885	-	-	69.64
A9-5	0.2184	1885	-	-	50.26	0.2184	1885	-	-	69.62

Table 5.7: Solutions for Problem Set A using Tabu Search with NFT Penalty and Best PST Selection

problem subsets A6 and A9. For problem subset A6, there is a deterioration in the running times with the adaptive policy resulting in problems A6-3, A6-4 and A6-5, running on average 3.54, 3.39 and 3.00 times slower. The opposite effect is observed for A9, with the adaptive policy resulting in faster running times for the problems in this subset by at best, 30% (A9-2).

The tabu search with NFT penalty function, best PST selection policy, and SPT diversification found six non-optimal solutions, but when the adaptive PST selection is used, five non-optimal solutions are found, the improvement being in the solution to problem A6-3. The number of over budget solutions remains the same. Comparing the best and adaptive PST policies for the tabu search algorithm utilising the NFT penalty function and the Takahashi-Matsuyama diversification strategy, the adaptive PST selection policy gives three less non-optimal solutions, but the running times for all problems are slower. The adaptive policy has an adverse effect on the solution found for problem A5-3, where the relative error in the overhead value increases to 0.4129 and the relative error in the trust value is 0.4976. The running times for A6 and A9 differ significantly when compared to those of the tabu search with NFT penalty, best PST selection and SPT diversification. For problems A6-3, A6-4, and A6-5, the running times are 2.84, 2.34, and 2.60 times slower on average. For all problems in the subset A9, there is an improvement in the running times.

5.4.1.5 Summary

Table 5.9 shows that the static penalty function outperforms the NFT penalty function with respect to the number of non-optimal and over-budget solutions found, the average trust error of the non-optimal solutions, and the average overhead error of the non-optimal solutions. For each algorithm, the static penalty function variant finds more optimal solutions than those that use the NFT penalty function, except when the adaptive PST selection policy and SPT diversification are used, where they find the same number. All algorithms using the static penalty function find no over-budget solutions, but for the NFT penalty function this is not the case, as in conjunction with the Takahashi-Matsuyama diversification strategy, one over budget solution is found, and with SPT diversification, two over budget solutions are found. The average trust error is less for all static penalty tabu search algorithms when compared to the algorithm with identical properties other than the penalty function. The algorithms using the NFT penalty function, other than the variant with best PST selection policy and SPT diversification, have average trust

	Takahashi					SPT				
	PST		Rel. Error			PST		Rel. Error		
Pr	τ_T	O_T	η_τ	η_O	Sec	τ_T	O_T	η_τ	η_O	Sec
A1-2	0.0181	2398	-	-	2.90	0.0181	2398	-	-	3.06
A1-3	0.0181	2398	-	-	2.88	0.0181	2398	-	-	3.06
A1-4	0.0181	2398	-	-	2.88	0.0181	2398	-	-	3.04
A1-5	0.0181	2398	-	-	2.90	0.0181	2398	-	-	3.03
A2-1	0.0931	1850	-	-	8.43	0.0931	1850	-	-	8.74
A2-2	0.0931	1850	-	-	8.41	0.0931	1850	-	-	8.81
A2-3	0.0931	1850	-	-	8.37	0.0931	1850	-	-	8.77
A2-4	0.0931	1850	-	-	8.36	0.0931	1850	-	-	8.75
A2-5	0.0931	1850	-	-	8.36	0.0931	1850	-	-	8.73
A3-2	0.0224	2917	-	-	9.02	0.0224	2917	-	-	11.32
A3-3	0.0224	2917	-	-	9.04	0.0224	2917	-	-	11.28
A3-4	0.0224	2917	-	-	9.06	0.0224	2917	-	-	11.33
A3-5	0.0224	2917	-	-	9.11	0.0224	2917	-	-	11.19
A4-2	0.1855	2224	-	-	7.47	0.1855	2224	-	-	9.47
A4-3	0.1855	2224	-	-	7.35	0.1855	2224	-	-	9.40
A4-4	0.1855	2224	-	-	7.30	0.1855	2224	-	-	9.32
A4-5	0.1855	2224	-	-	7.30	0.1855	2224	-	-	9.36
A5-2	0.0812	3196	0.4976	0.4129	13.64	0.0812	3196	0.4976	0.4129	14.75
A5-3	0.0812	3580	-	0.1202	8.30	0.0812	3580	-	0.1202	8.35
A5-4	0.0812	3580	-	0.1202	8.23	0.0812	3580	-	0.1202	8.33
A5-5	0.0812	3580	-	0.1202	8.25	0.0812	3580	-	0.1202	8.28
A6-3	0.0360	3846	-	-	139.74	0.0360	3846	-	-	178.15
A6-4	0.0360	3846	5×10^{-7}	0.1287	139.72	0.0360	4414	-	-	206.86
A6-5	0.0360	3846	5×10^{-7}	0.1287	127.82	0.0360	4414	-	-	228.70
A7-3	0.0692	3570	-	-	70.76	0.0692	3570	-	-	78.84
A7-4	0.0692	3570	-	-	72.65	0.0692	3570	-	-	78.54
A7-5	0.0692	3570	-	-	77.97	0.0692	3570	-	-	86.64
A8-3	0.0031	3657	-	-	8.09	0.0031	4031	1×10^{-6}	0.1023	12.30
A8-4	0.0031	3657	1×10^{-6}	0.0928	8.08	0.0031	4031	-	-	12.30
A8-5	0.0031	3657	1×10^{-6}	0.0928	8.09	0.0031	4031	-	-	12.40
A9-1	0.2184	1885	-	-	21.14	0.2184	1885	-	-	36.14
A9-2	0.2184	1885	-	-	15.22	0.2184	1885	-	-	29.87
A9-3	0.2184	1885	-	-	21.08	0.2184	1885	-	-	37.14
A9-4	0.2184	1885	-	-	21.07	0.2184	1885	-	-	36.99
A9-5	0.2184	1885	-	-	21.07	0.2184	1885	-	-	36.95

Table 5.8: Solutions for Problem Set A using Tabu Search with NFT Penalty and Adaptive PST Selection

values that are five orders of magnitude greater than those of the tabu search algorithms using the static penalty function. Given these observations, the NFT penalty function gives no benefits. This may be because the technique is less beneficial when there are few constraints (Kulturel-Konak et al., 2004), or due to the choice of value for the amplification exponent, k, of the NFT penalty function. Increasing k would penalise the non-optimal PSTs that are beyond the NFT to a greater degree, but reduce the penalty applied to those solutions within the NFT.

Comparing the static penalty tabu search algorithms, it is evident that of the two diversification strategies, SPT outperforms Takahashi-Matsuyama (this is also true of the tabu search algorithms using the NFT penalty function, with respect to the number of non-optimal solutions, but not the number of over budget solutions). Both pairs of algorithms using the same diversification policies have the same results, so it can be concluded that the initial and diversified SPTs result in more optimal solutions being found, despite the fact that in some cases the initial SPT solution can be a worse PST than that given by the Takahashi-Matsuyama heuristic.

For the average running time, the use of the Takahashi-Matsuyama policy results in faster running times. Static penalty algorithms outperform the exhaustive search for problem subsets A5 and above. This is also true of the NFT tabu search algorithms except where the best PST selection and SPT diversification are used, where A5-2 is slower than the exhaustive search.

Algorithm	Non-optimal	Over-budget	Avg η_τ	Avg η_O	Avg Time
Static, Best, Takahashi	7	0	9×10^{-7}	0.1148	24.78
Static, Best, SPT	5	0	5×10^{-7}	0.1236	28.53
Static, Adaptive, Takahashi	7	0	9×10^{-7}	0.1148	26.27
Static, Adaptive, SPT	5	0	5×10^{-7}	0.1236	28.47
NFT, Best, Takahashi	8	1	0.0995	0.1521	25.06
NFT, Best, SPT	6	2	9×10^{-7}	0.1301	33.57
NFT, Adaptive, Takahashi	8	1	0.0995	0.1521	26.00
NFT, Adaptive, SPT	5	2	0.2487	0.1752	35.72

Table 5.9: Average Results Overview for Problem Set A

5.4.2 Problem Set B

This test data set has the same number of subscribers, $|S| = 5$, as problem set A, but the cardinality of R is greater, $10 \leq |R| \leq 19$. All other parameters used in the generation of this problem set are identical to those used to generate problem set A. No relative errors are given as it was impractical to run the exhaustive search algorithm on this problem set.

The last column in each table of results for problem set B contains SPT where the SPT solution is better than the Takahashi-Matsuyama one, Tak where the opposite is true, N/A where the solutions found using both diversification policies are over-budget, or '-' where the solutions found are identical and neither are over budget.

5.4.2.1 Static Penalty Function and Best PST Selection

The results of the application of this algorithm to the problem set B are given in table 5.10. Although in every problem instance, SPT diversification results in a slower running time (for some problems by an order of magnitude difference) than the Takahashi-Matsuyama diversification. The use of the former gives better solutions for eleven problem instances. This follows from the tabu search results for problem set A where the use of SPT diversification gives more optimal solutions. The number of problems where no solution is within budget is 12.

The longest average running time (748.67s) is that of problem A16-5 where the SPT diversification strategy is used. This is longer than any of the average running times for problem set A, but it is less than that of the exhaustive search for problem subset A7.

5.4.2.2 Static Penalty Function and Adaptive PST Selection

The use of the adaptive policy gives rise to a reduction in the number of problem instances where the SPT diversification gives a better solution than the Takahashi-Matsuyama diversification. Two problems, B16-4 and B16-5, have a better solution when SPT diversification is used and one problem, B14-2, when Takahashi-Matsuyama diversification is used. This change is due to the adaptive policy resulting in an improvement in the solutions found with the Takahashi-Matsuyama diversification, and not a deterioration in those found with SPT diversification. For Takahashi-Matsuyama, improvements are seen in B10-2, B12-3, B12-4, B12-5, B14-3, B16-4, B16-5, B17-2, B17-3, B17-5 when adaptive PST selection rather than the best PST selection policy. The number of problems where no solution within budget is found is 12, all of which are for the same problems as the over budget PSTs found by the tabu search with static penalty function and best PST selection.

As with the best PST selection policy, the use of the adaptive PST selection shows that running times are slower when SPT diversification is used. The extent of the difference is dependent on the problem subset. For problems in B9, SPT diversification is approximately seven times slower than that of Takahashi-Matsuyama, but the difference

	Takahashi			SPT			
	PST			PST			
Pr	τ_T	O_T	Sec	τ_T	O_T	Sec	Best
B10-1	0.1095	3592	30.88	0.1095	3592	102.42	N/A
B10-2	0.0850	2912	31.06	0.1095	2912	65.36	SPT
B10-3	0.1095	3592	45.59	0.1095	3592	127.72	-
B10-4	0.1095	3592	40.37	0.1095	3592	146.14	-
B10-5	0.1095	3592	40.41	0.1095	3592	145.93	-
B11-1	0.0722	3495	30.49	0.0722	3495	592.05	N/A
B11-2	0.0722	3495	30.63	0.0722	3495	592.70	N/A
B11-3	0.0722	3495	74.09	0.0722	3495	379.10	-
B11-4	0.0722	3495	69.61	0.0722	3495	144.23	-
B11-5	0.0722	3495	69.60	0.0722	3495	144.17	-
B12-1	0.0394	3002	35.71	0.0395	3682	89.38	N/A
B12-2	0.0394	2944	12.93	0.0394	2944	30.34	-
B12-3	0.0394	3753	44.90	0.0395	3812	74.98	SPT
B12-4	0.0394	3002	13.44	0.0395	3812	33.08	SPT
B12-5	0.0394	3002	13.45	0.0395	3812	34.24	SPT
B13-1	0.0770	1479	10.43	0.0770	1479	11.29	-
B13-2	0.0770	1479	10.43	0.0770	1479	11.27	-
B13-3	0.0770	1479	10.45	0.0770	1479	11.27	-
B13-4	0.0770	1479	10.46	0.0770	1479	11.25	-
B13-5	0.0770	1479	10.44	0.0770	1479	11.28	-
B14-1	0.0705	3359	15.13	0.0705	3743	311.10	N/A
B14-2	0.0628	2791	15.52	0.0705	3743	311.15	Tak.
B14-3	0.0705	3390	18.09	0.0705	3587	142.02	SPT
B14-4	0.0705	3774	30.75	0.0705	3587	145.03	-
B14-5	0.0705	3774	30.83	0.0705	3587	85.70	-
B15-1	0.1368	3977	9.96	0.0719	4493	28.51	N/A
B15-2	0.1368	3977	9.97	0.0719	4493	28.43	N/A
B15-3	0.1368	3813	12.92	0.1368	3603	31.38	-
B15-4	0.1368	3813	12.48	0.1368	3603	45.72	-
B15-5	0.1368	3813	12.47	0.1368	3603	44.92	-
B16-1	0.0177	4414	85.29	0.0178	5236	201.69	N/A
B16-2	0.0177	4414	85.36	0.0178	5236	201.92	N/A
B16-3	0.0177	4414	85.51	0.0178	5236	201.92	N/A
B16-4	0.0177	4930	88.11	0.0178	4852	200.25	SPT
B16-5	0.0177	5418	46.12	0.0178	5752	748.67	SPT
B17-1	0.1124	2079	29.12	0.1124	2970	117.99	N/A
B17-2	0.1124	2079	29.09	0.1124	2970	45.49	SPT
B17-3	0.1124	2079	29.05	0.1124	3912	49.69	SPT
B17-4	0.1124	2079	29.08	0.1124	3912	58.93	SPT
B17-5	0.1124	2079	29.05	0.1124	3912	66.49	SPT
B18-1	0.2218	1829	20.60	0.2218	1829	30.11	-
B18-2	0.2218	1829	20.64	0.2218	1829	30.09	-
B18-3	0.2218	1829	20.62	0.2218	1829	30.10	-
B18-4	0.2218	1829	20.69	0.2218	1829	30.13	-
B18-5	0.2218	1829	20.62	0.2218	1829	30.16	-
B19-1	0.1457	1720	1.75	0.1457	1720	14.14	-
B19-2	0.1457	1720	1.73	0.1457	1720	14.11	-
B19-3	0.1457	1720	1.72	0.1457	1720	14.62	-
B19-4	0.1457	1720	1.72	0.1457	1720	14.61	-
B19-5	0.1457	1720	1.72	0.1457	1720	14.62	-

Table 5.10: Solutions for Problem Set B using Tabu Search with Static Penalty and Best PST Selection

for problems in B13 is less than a second. The times for the problems in B16 are considerably higher than all others for both diversification strategies, where there is an order of magnitude difference between the running times of the problem in B16 and the non-B16 problem with the longest running time. However, the longest running times, B16-5 at 1683.23s and 3562.55s, for the Takahashi-Matsuyama and SPT diversifications respectively are a significant improvement over what would be expected from an exhaustive search for these problems, as both are faster than the exhaustive search time for problem subset A9. Although the use of the adaptive PST selection policy leads to longer running times for problem subset B16, there is an improvement in the quality of the solutions. Despite this, the solutions found with Takahashi-Matsuyama diversification still remain inferior to those found when SPT diversification is used.

5.4.2.3 NFT Penalty Function and Best PST Selection

The results for the tabu search algorithm using the NFT penalty function and best PST selection policy are given in table 5.12. When implemented with SPT diversification, it finds identical solutions (excluding those where the solution found is over-budget) to those found where the static penalty function and the same PST selection and diversification policies are used. For Takahashi-Matsuyama diversification, there are differences in the solutions found for the problems B10-2, B12-2 and B14-2 between NFT and static penalty function variants using the best PST selection and either diversification policies. Each PST found for these problems has a higher trust value, but the overheads are all over-budget.

The number of solutions where the SPT diversification finds a better solution than Takahashi diversification is 12, one more than the tabu search with static penalty, best PST selection and SPT diversification (B12-2 degrades for Takahashi-Matsuyama). As with problem set A, there is an increase in the number of over-budget solutions found when compared to the algorithms using the static penalty function. The number of problems where both SPT and Takahashi solutions are over-budget is 13, one more than the tabu search with static penalty, best PST selection and SPT diversification (B14-2 degrades for Takahashi-Matsuyama).

There is little difference between the running times when the algorithm is compared to that with the static penalty and best PST selection for both diversification strategies, a similar finding to that of problem set A results. The running times for the tabu search with NFT penalty function and best PST selection policy are slower for all problems where the SPT diversification strategy is used instead of Takahashi-Matsuyama, a recurring theme.

	Takahashi			SPT			
	PST			PST			
Pr	τ_T	O_T	Sec	τ_T	O_T	Sec	Best
B10-1	0.1095	3592	29.05	0.1095	3592	76.74	N/A
B10-2	0.1095	2912	32.62	0.1095	2912	78.90	-
B10-3	0.1095	3592	41.03	0.1095	3592	96.70	-
B10-4	0.1095	3592	56.93	0.1095	3592	123.66	-
B10-5	0.1095	3592	56.92	0.1095	3592	123.72	-
B11-1	0.0722	3495	13.22	0.0722	3495	374.69	N/A
B11-2	0.0722	3495	13.25	0.0722	3495	374.81	N/A
B11-3	0.0722	3495	26.98	0.0722	3495	143.87	-
B11-4	0.0722	3495	23.48	0.0722	3495	64.20	-
B11-5	0.0722	3495	23.51	0.0722	3495	64.57	-
B12-1	0.0395	3682	27.34	0.0395	3682	50.13	N/A
B12-2	0.0394	2944	27.17	0.0394	2944	49.74	-
B12-3	0.0395	3812	43.79	0.0395	3812	74.08	-
B12-4	0.0395	3812	35.60	0.0395	3812	61.27	-
B12-5	0.0395	3812	35.59	0.0395	3812	61.39	-
B13-1	0.0770	1479	11.83	0.0770	1479	12.32	-
B13-2	0.0770	1479	11.81	0.0770	1479	12.34	-
B13-3	0.0770	1479	11.80	0.0770	1479	12.39	-
B13-4	0.0770	1479	11.83	0.0770	1479	12.34	-
B13-5	0.0770	1479	11.81	0.0770	1479	12.36	-
B14-1	0.0705	3359	24.24	0.0705	3498	133.28	N/A
B14-2	0.0628	2791	25.42	0.0705	3498	133.28	Tak.
B14-3	0.0705	3774	41.15	0.0705	3774	116.23	-
B14-4	0.0705	3774	64.50	0.0705	3587	142.68	-
B14-5	0.0705	3774	68.37	0.0705	3587	117.68	-
B15-1	0.1368	3977	6.40	0.0867	3637	17.27	N/A
B15-2	0.1368	3977	6.39	0.0867	3637	17.29	N/A
B15-3	0.1368	3977	9.94	0.1368	3603	26.55	-
B15-4	0.1368	4493	11.56	0.1368	3603	46.34	-
B15-5	0.1368	4493	11.73	0.1368	3603	45.73	-
B16-1	0.0178	5236	1,154.96	0.0178	5236	1,825.15	N/A
B16-2	0.0178	5236	1,155.46	0.0178	5236	1,829.69	N/A
B16-3	0.0178	5236	1,155.62	0.0178	5236	1,830.59	N/A
B16-4	0.0178	4852	1,403.36	0.0178	4852	1,435.04	SPT
B16-5	0.0178	5120	1,683.23	0.0178	5120	3,562.55	SPT
B17-1	0.1124	3426	24.31	0.1124	2970	111.84	N/A
B17-2	0.1124	2970	23.24	0.1124	2970	39.37	-
B17-3	0.1124	3426	24.78	0.1124	3912	44.66	-
B17-4	0.1124	3912	28.84	0.1124	3912	57.80	-
B17-5	0.1124	3912	28.83	0.1124	3912	65.30	-
B18-1	0.2218	1829	13.53	0.2218	1829	21.30	-
B18-2	0.2218	1829	13.57	0.2218	1829	21.28	-
B18-3	0.2218	1829	14.84	0.2218	1829	22.21	-
B18-4	0.2218	1829	14.79	0.2218	1829	22.21	-
B18-5	0.2218	1829	14.82	0.2218	1829	22.22	-
B19-1	0.1457	1720	1.93	0.1457	1720	14.30	-
B19-2	0.1457	1720	2.20	0.1457	1720	14.57	-
B19-3	0.1457	1720	2.14	0.1457	1720	15.25	-
B19-4	0.1457	1720	2.16	0.1457	1720	15.00	-
B19-5	0.1457	1720	2.14	0.1457	1720	14.97	-

Table 5.11: Solutions for Problem Set B using Tabu Search with Static Penalty and Adaptive PST Selection

The average times for Problem B16-5 is where the greatest difference is found with SPT diversification being close to 17 times slower.

5.4.2.4 NFT Penalty Function and Adaptive PST Selection

As with the static penalty tabu search algorithms, the use of an adaptive PST policy sees an improvement in quality of the solutions when the Takahashi-Matsuyama diversification policy is used. There are only two problems, B16-4 and B16-5, where the SPT diversification solution is better, compared to 12 where the best PST selection policy is used. No solution found with the Takahashi-Matsuyama diversification is better than that found by SPT diversification, as is also the case when the NFT penalty function and best PST selection is used.

The running times are similar to those of the algorithm with the static penalty function with adaptive PST selection irrespective of the diversification policy used. For all problems, the SPT diversification has longer running times than the Takahashi-Matsuyama strategy. Problem subset B16 has the longest running time with problem B16-5 being the longest (1700.79s —Takahashi-Matsuyama, 3578.68s —SPT), an order of magnitude greater than the next longest running non-B16 problem running time.

5.4.2.5 Summary

Table 5.14 shows the number of best solutions found, the number of over-budget solutions found and the average running time for each algorithm. Similar observations made for problem set A can also be made for this problem set with respect to the quality of the solutions found and the running time.

For each algorithm, the static penalty variant finds more best solutions and less over-budget ones than the NFT variant, except for where the best selection policy and SPT diversification are used, where they are both equal. Of the static algorithms, the tabu search with static penalty, adaptive PST selection, and SPT diversification finds the most number of best solutions, 38, but given the 12 over budget solutions, and the excessive running times of problems in B16, the tabu search with static penalty, best PST selection, and SPT diversification offers a better compromise between solution quality and running time. However, even for this algorithm the B16-5 problem has a running time of 12 minutes 28 seconds. The running times do not significantly differ between static and NFT penalty functions, but do so considerably between the best PST selection and adaptive selection policies, primarily due to the increased running times of problems in B16. The adaptive

	Takahashi			SPT			
	PST			PST			
Pr	τ_T	O_T	Sec	τ_T	O_T	Sec	Best
B10-1	0.0255	2564	17.82	0.1095	2912	65.99	N/A
B10-2	0.1095	3602	30.44	0.1095	2912	66.11	SPT
B10-3	0.1095	3592	44.63	0.1095	3592	129.26	-
B10-4	0.1095	3592	39.39	0.1095	3592	147.96	-
B10-5	0.1095	3592	39.37	0.1095	3592	148.06	-
B11-1	0.0722	3495	31.16	0.0501	3682	557.31	N/A
B11-2	0.0722	3495	30.95	0.0722	3260	557.48	N/A
B11-3	0.0722	3495	74.81	0.0722	3495	385.37	-
B11-4	0.0722	3495	71.17	0.0722	3495	146.39	-
B11-5	0.0722	3495	70.49	0.0722	3495	146.68	-
B12-1	0.0394	3002	36.13	0.0394	3002	58.19	N/A
B12-2	0.0394	3002	13.56	0.0394	2944	30.83	SPT
B12-3	0.0394	3753	45.25	0.0395	3812	76.59	SPT
B12-4	0.0394	3002	13.61	0.0395	3812	33.67	SPT
B12-5	0.0394	3002	13.71	0.0395	3812	34.64	SPT
B13-1	0.0770	1479	10.52	0.0770	1479	11.59	-
B13-2	0.0770	1479	10.56	0.0770	1479	11.58	-
B13-3	0.0770	1479	10.56	0.0770	1479	11.58	-
B13-4	0.0770	1479	10.55	0.0770	1479	11.59	-
B13-5	0.0770	1479	10.57	0.0770	1479	11.41	-
B14-1	0.0198	3222	14.84	0.0175	3390	238.91	N/A
B14-2	0.0705	3359	14.75	0.0285	3570	238.76	N/A
B14-3	0.0705	3390	17.55	0.0705	3587	141.91	SPT
B14-4	0.0705	3774	29.84	0.0705	3587	145.05	-
B14-5	0.0705	3774	29.92	0.0705	3587	85.44	-
B15-1	0.0438	3035	8.66	0.0167	3813	18.62	N/A
B15-2	0.0438	3035	8.71	0.0438	3035	18.65	N/A
B15-3	0.1368	3813	12.94	0.1368	3603	32.63	-
B15-4	0.1368	3813	12.48	0.1368	3603	47.26	-
B15-5	0.1368	3813	12.52	0.1368	3603	46.53	-
B16-1	0.0157	5340	85.48	0.0034	5916	201.60	N/A
B16-2	0.0157	5340	85.36	0.0034	5916	201.96	N/A
B16-3	0.0177	5478	85.47	0.0157	5778	202.22	N/A
B16-4	0.0177	4930	88.06	0.0178	4852	200.24	SPT
B16-5	0.0177	5418	45.97	0.0178	5752	747.87	SPT
B17-1	0.1124	2079	29.05	0.1124	2079	117.70	N/A
B17-2	0.1124	2079	29.14	0.1124	2970	45.88	SPT
B17-3	0.1124	2079	29.32	0.1124	3912	50.37	SPT
B17-4	0.1124	2079	29.32	0.1124	3912	59.04	SPT
B17-5	0.1124	2079	29.13	0.1124	3912	66.79	SPT
B18-1	0.2218	1829	20.90	0.2218	1829	30.12	-
B18-2	0.2218	1829	20.88	0.2218	1829	29.83	-
B18-3	0.2218	1829	20.93	0.2218	1829	29.87	-
B18-4	0.2218	1829	20.88	0.2218	1829	29.84	-
B18-5	0.2218	1829	21.07	0.2218	1829	29.93	-
B19-1	0.1457	1720	1.77	0.1457	1720	14.25	-
B19-2	0.1457	1720	1.75	0.1457	1720	14.25	-
B19-3	0.1457	1720	1.76	0.1457	1720	14.75	-
B19-4	0.1457	1720	1.74	0.1457	1720	14.76	-
B19-5	0.1457	1720	1.74	0.1457	1720	14.74	-

Table 5.12: Solutions for Problem Set B using Tabu Search with NFT Penalty and Best PST Selection

	Takahashi			SPT			
	PST			PST			
Pr	τ_T	O_T	Sec	τ_T	O_T	Sec	Best
B10-1	0.0255	2564	30.09	0.1095	2912	75.92	N/A
B10-2	0.1095	2912	32.54	0.1095	2912	79.37	-
B10-3	0.1095	3592	41.27	0.1095	3592	97.70	-
B10-4	0.1095	3592	57.30	0.1095	3592	124.23	-
B10-5	0.1095	3592	56.93	0.1095	3592	123.71	-
B11-1	0.0722	3495	13.40	0.0501	3682	372.94	N/A
B11-2	0.0722	3495	13.47	0.0722	3260	372.85	N/A
B11-3	0.0722	3495	27.55	0.0722	3495	142.94	-
B11-4	0.0722	3495	23.98	0.0722	3495	63.78	-
B11-5	0.0722	3495	23.77	0.0722	3495	64.21	-
B12-1	0.0394	3002	27.14	0.0394	3002	48.01	N/A
B12-2	0.0394	3002	27.39	0.0394	3002	47.92	N/A
B12-3	0.0395	3812	44.10	0.0395	3812	71.17	-
B12-4	0.0395	3812	35.50	0.0395	3812	59.23	-
B12-5	0.0395	3812	35.46	0.0395	3812	58.84	-
B13-1	0.0770	1479	11.54	0.0770	1479	12.32	-
B13-2	0.0770	1479	11.52	0.0770	1479	12.33	-
B13-3	0.0770	1479	11.54	0.0770	1479	12.43	-
B13-4	0.0770	1479	11.51	0.0770	1479	12.52	-
B13-5	0.0770	1479	11.52	0.0770	1479	12.55	-
B14-1	0.0198	3222	24.90	0.0175	3390	132.68	N/A
B14-2	0.0705	3359	25.28	0.0285	3570	132.87	N/A
B14-3	0.0705	3774	42.50	0.0705	3774	116.23	-
B14-4	0.0705	3774	66.66	0.0705	3587	142.62	-
B14-5	0.0705	3774	71.11	0.0705	3587	118.02	-
B15-1	0.0438	3035	5.52	0.0167	3813	15.64	N/A
B15-2	0.0438	3035	5.58	0.0438	3035	15.63	N/A
B15-3	0.1368	3977	9.99	0.1368	3603	27.30	-
B15-4	0.1368	4493	11.62	0.1368	3603	47.43	-
B15-5	0.1368	4493	11.76	0.1368	3603	47.32	-
B16-1	0.0157	5340	1,037.95	0.0034	5916	1,436.47	N/A
B16-2	0.0157	5340	1,039.50	0.0034	5916	1,434.91	N/A
B16-3	0.0177	5478	1,040.55	0.0157	5778	1,429.08	N/A
B16-4	0.0178	4852	1,418.33	0.0178	4852	1,439.70	SPT
B16-5	0.0178	5120	1,700.79	0.0178	5120	3,578.68	SPT
B17-1	0.1124	2079	22.58	0.1124	2079	112.19	N/A
B17-2	0.1124	2970	23.16	0.1124	2970	40.03	-
B17-3	0.1124	3426	24.69	0.1124	3912	45.15	-
B17-4	0.1124	3912	28.69	0.1124	3912	58.16	-
B17-5	0.1124	3912	28.66	0.1124	3912	66.39	-
B18-1	0.2218	1829	13.66	0.2218	1829	21.50	-
B18-2	0.2218	1829	13.76	0.2218	1829	21.55	-
B18-3	0.2218	1829	15.03	0.2218	1829	22.59	-
B18-4	0.2218	1829	14.99	0.2218	1829	22.78	-
B18-5	0.2218	1829	14.85	0.2218	1829	22.80	-
B19-1	0.1457	1720	1.95	0.1457	1720	14.73	-
B19-2	0.1457	1720	2.23	0.1457	1720	14.82	-
B19-3	0.1457	1720	2.24	0.1457	1720	15.42	-
B19-4	0.1457	1720	2.20	0.1457	1720	15.55	-
B19-5	0.1457	1720	2.19	0.1457	1720	15.41	-

Table 5.13: Solutions for Problem Set B using Tabu Search with NFT Penalty and Adaptive PST Selection

PST policy leads to a significant increase in the number of best solutions found for the Takahashi-Matsuyama diversification regardless of the other properties of the tabu search.

A direct comparison between the running times of the tabu search and exhaustive search algorithms can not be made for this problem set, as the running times for the exhaustive search would be excessive, thus rendering the experiments impractical. The average running time shown in table 5.14 are favourable when compared to the running times given for the exhaustive search in table 5.2, the longest average running time of 273.24s is approximately equal to that of the exhaustive search average running time of A6. In the worst case, the longest running times are those of B16-5 when the tabu search utilises that of the adaptive SPT policy, but is still faster than the exhaustive search for $|R| = 7$. Despite this, the running times for B16 for both PST selection policies are impractical for real-world problems.

In table 5.15, the algorithms that find the best solution (it is not possible to determine if these are also optimal due to running time of the exhaustive search) for each problem are given. Algorithm (1) is the static/best/takahashi tabu search, (2) is the static/best/spt, (3) is the static/adaptive/takahashi, (4) is the static/adaptive/spt, (5) is the nft/best/takahashi, (6) is the nft/best/spt, (7) is the nft/adaptive/takahashi, and (8) is the nft/adaptive/spt.

Algorithm	Best	Over-budget	Avg Time
Static, Best, Takahashi	28	11	29.02
Static, Best, SPT	37	12	120.96
Static, Adaptive, Takahashi	37	11	151.60
Static, Adaptive, SPT	38	12	273.24
NFT, Best, Takahashi	26	14	28.74
NFT, Best, SPT	37	12	115.44
NFT, Adaptive, Takahashi	35	13	145.41
NFT, Adaptive, SPT	37	13	249.47

Table 5.14: Average Results Overview for Problem Set B

5.4.3 Problem Set C

The results for problem set C show the least deviation in results. Of the static penalty tabu search algorithms, three find identical results: best PST selection and Takahashi diversification (table 5.16); best PST selection and SPT diversification (table 5.16); and Adaptive PST selection and Takahashi-Matsuyama diversification (table 5.17). The same is also true when then the tabu search algorithm utilises the NFT penalty function, the results for these algorithms can be found in tables 5.18 and 5.19. The tabu search al-

Pr	Algorithms
B10-1	
B10-2	(2) (3) (4) (6) (7) (8)
B10-3	(1) (2) (3) (4) (5) (6) (7) (8)
B10-4	(1) (2) (3) (4) (5) (6) (7) (8)
B10-5	(1) (2) (3) (4) (5) (6) (7) (8)
B11-1	
B11-2	
B11-3	(1) (2) (3) (4) (5) (6) (7) (8)
B11-4	(1) (2) (3) (4) (5) (6) (7) (8)
B11-5	(1) (2) (3) (4) (5) (6) (7) (8)
B12-1	
B12-2	(1) (2) (3) (4) (6)
B12-3	(2) (3) (4) (6) (7) (8)
B12-4	(2) (3) (4) (6) (7) (8)
B12-5	(2) (3) (4) (6) (7) (8)
B13-1	(1) (2) (3) (4) (5) (6) (7) (8)
B13-2	(1) (2) (3) (4) (5) (6) (7) (8)
B13-3	(1) (2) (3) (4) (5) (6) (7) (8)
B13-4	(1) (2) (3) (4) (5) (6) (7) (8)
B13-5	(1) (2) (3) (4) (5) (6) (7) (8)
B14-1	
B14-2	(1) (3)
B14-3	(2) (3) (4) (6) (7) (8)
B14-4	(1) (2) (3) (4) (5) (6) (7) (8)
B14-5	(1) (2) (3) (4) (5) (6) (7) (8)
B15-1	
B15-2	
B15-3	(1) (2) (3) (4) (5) (6) (7) (8)
B15-4	(1) (2) (3) (4) (5) (6) (7) (8)
B15-5	(1) (2) (3) (4) (5) (6) (7) (8)
B16-1	
B16-2	
B16-3	
B16-4	(2) (4) (6) (8)
B16-5	(4) (8)
B17-1	
B17-2	(2) (3) (4) (6) (7) (8)
B17-3	(2) (3) (4) (6) (7) (8)
B17-4	(2) (3) (4) (6) (7) (8)
B17-5	(2) (3) (4) (6) (7) (8)
B18-1	(1) (2) (3) (4) (5) (6) (7) (8)
B18-2	(1) (2) (3) (4) (5) (6) (7) (8)
B18-3	(1) (2) (3) (4) (5) (6) (7) (8)
B18-4	(1) (2) (3) (4) (5) (6) (7) (8)
B18-5	(1) (2) (3) (4) (5) (6) (7) (8)
B19-1	(1) (2) (3) (4) (5) (6) (7) (8)
B19-2	(1) (2) (3) (4) (5) (6) (7) (8)
B19-3	(1) (2) (3) (4) (5) (6) (7) (8)
B19-4	(1) (2) (3) (4) (5) (6) (7) (8)
B19-5	(1) (2) (3) (4) (5) (6) (7) (8)

Table 5.15: Algorithm Finding Best Solutions for Problem Set B

gorithm with adaptive PST selection and SPT diversification, utilising either the static (table 5.17) or NFT penalty function (table 5.18), outperformes the other techniques by finding better solutions for problems D100-4 and D100-5, as show in table 5.21, where algorithm (1) is the static/best/takahashi tabu search, (2) is the static/best/spt, (3) is the static/adaptive/takahashi, (4) is the static/adaptive/spt, (5) is the nft/best/takahashi, (6) is the nft/best/spt, (7) is the nft/adaptive/takahashi, and (8) is the nft/adaptive/spt. While the combination of adaptive PST selection and SPT diversification yields more better solutions, its average running time is only third fastest for both static and NFT penalty functions. Unlike previous problem sets, the NFT penalty tabu search algorithms do not find more over budget solutions than the static ones, as all algorithms find six over budget PSTs for the same problems.

The running times are dependent on the PST selection policy and diversification strategies. For problem sets A and B, the use of the adaptive PST selection policy resulted in an increase in the running time for each tabu search algorithm where all other properties remain identical. For problem set C, the opposite behaviour is observed, as changing the PST selection policy from best to adaptive gives faster running times, a little over three seconds improvement when the Takahashi-Matsuyama diversification is used and 1.90 times faster when the diversification strategy is SPT on average. Although the PST selection policy has an impact on running times, the diversification strategy contributes to it to a much greater extent. The tabu search with static penalty and best PST selection is on average 9.8 times faster when the Takahashi-Matsuyama diversification is used, and for the tabu search with NFT penalty function and the same PST selection policy, use of the Takahashi-Matsuyama diversification gives a running time that is 9.3 times faster. For the tabu search with static penalty and adaptive PST policy, and for the tabu search with NFT penalty and adaptive PST policy, the running times when used in conjunction with the Takahashi-Matsuyama diversification strategy are 5.47 and 5.38 times faster respectively. The average running times of the tabu search with SPT diversification are elevated as a result of the increased running times for problem subsets C20, C40, C80 and C90, which are an order of magnitude slower than those of the Takahashi-Matsuyama diversification.

With respect to the number of best solution found, the static and NFT penalty tabu search algorithms with adaptive PST selection and SPT diversification, find the most best solutions for the problem set, but some of the running times are impractical for real-world use. For the static variant, the running times of problem subsets C20 and C40

are in the order of hundreds of seconds, with the longest running time being 643.89s for problem C20-4. For the NFT variant, the running times for C20 and C40 are similar with the longest running time being 626.41s for problem C20-4. These running times are impractical in real-world applications where the connectivity graph is subject to change due to churn or mobility, and likely to change within these lengths of time, so for practical applications and given the results for this problem set, that tabu search algorithms using Takahashi-Matsuyama diversification are preferable.

5.5 Takahashi-Matsuyama and SPT Heuristics

The Steiner tree rooted at the publisher p, spanning the set of subscribers S, and the shortest path tree rooted at p, where each $s \in S$ is a terminal node, are in the set of all possible PSTs of an MTPSTO problem. These trees are the initial solutions to the tabu search algorithms. In tables 5.22, 5.23, and 5.24, the trust and overhead values of the initial PSTs using these methods are given for the three problem sets. In addition, for problem set A, their relative error with respect to the trust and overhead values, but as this information is not available for problem sets B and C, the relative errors to the best solution found by the tabu search algorithms is given for each problem.

Both heuristics are naive, in the sense that neither uses the trust information nor the subscriptions of the subscribers to estimate or to seek to reduce overheads, perhaps by maximising subscription covering. Despite this, for problem subset C, solutions are found to C60 and C90 using the Takahashi-Matsuyama heuristic, and for C80 and C90 by the SPT heuristic. The connectivity graph for problem subset C90, has the publisher directly connected to all the subscribers, so both the heuristics find the most trusted tree, however for almost all problems in problem sets B and C, the relative errors are high. Problem subset C30 is of particular interest as the solution for C30-1 is not selected by any tabu search algorithm as the best solution, instead they find an over budget solution and despite the solution being the initial solution of the search. This is due to the penalty functions penalising the initial solution such that inferior ones are preferred. For small connectivity graph size, where the publisher is located nearer to the publishers, the use of either of these heuristics may be preferable to the tabu search and the exhaustive search given the solutions found and the running times for the smaller problems of problem set A, however this may not always be the case, as is shown by the relative error for the problem subset A1 when the Takahashi-Matsuyama heuristic is used.

	Takahashi			SPT			
	PST			PST			
Pr	τ_T	O_T	Sec	τ_T	O_T	Sec	Best
C20-1	0.1210	2948	71.76	0.1210	2948	493.05	N/A
C20-2	0.1210	2948	71.57	0.1210	2948	493.83	-
C20-3	0.1210	3254	70.10	0.1210	3934	498.79	-
C20-4	0.1210	3254	70.15	0.1210	3934	631.74	-
C20-5	0.1210	3254	70.08	0.1210	3934	489.76	-
C30-1	0.1329	2234	59.38	0.1329	2234	98.42	N/A
C30-2	0.1329	2234	95.97	0.1329	2234	237.14	-
C30-3	0.1329	2234	95.65	0.1329	2234	237.92	-
C30-4	0.1329	2234	95.77	0.1329	2234	237.95	-
C30-5	0.1329	2234	95.94	0.1329	2234	238.15	-
C40-1	0.0245	2564	58.11	0.0245	3132	518.75	N/A
C40-2	0.0245	2564	57.97	0.0245	2564	520.06	-
C40-3	0.0245	2564	57.90	0.0245	3132	519.93	-
C40-4	0.0245	2564	57.80	0.0245	3132	530.84	-
C40-5	0.0245	2564	57.84	0.0245	3132	530.01	-
C50-1	0.0124	2224	15.05	0.0124	2224	26.08	N/A
C50-2	0.0124	2224	15.03	0.0124	2224	26.11	-
C50-3	0.0124	2224	15.03	0.0124	2224	26.09	-
C50-4	0.0124	2224	15.03	0.0124	2224	26.06	-
C50-5	0.0124	2224	15.06	0.0124	2224	26.00	-
C60-1	0.0661	1630	9.26	0.0661	1630	40.16	-
C60-2	0.0661	1630	9.25	0.0661	1630	40.05	-
C60-3	0.0661	1630	9.22	0.0661	1630	39.88	-
C60-4	0.0661	1630	9.24	0.0661	1630	40.12	-
C60-5	0.0661	1630	9.24	0.0661	1630	40.08	-
C70-1	0.0381	2838	14.39	0.0381	2838	41.97	N/A
C70-2	0.0381	2838	14.39	0.0381	2838	41.98	-
C70-3	0.0381	2838	14.79	0.0381	2838	40.38	-
C70-4	0.0381	2838	14.80	0.0381	2838	40.46	-
C70-5	0.0381	2838	14.82	0.0381	2838	40.40	-
C80-1	0.1320	1962	9.89	0.1320	1962	103.20	-
C80-2	0.1320	1962	9.89	0.1320	1962	103.40	-
C80-3	0.1320	1962	9.88	0.1320	1962	103.48	-
C80-4	0.1320	1962	9.88	0.1320	1962	103.54	-
C80-5	0.1320	1962	9.86	0.1320	1962	103.64	-
C90-1	0.0354	1282	27.12	0.0354	1282	1,412.56	-
C90-2	0.0354	1282	27.09	0.0354	1282	1,412.63	-
C90-3	0.0354	1282	27.03	0.0354	1282	1,412.73	-
C90-4	0.0354	1282	27.10	0.0354	1282	1,414.15	-
C90-5	0.0354	1282	27.06	0.0354	1282	1,413.67	-
C100-1	0.0112	2968	22.91	0.0128	2788	47.25	N/A
C100-2	0.0128	2798	18.91	0.0128	2798	38.46	-
C100-3	0.0128	2798	19.24	0.0128	2798	37.31	-
C100-4	0.0128	2798	19.26	0.0128	2798	36.00	-
C100-5	0.0128	2798	20.39	0.0128	2798	35.87	-

Table 5.16: Solutions for Problem Set C using Tabu Search with Static Penalty and Best PST Selection

	Takahashi			SPT			
	PST			PST			
Pr	τ_T	O_T	Sec	τ_T	O_T	Sec	Best
C20-1	0.1210	2948	42.00	0.1210	2948	463.82	N/A
C20-2	0.1210	2948	41.97	0.1210	2948	456.55	-
C20-3	0.1210	3254	36.33	0.1210	3934	458.79	-
C20-4	0.1210	3254	33.76	0.1210	3934	643.89	-
C20-5	0.1210	3254	33.73	0.1210	3934	445.46	-
C30-1	0.1329	2234	57.19	0.1329	2234	91.39	N/A
C30-2	0.1329	2234	61.82	0.1329	2234	105.19	-
C30-3	0.1329	2234	72.58	0.1329	2234	101.00	-
C30-4	0.1329	2234	88.44	0.1329	2234	113.33	-
C30-5	0.1329	2234	84.46	0.1329	2234	120.65	-
C40-1	0.0245	2564	56.52	0.0245	3132	505.12	N/A
C40-2	0.0245	2564	60.04	0.0245	2564	512.88	-
C40-3	0.0245	2564	50.73	0.0245	3132	499.93	-
C40-4	0.0245	2564	50.77	0.0245	3132	505.03	-
C40-5	0.0245	2564	50.81	0.0245	3132	508.89	-
C50-1	0.0124	2224	18.96	0.0124	2224	32.27	N/A
C50-2	0.0124	2224	18.87	0.0124	2224	32.12	-
C50-3	0.0124	2224	18.70	0.0124	2224	31.83	-
C50-4	0.0124	2224	19.70	0.0124	2224	33.61	-
C50-5	0.0124	2224	19.96	0.0124	2224	33.95	-
C60-1	0.0661	1630	9.86	0.0661	1630	41.32	-
C60-2	0.0661	1630	9.98	0.0661	1630	41.03	-
C60-3	0.0661	1630	9.82	0.0661	1630	41.26	-
C60-4	0.0661	1630	9.89	0.0661	1630	41.32	-
C60-5	0.0661	1630	9.91	0.0661	1630	41.39	-
C70-1	0.0381	2838	30.00	0.0381	2838	61.08	N/A
C70-2	0.0381	2838	29.99	0.0381	2838	61.28	-
C70-3	0.0381	2838	46.44	0.0381	2838	81.15	-
C70-4	0.0381	2838	46.77	0.0381	2838	82.05	-
C70-5	0.0381	2838	45.85	0.0381	2838	80.47	-
C80-1	0.1320	1962	17.84	0.1320	1962	114.96	-
C80-2	0.1320	1962	13.54	0.1320	1962	109.63	-
C80-3	0.1320	1962	13.56	0.1320	1962	109.79	-
C80-4	0.1320	1962	13.55	0.1320	1962	109.82	-
C80-5	0.1320	1962	13.57	0.1320	1962	109.80	-
C90-1	0.0354	1282	11.56	0.0354	1282	131.76	-
C90-2	0.0354	1282	11.59	0.0354	1282	131.87	-
C90-3	0.0354	1282	11.59	0.0354	1282	132.09	-
C90-4	0.0354	1282	11.57	0.0354	1282	131.87	-
C90-5	0.0354	1282	11.57	0.0354	1282	131.71	-
C100-1	0.0112	2968	28.40	0.0128	2788	66.01	N/A
C100-2	0.0128	2798	23.02	0.0128	2798	51.20	-
C100-3	0.0128	2798	18.89	0.0128	2798	49.50	-
C100-4	0.0128	2798	19.64	0.0128	2978	53.15	SPT
C100-5	0.0128	2798	19.52	0.0128	2978	53.08	SPT

Table 5.17: Solutions for Problem Set C using Tabu Search with Static Penalty and Adaptive PST Selection

	Takahashi			SPT			
	PST			PST			
Pr	τ_T	O_T	Sec	τ_T	O_T	Sec	Best
C20-1	0.1210	2948	72.17	0.0198	2564	477.44	N/A
C20-2	0.1210	2948	71.80	0.1210	2948	477.92	-
C20-3	0.1210	3254	70.83	0.1210	3934	482.08	-
C20-4	0.1210	3254	70.29	0.1210	3934	611.87	-
C20-5	0.1210	3254	70.70	0.1210	3934	474.06	-
C30-1	0.1329	2234	57.40	0.1329	2234	97.70	N/A
C30-2	0.1329	2234	92.82	0.1329	2234	234.46	-
C30-3	0.1329	2234	92.88	0.1329	2234	235.38	-
C30-4	0.1329	2234	92.87	0.1329	2234	235.41	-
C30-5	0.1329	2234	92.85	0.1329	2234	235.40	-
C40-1	0.0245	2564	58.55	0.0045	2564	517.87	N/A
C40-2	0.0245	2564	58.58	0.0245	2564	518.35	-
C40-3	0.0245	2564	58.51	0.0245	3132	519.12	-
C40-4	0.0245	2564	58.32	0.0245	3132	529.66	-
C40-5	0.0245	2564	58.39	0.0245	3132	530.63	-
C50-1	0.0124	2224	16.09	0.0124	2224	26.90	N/A
C50-2	0.0124	2224	16.08	0.0124	2224	26.95	-
C50-3	0.0124	2224	16.09	0.0124	2224	26.92	-
C50-4	0.0124	2224	16.07	0.0124	2224	26.87	-
C50-5	0.0124	2224	16.08	0.0124	2224	26.88	-
C60-1	0.0661	1630	9.62	0.0661	1630	40.00	-
C60-2	0.0661	1630	9.59	0.0661	1630	40.05	-
C60-3	0.0661	1630	9.57	0.0661	1630	40.06	-
C60-4	0.0661	1630	9.58	0.0661	1630	40.03	-
C60-5	0.0661	1630	9.58	0.0661	1630	40.13	-
C70-1	0.0381	2838	14.19	0.0381	2838	41.83	N/A
C70-2	0.0381	2838	14.20	0.0381	2838	41.77	-
C70-3	0.0381	2838	14.61	0.0381	2838	40.21	-
C70-4	0.0381	2838	14.58	0.0381	2838	40.39	-
C70-5	0.0381	2838	14.59	0.0381	2838	40.22	-
C80-1	0.1320	1962	9.71	0.1320	1962	104.78	-
C80-2	0.1320	1962	9.66	0.1320	1962	105.21	-
C80-3	0.1320	1962	9.65	0.1320	1962	104.97	-
C80-4	0.1320	1962	9.65	0.1320	1962	105.23	-
C80-5	0.1320	1962	9.68	0.1320	1962	105.18	-
C90-1	0.0354	1282	27.45	0.0354	1282	1,418.56	-
C90-2	0.0354	1282	27.43	0.0354	1282	1,420.29	-
C90-3	0.0354	1282	27.47	0.0354	1282	1,418.59	-
C90-4	0.0354	1282	27.43	0.0354	1282	1,418.93	-
C90-5	0.0354	1282	27.52	0.0354	1282	1,421.42	-
C100-1	0.0038	2448	20.42	0.0031	2628	38.92	N/A
C100-2	0.0128	2798	19.21	0.0128	2798	37.90	-
C100-3	0.0128	2798	19.12	0.0128	2798	37.56	-
C100-4	0.0128	2798	19.21	0.0128	2798	36.22	-
C100-5	0.0128	2798	19.71	0.0128	2798	36.04	-

Table 5.18: Solutions for Problem Set C using Tabu Search with NFT Penalty and Best PST Selection

	Takahashi			SPT			
	PST			PST			
Pr	τ_T	O_T	Sec	τ_T	O_T	Sec	Best
C20-1	0.1210	2948	42.49	0.0198	2564	451.71	N/A
C20-2	0.1210	2948	42.31	0.1210	2948	444.22	-
C20-3	0.1210	3254	36.61	0.1210	3934	445.57	-
C20-4	0.1210	3254	34.08	0.1210	3934	626.41	-
C20-5	0.1210	3254	34.05	0.1210	3934	433.22	-
C30-1	0.1329	2234	57.16	0.1329	2234	93.44	N/A
C30-2	0.1329	2234	61.67	0.1329	2234	107.19	-
C30-3	0.1329	2234	72.13	0.1329	2234	103.09	-
C30-4	0.1329	2234	87.94	0.1329	2234	115.90	-
C30-5	0.1329	2234	84.20	0.1329	2234	122.99	-
C40-1	0.0245	2564	56.92	0.0045	2564	523.61	N/A
C40-2	0.0245	2564	60.52	0.0245	2564	529.70	-
C40-3	0.0245	2564	51.25	0.0245	3132	517.41	-
C40-4	0.0245	2564	51.34	0.0245	3132	522.35	-
C40-5	0.0245	2564	51.25	0.0245	3132	526.18	-
C50-1	0.0124	2224	20.13	0.0124	2224	32.22	N/A
C50-2	0.0124	2224	20.16	0.0124	2224	32.10	-
C50-3	0.0124	2224	19.95	0.0124	2224	31.86	-
C50-4	0.0124	2224	20.98	0.0124	2224	33.57	-
C50-5	0.0124	2224	21.20	0.0124	2224	33.98	-
C60-1	0.0661	1630	10.21	0.0661	1630	41.28	-
C60-2	0.0661	1630	10.38	0.0661	1630	41.06	-
C60-3	0.0661	1630	10.21	0.0661	1630	41.26	-
C60-4	0.0661	1630	10.25	0.0661	1630	41.34	-
C60-5	0.0661	1630	10.25	0.0661	1630	41.33	-
C70-1	0.0327	2838	30.40	0.0327	2838	63.12	N/A
C70-2	0.0381	2838	30.47	0.0381	2838	62.59	-
C70-3	0.0381	2838	47.12	0.0381	2838	83.22	-
C70-4	0.0381	2838	47.50	0.0381	2838	84.08	-
C70-5	0.0381	2838	46.52	0.0381	2838	82.36	-
C80-1	0.1320	1962	18.13	0.1320	1962	116.49	-
C80-2	0.1320	1962	13.77	0.1320	1962	111.15	-
C80-3	0.1320	1962	13.77	0.1320	1962	111.02	-
C80-4	0.1320	1962	13.77	0.1320	1962	111.29	-
C80-5	0.1320	1962	13.80	0.1320	1962	111.39	-
C90-1	0.0354	1282	11.48	0.0354	1282	128.57	-
C90-2	0.0354	1282	11.53	0.0354	1282	129.00	-
C90-3	0.0354	1282	11.51	0.0354	1282	129.17	-
C90-4	0.0354	1282	11.57	0.0354	1282	129.25	-
C90-5	0.0354	1282	11.66	0.0354	1282	129.35	-
C100-1	0.0038	2448	28.06	0.0031	2628	46.02	N/A
C100-2	0.0128	2798	22.92	0.0128	2798	40.36	-
C100-3	0.0128	2798	18.84	0.0128	2798	39.03	-
C100-4	0.0128	2798	19.57	0.0128	2978	41.90	SPT
C100-5	0.0128	2798	19.44	0.0128	2978	41.92	SPT

Table 5.19: Solutions for Problem Set C using Tabu Search with NFT Penalty and Adaptive PST Selection

Algorithm	Best	Over-budget	Avg Time
Static, Best, Takahashi	37	6	34.80
Static, Best, SPT	37	6	324.22
Static, Adaptive, Takahashi	37	6	31.21
Static, Adaptive, SPT	39	6	170.93
NFT, Best, Takahashi	37	6	34.68
NFT, Best, SPT	37	6	322.80
NFT, Adaptive, Takahashi	37	6	31.50
NFT, Adaptive, SPT	39	6	169.61

Table 5.20: Average Results Overview for Problem Set C

5.6 Summary

Given the results and their analysis in this chapter, it is possible to draw a number of conclusions with respect to the evaluation objectives given at the beginning of this chapter. The principle conclusion is that the tabu search algorithms are well suited to the MTPSTO problem, outperforming the exhaustive search and naive heuristics for all but problems of small graph sizes. The tabu search is able to find solutions to problems with much larger connectivity graphs than the exhaustive search within a reasonable time, with optimal solutions being found for problems where they are known. Compared to the Takahashi-Matsuyama and SPT heuristics, the relative error of the solutions found by the naive heuristics are, for the most part, very high.

The properties of the tabu search, the penalty function, the PST selection policy, and the diversification strategy were also considered. The use of the NFT penalty function was expected to give improved results when compared to the static penalty function, however this proved not to be the case, with the NFT penalty function finding some over-budget solutions for problem set A and B, while none were found when the static penalty was used. For problem set C, there was no difference between the two. The poor performance of the NFT method may be due to the user-set parameter of the NFT penalty function not being ideal, or it may be due to the method being less effective where there are few constraints, as highlighted by its authors. The static penalty function is by no means perfect, with one particularly interesting case where the initial solution is a better solution than that eventually found by the tabu search using this penalty method.

For the PST selection policy, conflicting conclusions can be drawn. For problem set B, the running times increase with the use of the adaptive policy, but for problem set C, they decrease. The impact on the results can be seen when it is used with the Takahashi-Matsuyama diversification for problem set B and the SPT diversification for problem set C. Clearly, further research is needed on the PST selection policy. For many of the

Pr	Algorithms
C20-1	
C20-2	(1) (2) (3) (4) (5) (6) (7) (8)
C20-3	(1) (2) (3) (4) (5) (6) (7) (8)
C20-4	(1) (2) (3) (4) (5) (6) (7) (8)
C20-5	(1) (2) (3) (4) (5) (6) (7) (8)
C30-1	
C30-2	(1) (2) (3) (4) (5) (6) (7) (8)
C30-3	(1) (2) (3) (4) (5) (6) (7) (8)
C30-4	(1) (2) (3) (4) (5) (6) (7) (8)
C30-5	(1) (2) (3) (4) (5) (6) (7) (8)
C40-1	
C40-2	(1) (2) (3) (4) (5) (6) (7) (8)
C40-3	(1) (2) (3) (4) (5) (6) (7) (8)
C40-4	(1) (2) (3) (4) (5) (6) (7) (8)
C40-5	(1) (2) (3) (4) (5) (6) (7) (8)
C50-1	
C50-2	(1) (2) (3) (4) (5) (6) (7) (8)
C50-3	(1) (2) (3) (4) (5) (6) (7) (8)
C50-4	(1) (2) (3) (4) (5) (6) (7) (8)
C50-5	(1) (2) (3) (4) (5) (6) (7) (8)
C60-1	(1) (2) (3) (4) (5) (6) (7) (8)
C60-2	(1) (2) (3) (4) (5) (6) (7) (8)
C60-3	(1) (2) (3) (4) (5) (6) (7) (8)
C60-4	(1) (2) (3) (4) (5) (6) (7) (8)
C60-5	(1) (2) (3) (4) (5) (6) (7) (8)
C70-1	
C70-2	(1) (2) (3) (4) (5) (6) (7) (8)
C70-3	(1) (2) (3) (4) (5) (6) (7) (8)
C70-4	(1) (2) (3) (4) (5) (6) (7) (8)
C70-5	(1) (2) (3) (4) (5) (6) (7) (8)
C80-1	(1) (2) (3) (4) (5) (6) (7) (8)
C80-2	(1) (2) (3) (4) (5) (6) (7) (8)
C80-3	(1) (2) (3) (4) (5) (6) (7) (8)
C80-4	(1) (2) (3) (4) (5) (6) (7) (8)
C80-5	(1) (2) (3) (4) (5) (6) (7) (8)
C90-1	(1) (2) (3) (4) (5) (6) (7) (8)
C90-2	(1) (2) (3) (4) (5) (6) (7) (8)
C90-3	(1) (2) (3) (4) (5) (6) (7) (8)
C90-4	(1) (2) (3) (4) (5) (6) (7) (8)
C90-5	(1) (2) (3) (4) (5) (6) (7) (8)
C100-1	
C100-2	(1) (2) (3) (4) (5) (6) (7) (8)
C100-3	(1) (2) (3) (4) (5) (6) (7) (8)
C100-4	(4) (8)
C100-5	(4) (8)

Table 5.21: Algorithm Finding Best Solutions for Problem Set C

	Takahashi					SPT				
	PST		Rel. Error			PST		Rel. Error		
Pr	τ_T	O_T	η_τ	η_O	Sec	τ_T	O_T	η_τ	η_O	Sec
A0-2	0.0322	2179	-	-	0.0071	0.0322	2179	-	-	0.0087
A0-3	0.0322	2179	-	-	0.0071	0.0322	2179	-	-	0.0087
A0-4	0.0322	2179	-	-	0.0071	0.0322	2179	-	-	0.0087
A0-5	0.0322	2179	-	-	0.0071	0.0322	2179	-	-	0.0087
A1-2	0.0023	2884	0.873	0.2027	0.0064	0.0181	2398	-	-	0.006
A1-3	0.0023	2884	0.873	0.2027	0.0064	0.0181	2398	-	-	0.006
A1-4	0.0023	2884	0.873	0.2027	0.0064	0.0181	2398	-	-	0.006
A1-5	0.0023	2884	0.873	0.2027	0.0064	0.0181	2398	-	-	0.006
A2-1	0.0931	1850	-	-	0.0076	0.0931	1850	-	-	0.0059
A2-2	0.0931	1850	-	-	0.0076	0.0931	1850	-	-	0.0059
A2-3	0.0931	1850	-	-	0.0076	0.0931	1850	-	-	0.0059
A2-4	0.0931	1850	-	-	0.0076	0.0931	1850	-	-	0.0059
A2-5	0.0931	1850	-	-	0.0076	0.0931	1850	-	-	0.0059
A3-2	0.0223	3297	0.00128	0.1303	0.0086	0.0223	3436	-	-	0.0059
A3-3	0.0223	3297	0.00128	0.1303	0.0086	0.0223	3436	-	-	0.0059
A3-4	0.0223	3297	0.00128	0.1303	0.0086	0.0223	3436	-	-	0.0059
A3-5	0.0223	3297	0.00128	0.1303	0.0086	0.0223	3436	-	-	0.0059
A4-2	0.1855	2224	-	-	0.0083	0.0933	2564	0.4968	0.1529	0.0052
A4-3	0.1855	2224	-	-	0.0083	0.0933	2564	0.4968	0.1529	0.0052
A4-4	0.1855	2224	-	-	0.0083	0.0933	2564	0.4968	0.1529	0.0052
A4-5	0.1855	2224	-	-	0.0083	0.0933	2564	0.4968	0.1529	0.0052
A5-2	0.0542	2262	-	-	0.0078	0.0369	2968	0.3192	0.3121	0.0079
A5-3	0.0542	2262	0.3323	0.2922	0.0078	0.0369	2968	0.5454	0.0713	0.0079
A5-4	0.0542	2262	0.3323	0.2922	0.0078	0.0369	2968	0.5454	0.0713	0.0079
A5-5	0.0542	2262	0.3323	0.2922	0.0078	0.0369	2968	0.5454	0.0713	0.0079
A6-3	0.0241	3911	0.3318	0.0169	0.0185	0.0253	3828	0.2966	0.0047	0.0081
A6-4	0.0241	3911	0.3318	0.114	0.0185	0.0253	3828	0.2966	0.1328	0.0081
A6-5	0.0241	3911	0.3318	0.114	0.0185	0.0253	3828	0.2966	0.1328	0.0081
A7-3	0.0644	3512	0.0685	0.0162	0.0091	0.0057	3132	0.9458	0.1124	0.0053
A7-4	0.0644	3512	0.0685	0.0162	0.0091	0.0057	3132	0.9458	0.1124	0.0053
A7-5	0.0644	3512	0.0685	0.0162	0.0091	0.0057	3132	0.9458	0.1124	0.0053
A8-3	0.00031	4293	0.9011	0.1739	0.0106	0.00017	4484	0.9458	0.2261	0.0053
A8-4	0.00031	4293	0.9011	0.065	0.0106	0.00017	4484	0.9458	0.1124	0.0053
A8-5	0.00031	4293	0.9011	0.065	0.0106	0.00017	4484	0.9458	0.1124	0.0053
A9-1	0.0193	2278	0.9114	0.2085	0.0093	0.0359	2104	0.8358	0.1162	0.0058
A9-2	0.0193	2278	0.9114	0.2085	0.0093	0.0359	2104	0.8358	0.1162	0.0058
A9-3	0.0193	2278	0.9114	0.2085	0.0093	0.0359	2104	0.8358	0.1162	0.0058
A9-4	0.0193	2278	0.9114	0.2085	0.0093	0.0359	2104	0.8358	0.1162	0.0058
A9-5	0.0193	2278	0.9114	0.2085	0.0093	0.0359	2104	0.8358	0.1162	0.0058

Table 5.22: Solutions for Problem Set A using the Takahashi-Matsuyama Steiner Tree Heuristic & the Shortest Path Tree Heuristic

	Takahashi					SPT				
	PST		Rel. Error			PST		Rel. Error		
Pr	τ_T	O_T	η_τ	η_O	Sec	τ_T	O_T	η_τ	η_O	Sec
B10-1	0.00147	3871	N/A	N/A	0.0294	0.0254	4916	N/A	N/A	0.0181
B10-2	0.00147	3871	0.9865	0.3293	0.0294	0.0254	4916	0.768	0.6882	0.0181
B10-3	0.00147	3871	0.9865	0.0777	0.0294	0.0254	4916	0.768	0.3686	0.0181
B10-4	0.00147	3871	0.9865	0.0777	0.0294	0.0254	4916	0.768	0.3686	0.0181
B10-5	0.00147	3871	0.9865	0.0777	0.0294	0.0254	4916	0.768	0.3686	0.0181
B11-1	0.0134	4820	N/A	N/A	0.0268	0.05	5354	N/A	N/A	0.0185
B11-2	0.0134	4820	N/A	N/A	0.0268	0.05	5354	N/A	N/A	0.0185
B11-3	0.0134	4820	0.8149	0.3791	0.0268	0.05	5354	0.307	0.5319	0.0185
B11-4	0.0134	4820	0.8149	0.3791	0.0268	0.05	5354	0.307	0.5319	0.0185
B11-5	0.0134	4820	0.8149	0.3791	0.0268	0.05	5354	0.307	0.5319	0.0185
B12-1	8×10^{-4}	4699	N/A	N/A	0.0473	0.0394	4674	N/A	N/A	0.0211
B12-2	8×10^{-4}	4699	0.9787	0.5961	0.0473	0.0394	4674	-	0.5876	0.0211
B12-3	8×10^{-4}	4699	0.9787	0.2327	0.0473	0.0394	4674	0.001	0.2261	0.0211
B12-4	8×10^{-4}	4699	0.9787	0.2327	0.0473	0.0394	4674	0.001	0.2261	0.0211
B12-5	8×10^{-4}	4699	0.9787	0.2327	0.0473	0.0394	4674	0.001	0.2261	0.0211
B13-1	0.00244	3224	0.9683	1.1799	0.023	0.0207	3223	0.7317	1.1792	0.0192
B13-2	0.00244	3224	0.9683	1.1799	0.023	0.0207	3223	0.7317	1.1792	0.0192
B13-3	0.00244	3224	0.9683	1.1799	0.023	0.0207	3223	0.7317	1.1792	0.0192
B13-4	0.00244	3224	0.9683	1.1799	0.023	0.0207	3223	0.7317	1.1792	0.0192
B13-5	0.00244	3224	0.9683	1.1799	0.023	0.0207	3223	0.7317	1.1792	0.0192
B14-1	0.0198	4271	N/A	N/A	0.0263	0.0175	5062	N/A	N/A	0.0234
B14-2	0.0198	4271	0.6849	0.5303	0.0263	0.0175	5062	0.7205	0.8137	0.0234
B14-3	0.0198	4271	0.7196	0.1317	0.0263	0.0175	5062	0.7513	0.3413	0.0234
B14-4	0.0198	4271	0.7196	0.1317	0.0263	0.0175	5062	0.7513	0.3413	0.0234
B14-5	0.0198	4271	0.7196	0.1317	0.0263	0.0175	5062	0.7513	0.3413	0.0234
B15-1	3×10^{-4}	3811	N/A	N/A	0.0282	0.0167	5267	N/A	N/A	0.0209
B15-2	3×10^{-4}	3811	N/A	N/A	0.0282	0.0167	5267	N/A	N/A	0.0209
B15-3	3×10^{-4}	3811	0.9976	0.0417	0.0282	0.0167	5267	0.8778	0.3244	0.0209
B15-4	3×10^{-4}	3811	0.9976	0.1518	0.0282	0.0167	5267	0.8778	0.1723	0.0209
B15-5	3×10^{-4}	3811	0.9976	0.1518	0.0282	0.0167	5267	0.8778	0.1723	0.0209
B16-1	7×10^{-4}	5738	N/A	N/A	0.0291	0.00343	7588	N/A	N/A	0.0284
B16-2	7×10^{-4}	5738	N/A	N/A	0.0291	0.00343	7588	N/A	N/A	0.0284
B16-3	7×10^{-4}	5738	N/A	N/A	0.0291	0.00343	7588	N/A	N/A	0.0284
B16-4	7×10^{-4}	5738	0.9963	0.1826	0.0291	0.00343	7588	0.8069	0.5639	0.0284
B16-5	7×10^{-4}	5738	0.9963	0.1207	0.0291	0.00343	7588	0.8069	0.482	0.0284
B17-1	0.0539	4001	N/A	N/A	0.0301	0.0540	3743	N/A	N/A	0.0252
B17-2	0.0539	4001	0.5207	0.3471	0.0301	0.0540	3743	0.5197	0.2603	0.0252
B17-3	0.0539	4001	0.5207	0.0228	0.0301	0.0540	3743	0.5197	0.0432	0.0252
B17-4	0.0539	4001	0.5207	0.0228	0.0301	0.0540	3743	0.5197	0.0432	0.0252
B17-5	0.0539	4001	0.5207	0.0228	0.0301	0.0540	3743	0.5197	0.0432	0.0252
B18-1	0.0935	3491	0.5785	0.9087	0.033	0.0360	3635	0.8378	0.9874	0.0192
B18-2	0.0935	3491	0.5785	0.9087	0.033	0.0360	3635	0.8378	0.9874	0.0192
B18-3	0.0935	3491	0.5785	0.9087	0.033	0.0360	3635	0.8378	0.9874	0.0192
B18-4	0.0935	3491	0.5785	0.9087	0.033	0.0360	3635	0.8378	0.9874	0.0192
B18-5	0.0935	3491	0.5785	0.9087	0.033	0.0360	3635	0.8378	0.9874	0.0192
B19-1	0.1117	2971	-	-	0.0224	0.1457	3392	-	-	0.0216
B19-2	0.1117	2971	-	-	0.0224	0.1457	3392	-	-	0.0216
B19-3	0.1117	2971	-	-	0.0224	0.1457	3392	-	-	0.0216
B19-4	0.1117	2971	-	-	0.0224	0.1457	3392	-	-	0.0216
B19-5	0.1117	2971	-	-	0.0224	0.1457	3392	-	-	0.0216

Table 5.23: Solutions for Problem Set B using the Takahashi-Matsuyama Steiner Tree Heuristic & the Shortest Path Tree Heuristic

	Takahashi					SPT				
	PST		Rel. Error			PST		Rel. Error		
Pr	τ_T	O_T	η_τ	η_O	Sec	τ_T	O_T	η_τ	η_O	Sec
C20-1	0.0213	4456	N/A	N/A	0.0212	0.0198	4620	N/A	N/A	0.0192
C20-2	0.0213	4456	0.8241	0.5115	0.0212	0.0198	4620	0.8366	0.5672	0.0192
C20-3	0.0213	4456	0.8241	0.1327	0.0212	0.0198	4620	0.8366	0.1744	0.0192
C20-4	0.0213	4456	0.8241	0.1327	0.0212	0.0198	4620	0.8366	0.1744	0.0192
C20-5	0.0213	4456	0.8241	0.1327	0.0212	0.0198	4620	0.8366	0.1744	0.0192
C30-1	0.0170	1758	N/A	N/A	0.0164	0.0170	1758	N/A	N/A	0.0116
C30-2	0.0170	1758	0.8723	0.2131	0.0164	0.0170	1758	0.8723	0.2131	0.0116
C30-3	0.0170	1758	0.8723	0.2131	0.0164	0.0170	1758	0.8723	0.2131	0.0116
C30-4	0.0170	1758	0.8723	0.2131	0.0164	0.0170	1758	0.8723	0.2131	0.0116
C30-5	0.0170	1758	0.8723	0.2131	0.0164	0.0170	1758	0.8723	0.2131	0.0116
C40-1	0.00322	2280	N/A	N/A	0.0178	0.00445	3002	N/A	N/A	0.00978
C40-2	0.00322	2280	0.8686	0.1108	0.0178	0.00445	3002	0.8179	0.1708	0.00978
C40-3	0.00322	2280	0.8686	0.272	0.0178	0.00445	3002	0.8179	0.0415	0.00978
C40-4	0.00322	2280	0.8686	0.272	0.0178	0.00445	3002	0.8179	0.0415	0.00978
C40-5	0.00322	2280	0.8686	0.272	0.0178	0.00445	3002	0.8179	0.0415	0.00978
C50-1	0.00267	2224	N/A	N/A	0.0171	0.00937	2564	N/A	N/A	0.013
C50-2	0.00267	2224	0.7852	-	0.0171	0.00937	2564	0.2454	0.1529	0.013
C50-3	0.00267	2224	0.7852	-	0.0171	0.00937	2564	0.2454	0.1529	0.013
C50-4	0.00267	2224	0.7852	-	0.0171	0.00937	2564	0.2454	0.1529	0.013
C50-5	0.00267	2224	0.7852	-	0.0171	0.00937	2564	0.2454	0.1529	0.013
C60-1	0.0661	1630	-	-	0.021	0.0661	1720	2×10^{-4}	0.0552	0.0126
C60-2	0.0661	1630	-	-	0.021	0.0661	1720	2×10^{-4}	0.0552	0.0126
C60-3	0.0661	1630	-	-	0.021	0.0661	1720	2×10^{-4}	0.0552	0.0126
C60-4	0.0661	1630	-	-	0.021	0.0661	1720	2×10^{-4}	0.0552	0.0126
C60-5	0.0661	1630	-	-	0.021	0.0661	1720	2×10^{-4}	0.0552	0.0126
C70-1	0.00424	3222	N/A	N/A	0.024	0.0131	2564	N/A	N/A	0.0101
C70-2	0.00424	3222	0.8886	0.1353	0.024	0.0131	2564	0.6564	0.0965	0.0101
C70-3	0.00424	3222	0.8886	0.1353	0.024	0.0131	2564	0.6564	0.0965	0.0101
C70-4	0.00424	3222	0.8886	0.1353	0.024	0.0131	2564	0.6564	0.0965	0.0101
C70-5	0.00424	3222	0.8886	0.1353	0.024	0.0131	2564	0.6564	0.0965	0.0101
C80-1	0.0316	2253	0.7607	0.1483	0.0314	0.1319	1962	-	-	0.0148
C80-2	0.0316	2253	0.7607	0.1483	0.0314	0.1319	1962	-	-	0.0148
C80-3	0.0316	2253	0.7607	0.1483	0.0314	0.1319	1962	-	-	0.0148
C80-4	0.0316	2253	0.7607	0.1483	0.0314	0.1319	1962	-	-	0.0148
C80-5	0.0316	2253	0.7607	0.1483	0.0314	0.1319	1962	-	-	0.0148
C90-1	0.0354	1282	-	-	0.0291	0.0354	1282	-	-	0.012
C90-2	0.0354	1282	-	-	0.0291	0.0354	1282	-	-	0.012
C90-3	0.0354	1282	-	-	0.0291	0.0354	1282	-	-	0.012
C90-4	0.0354	1282	-	-	0.0291	0.0354	1282	-	-	0.012
C90-5	0.0354	1282	-	-	0.0291	0.0354	1282	-	-	0.012
C100-1	3×10^{-4}	2738	N/A	N/A	0.0262	8×10^{-4}	2968	N/A	N/A	0.0137
C100-2	3×10^{-4}	2738	0.9792	0.0214	0.0262	8×10^{-4}	2968	0.9382	0.0608	0.0137
C100-3	3×10^{-4}	2738	0.9792	0.0214	0.0262	8×10^{-4}	2968	0.9382	0.0608	0.0137
C100-4	3×10^{-4}	2738	0.9792	0.0806	0.0262	8×10^{-4}	2968	0.9382	0.0034	0.0137
C100-5	3×10^{-4}	2738	0.9792	0.0806	0.0262	8×10^{-4}	2968	0.9382	0.0034	0.0137

Table 5.24: Solutions for Problem Set C using the Takahashi-Matsuyama Steiner Tree Heuristic & the Shortest Path Tree Heuristic

problems, the tabu search algorithms using the Takahashi-Matsuyama diversification finds less optimal solutions or better solutions than the SPT diversification, but the latter results in longer running times.

Although the tabu search algorithm is largely successful in finding optimal solutions and has a significantly lower growth rate in the running time, there are some problems in the experimentation that led to excessive running time. While these still outperform what one would expect of the exhaustive search, these rare cases give impractical times for real-world application. For these problems, and perhaps others too, there would be an improvement in the running times if the PST enumeration could be eliminated from the algorithm and replaced by a tabu search move selection heuristic that not only considers the impact of the trust by applying the move to a PST, but also the change in overhead value. This would result in the application of a move to a PST giving a new PST and not a subgraph, upon which PSTs must be enumerated. While this and the penalty functions require further investigation, it is clear that the tabu search methods proposed provide good approximations where the optimal solutions are not found, and outperform the running times of the exhaustive search.

Chapter 6

Conclusions and Future Work

6.1 Research Contributions

6.1.1 Trust Metric for PSTs

The application of trust in P2P networks has been proposed as a means to prevent selfish and malicious behaviour by peers. Inspired by this work and the security issues that afflict publish/subscribe systems, the primary research objective of this thesis was to investigate the construction of PSTs with respect to the trust preferences of publishers and subscribers, and the communication overhead costs. The premise for this is that a PST constructed with respect to these properties will reduce the likelihood of attacks against its participants, whilst ensuring that communication remains efficient.

Trust metrics in P2P networks define either local or global trust values of peers that can be used to determine if an entity is sufficiently trustworthy for an interaction with it to take place. This differs greatly to the trust metric required in this thesis, where it defines the trustworthiness of a network structure. Defining the trustworthiness of a PST, as a function of the trust held by the PST's publisher and subscribers in each other, was achieved by using social choice and welfare theory. After identifying the relationships in the PST between publishers, internal subscribers and leaf subscribers, a trust metric was defined that is based upon Rawls' principles of justice through the use of the Leximin social welfare functional. The trustworthiness of a PST is dominated by that of vertex with the least trust in the PST and the trust held in a PST by a vertex is given by the trustworthiness of the end-to-end communication paths between it and the other clients. Given the proposed mechanism to determine the trustworthiness of a PST, a set of feasible PSTs can be socially ordered with respect to the individuals' trust functions.

6.1.2 Inter-personal Incomparability of Trust

An important observation was made regarding the interpersonal incomparability of trust. In the existing literature, trust is assumed to be interpersonal comparable, however, we are argue that this is not the case. Regardless of the trust metric, ordinal or cardinal, two individuals may differ in how they value the trustworthiness of some entity even if the trust sources are identical, that is their trust functions used to evaluate the trustworthiness of an entity may not be the same. This has important ramifications to much of the existing work on computational trust, where it is assumed that individuals' trust functions and local trust values are comparable. As the proposed approach to devise the trustworthiness of PSTs utilises the analytical leximin aggregation function, cardinal full comparability of the individuals' trust functions must hold true, however arguments against this are presented. The strict assumptions made in this thesis in order to address this issue are to assume that evaluation of the trust sources and the individuals' trust functions are identical. Although it is unrealistic to assume such homogeneity of publishers, routers, and subscribers, it is no worse than not addressing the issue at all. Additionally, we postulate that the individual trust evaluation functions may evolve to a state where they are inter-personally comparable. If we assume the mechanism, that is the socially trusted PST social welfare function (definition 41), described in this thesis to determine the most trusted PST to be incentive-compatible, then the individuals' trust functions (that is their notions of trust) will evolve some common understanding, a Nash equilibrium.

6.1.3 The Maximum Trusted PST with Overhead Budget Problem

Having defined a trust metric for PSTs, the problem of finding the most trusted PST within some overhead budget was shown to be NP-Complete. As expected, the exhaustive search algorithm for this problem was shown to be unable to scale beyond small problem sizes. Algorithms using the tabu search metaheuristic were devised and shown to scale to problem sizes where $|R| = 100$ with good approximation solutions found and running times that are comparable to the exhaustive search for much smaller problems.

In a real-world deployment, the proposed tabu search algorithms would be executed at the publisher vertex. Algorithm execution could then take place in rounds, at regular intervals or when change in the state of any of the inputs warrants the reconfiguration of the PST. Subscribers would provide the required inputs such as trust information and subscriptions to the algorithm publisher prior to a round, while the connectivity graph could be maintained by the use of a gossip protocol.

6.2 Future Work

6.2.1 Monitoring

There are three components to P2P reputation management systems (Marti and Garcia-Molina, 2006): information gathering; scoring and ranking; response. If the past behaviour of nodes is to used as an input to individuals' trust functions, and consequently the social trust ordering of PSTs, then an information gathering stage is required. For P2P systems, this is typically feedback pertaining to transactions between peers, however this alone is not adequate for PSTs.

Consider a path in a PST, from a publisher to a terminal subscriber of length greater than two. The terminal subscriber does not receive a notification from the publisher, and as a consequence it may wish to reduce its trust of its parent node and the publisher's child node. This would be unfair on one of these nodes, as only one of them can be the culprit responsible for dropping notification (if we exclude communication error as a cause). This example assumes that terminal subscriber has a means to determine that it has missed a notification, which given the nature of publish/subscribe, can not be the case unless it verifies this to be true with the publisher or some other trusted subscriber. The space, time and synchronisation decoupling that give rise to scalability of publish/subscribe systems (Eugster et al., 2003) are attributes that inhibit determining the source of selfish and malicious behaviour.

Clearly the monitoring of PSTs presents unique challenges, however for PSTs in mobile ad hoc networks (MANETs), existing work on the use of reputation in MANET routing may prove to be a starting point for further research. CONFIDANT (Buchegger and Le Boudec, 2004) is a reputation management system for mobile ad hoc network routing that allows the routing protocol to determine trustworthy paths and reject route requests from untrustworthy nodes. Monitoring is conducted by nodes in promiscuous mode, observing the routing behaviour of others.

6.2.2 Improvements to the Tabu Search Algorithms

Although the tabu search algorithms provide good approximations of the optimal solutions and have a faster running time than the exhaustive search algorithm, there are a number of possible improvements that could be made. Spanning tree enumeration is used to find the set of PSTs in the subgraph formed by applying a tabu move (the addition or removal of a router) to the current PST solution. Char's spanning tree enumeration algorithm (Char,

1968) is used for this and has a running time of $\mathcal{O}(m + n + n(t + t_0))$, however Knuth (Knuth, 2011) documents an algorithm by Malcolm Smith (Smith, 1997) that generates spanning trees in gray code order (Gray, 1953) that has a running time of $\mathcal{O}(m + n + t)$. Smith's algorithm could provide a faster means to finding PSTs.

6.2.3 Self-organising Trusted PST Algorithm

Algorithms have been proposed for self-organising broker networks with respect to communication cost (Jaeger et al., 2007) (Baldoni et al., 2007) (Migliavacca and Cugola, 2007), and for the PST structure considered in this work, the SHOPPARENT (Huang and Garcia-Molina, 2003) and DSAPST (Cao and Shen, 2009) algorithms self-organise the PST with respect to the overhead costs as given in equations 2.3 and 2.4.

When executed at the publisher, the tabu search algorithms require knowledge of the connectivity graph, these subscriptions of the subscribers, the individual trust functions of the subscribers and the inputs to these functions. As the number of subscribers and the graph size increases, maintaining global state at the publisher increases in message and state complexity at this node. The development of a distributed heuristic may present a solution to this issue, however it is not clear how privacy of individuals' trust relationships could be preserved by the technique. A logical assumption of the distributed heuristic would be that it must not leak trust information to other routers and subscribers, even if they are trusted, as the information may reveal a loss in trust in them.

6.3 Closing Remarks

Publish/Subscribe has evolved a great deal from its precursor of group communication systems, the latter originally introduced in the System V kernel for interprocess communication (Cheriton and Zwaenepoel, 1985). Subsequent research had led to developments such as topic-based and content-based models, the use of decentralised event notification services, and routing and matchmaking algorithm optimisations. In comparison, there is significantly less research on security, which is at least in part due to the decoupled properties of publish/subscribe that are contradictory to the coupling that is required of most security techniques. The proposed approaches described in chapter 2 are either purely cryptographic-based or a form of role-based access-control, and while they address the issues of trust and confidentiality to varying degrees, the approach presented in this work has a number of advantages over them.

The trusted PST approach provides a greater degree of adaptability to the changing

behaviour in the publish/subscribe system, and it is suitable for ad hoc publish/subscribe applications, as the identities of entities need not be associated with roles and there is no requirement for administrative entities to define access-control policies. Although the monitoring system is beyond the scope of this work, its presence allows for PSTs to adapt to changes in the behaviour of entities. As trust evaluations change with interactions and over time, the PSTs can be reconstructed to eliminate selfish or malicious nodes. The network structure evolves in response to the behaviour of its participants, something that is not possible in the approaches evaluated in chapter 2. For example, in RBAC systems, the reconfiguration of policies, and the revoking and reissuing of keys is required to replicate this behaviour. No previously proposed approach allows for the clients of a publish/subscribe system to be considered either trustworthy or untrustworthy given their past behaviour and for their removal from the communication infrastructure given this information.

In this work, a trust evaluation function based on social choice and welfare theory has been proposed to define the trustworthiness of a PST, the problem to find the most trustworthy PST within some overhead budget has been shown to be NP-complete, and the tabu search metaheuristic has been shown to be effective at solving this problem. It has been shown that individual trust can be used to construct network structures that are socially trusted by its users, however consideration must be given to the inter-personal comparability of the individuals' trust functions.

Bibliography

Abadi, M. and Feigenbaum, J. (1990). Secure circuit evaluation. *Journal of Cryptology*, 2:1–12. Cited on 20

Abadi, M., Feigenbaum, J., and Kilian, J. (1987). On hiding information from an oracle. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, STOC '87, pages 195–203, New York, NY, USA. ACM. Cited on 20

Arrow, K. J. (1951). *Social Choice and Individual Values*. Yale University Press, First edition. Cited on 32, 34

Arrow, K. J. (1963). *Social Choice and Individual Values*. Yale University Press, Second edition. Cited on 32, 34

Arrow, K. J., Sen, A. K., and Suzumura, K., editors (2002). *Handbook of Social Choice and Welfare*, volume 1. Elsevier. Cited on 5, 31, 32

Baldoni, R., Beraldi, R., Querzoni, L., and Virgillito, A. (2007). Efficient publish/subscribe through a self-organizing broker overlay and its application to SIENA. *The Computer Journal*, 50(4):444. Cited on 120

Baldoni, R., Marchetti, C., Virgillito, A., and Vitenberg, R. (2005). Content-based publish-subscribe over structured overlay networks. In *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems*, ICDCS 2005, pages 437–446. IEEE. Cited on 9

Barabási, A., Albert, R., and Jeong, H. (2000). Scale-free characteristics of random networks: the topology of the world-wide web. *Physica A: Statistical Mechanics and its Applications*, 281(1-4):69–77. Cited on 81, 86

Beasley, J. E. (1990). OR-Library: Distributing test problems by electronic mail. *The Journal of the Operational Research Society*, 41(11):pp. 1069–1072. Cited on 77

Bellare, M., Rogaway, P., and Wagner, D. (2004). The EAX mode of operation. In *Fast Software Encryption*, volume 3017 of *Lecture Notes in Computer Science*, pages 389–407. Springer Berlin/Heidelberg. Cited on 27

Belokosztolszki, A., Eyers, D. M., Pietzuch, P. R., Bacon, J., and Moody, K. (2003). Role-based access control for publish/subscribe middleware architectures. In *Proceedings of the 2nd International Workshop on Distributed Event-Based Systems*, DEBS '03, pages 1–8, New York, NY, USA. ACM. Cited on 4, 26, 27

Bentham, J. (1781). *An Introduction to the Principles of Morals and Legislation.* T. Payne. Cited on 32

Bentley, J. L. and McIlroy, M. D. (1993). Engineering a sort function. *Software: Practice and Experience*, 23(11):1249–1265. Cited on 52

Birman, K. and Joseph, T. (1987). Exploiting virtual synchrony in distributed systems. *SIGOPS Operating Systems Review*, 21:123–138. Cited on 9

Bloom, B. H. (1970). Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13:422–426. Cited on 21

Borda, J. C. d. (1781). Mémoire sur les élections au scrutin. *Histoire de l'Académie Royale des Sciences.* Cited on 31

Buchegger, S. and Le Boudec, J. (2004). A robust reputation system for mobile ad-hoc networks. In *Proceedings of the 2nd Workshop on Economics of Peer-to-Peer Systems*, P2PEcon 2004, Cambridge, MA, USA. Cited on 2, 119

Cao, X. and Shen, C. (2009). Subscription-aware publish/subscribe tree construction in mobile ad-hoc networks. In *Proceedings of the 13th International Conference on Parallel and Distributed Systems (ICPADS 2007)*, volume 2, pages 1–9. IEEE. Cited on 13, 14, 52, 120

Caporuscio, M., Carzaniga, A., and Wolf, A. (2003). Design and evaluation of a support service for mobile, wireless publish/subscribe applications. *IEEE Transactions on Software Engineering*, 29(12):1059–1071. Cited on 9

Carzaniga, A., Rosenblum, D., and Wolf, A. (2001). Design and evaluation of a wide-area event notification service. *ACM Transactions on Computer Systems (TOCS)*, 19(3):332–383. Cited on 9, 20

Carzaniga, A. and Wolf, A. L. (2003). Forwarding in a content-based network. In *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '03, pages 163–174, New York, NY, USA. ACM. Cited on 9

Chang, Y.-C. and Mitzenmacher, M. (2005). Privacy preserving keyword searches on remote encrypted data. In *Applied Cryptography and Network Security*, volume 3531 of *Lecture Notes in Computer Science*, pages 391–421. Springer Berlin/Heidelberg. Cited on 21

Char, J. (1968). Generation of trees, two-trees, and storage of master forests. *IEEE Transactions on Circuit Theory*, 15(3):228–238. Cited on 54, 72, 119

Cheriton, D. R. and Zwaenepoel, W. (1985). Distributed process groups in the V Kernel. *ACM Transactions on Computer Systems*, 3(2):77–107. Cited on 9, 120

Choi, Y. and Park, D. (2006). Mirinae: A peer-to-peer overlay network for content-based publish/subscribe systems. *IEICE Transactions on Communications*, E89-B(6):1755–1765. Cited on 9

Chor, B., Goldreich, O., Kushilevitz, E., and Sudan, M. (1995). Private information retrieval. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science*, pages 41–50. IEEE. Cited on 20

Condorcet, M. J. A. N. d. C. d. (1785). *Essai sur l' application de l' analyse à la probabilité des décisions : rendues à la pluralité*. L'imprimerie royale. Cited on 31

Cugola, G., Di Nitto, E., and Fuggetta, A. (2002). The JEDI event-based infrastructure and its application to the development of the OPSS WFMS. *IEEE Transactions on Software Engineering*, 27(9):827–850. Cited on 9

Cugola, G. and Jacobsen, H. (2002). Using publish/subscribe middleware for mobile systems. *ACM SIGMOBILE Mobile Computing and Communications Review*, 6(4):25–33. Cited on 9

Di Crescenzo, G., Malkin, T., and Ostrovsky, R. (2000). Single database private information retrieval implies oblivious transfer. In *Advances in Cryptology - EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 122–138. Springer Berlin/Heidelberg. Cited on 20

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271. Cited on 70

Ebel, H., Mielsch, L.-I., and Bornholdt, S. (2002). Scale-free topology of e-mail networks. *Physical Review E*, 66(3):1–4. Cited on 81

Eppstein, D. and Wang, J. Y. (2002). A steady state model for graph power laws. ACM Computing Research Repository. Cited on 81

Eugster, P., Felber, P., Guerraoui, R., and Kermarrec, A. (2003). The many faces of publish/subscribe. *ACM Computing Surveys*, 35(2):114–131. Cited on 3, 9, 119

Fabret, F., Jacobsen, H. A., Llirbat, F., Pereira, J., Ross, K. A., and Shasha, D. (2001). Filtering algorithms and implementation for very fast publish/subscribe systems. In *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data*, SIGMOD '01, pages 115–126, New York, NY, USA. ACM. Cited on 9

Fan, J., Xu, J., Ammar, M. H., and Moon, S. B. (2004). Prefix-preserving IP address anonymization: measurement-based security evaluation and a new cryptography-based scheme. *Computer Networks*, 46(2):253 – 272. Cited on 23

Fidler, E., Jacobsen, H.-A., Li, G., and Mankovski, S. (2005). The PADRES distributed publish/subscribe system. In *Feature Interactions in Telecommunications and Software Systems VIII*, pages 12–30. IOS Press. Cited on 18

Fiege, L., Gärtner, F. C., Kasten, O., and Zeidler, A. (2003). Supporting mobility in content-based publish/subscribe middleware. In *Proceedings of the ACM/IFIP/USENIX 2003 International Conference on Middleware*, Middleware '03, pages 103–122, New York, NY, USA. Springer-Verlag New York, Inc. Cited on 9

Fiege, L., Zeidler, A., Buchmann, A., Kilian-Kehr, R., and Muhl, G. (2004). Security aspects in publish/subscribe systems. *IEE Seminar Digests*, 2004(918):44–49. Cited on 24

Fishburn, P. C. (1973). *The Theory of Social Choice*. Princeton University Press. Cited on 31

Gabow, H. N. and Myers, E. W. (1978). Finding all spanning trees of directed and undirected graphs. *SIAM Journal on Computing*, 7(3):280–287. Cited on 54

Gendreau, M. (2003). An introduction to tabu search. In *Handbook of Metaheuristics*, volume 57 of *International Series in Operations Research & Management Science*, pages 37–54. Springer New York. Cited on 63

Gendreau, M., Hertz, A., and Laporte, G. (1994). A tabu search heuristic for the vehicle routing problem. *Management Science*, 40(10):1276–1290. Cited on 69

Gendreau, M., Larochelle, J.-F., and Sans, B. (1999). A tabu search heuristic for the steiner tree problem. *Networks*, 34(2):162–172. Cited on 61

Gennaro, R. and Micali, S. (1995). Verifiable secret sharing as secure computation. In *Advances in Cryptology - EUROCRYPT 95*, volume 921 of *Lecture Notes in Computer Science*, pages 168–182. Springer Berlin/Heidelberg. Cited on 20

Glover, F. (1989). Tabu Search–Part I. *INFORMS Journal on Computing*, 1(3):190–206. Cited on 60

Glover, F. (1990). Tabu Search–Part II. *INFORMS Journal on Computing*, 2(1):4–32. Cited on 60

Glover, F., Taillard, E., and Taillard, E. (1993). A user's guide to tabu search. *Annals of Operations Research*, 41:1–28. Cited on 75

Goh, E.-J. (2003). Secure indexes. Cryptology ePrint Archive, Report 2003/216. `http://eprint.iacr.org/2003/216/`. Cited on 21

Gossen, H. (1854). *Die Entwickelung der Gesetze des menschlichen Verkehrs, und der daraus flieenden Regeln für menschliches Handeln.* F. Vieweg. Cited on 33

Grandison, T. and Sloman, M. (2000). A survey of trust in internet applications. *IEEE Communications Surveys & Tutorials*, 3(4):2–16. Cited on 40

Gray, F. (1953). Pulse code communication. *US Patent 2,632,058.* Cited on 120

Guha, R., Kumar, R., Raghavan, P., and Tomkins, A. (2004). Propagation of trust and distrust. In *Proceedings of the 13th International Conference on World Wide Web*, WWW '04, pages 403–412, New York, NY, USA. ACM. Cited on 84

Gupta, A., Sahin, O. D., Agrawal, D., and Abbadi, A. E. (2004). Meghdoot: content-based publish/subscribe over P2P networks. In *Proceedings of the 5th ACM/IFIP/USENIX International Conference on Middleware*, Middleware '04, pages 254–273, New York, NY, USA. Springer-Verlag New York, Inc. Cited on 9

Hammond, P. J. (1976). Equity, Arrow's Conditions, and Rawls' Difference Principle. *Econometrica*, 44(4):pp. 793–804. Cited on 38

Huang, Y. and Garcia-Molina, H. (2003). Publish/subscribe tree construction in wireless ad-hoc networks. In *Mobile Data Management*, volume 2574 of *Lecture Notes in Computer Science*, pages 122–140. Springer Berlin/Heidelberg. Cited on 10, 11, 12, 13, 79, 120

Jaeger, M. A., Parzyjegla, H., Mühl, G., and Herrmann, K. (2007). Self-organizing broker topologies for publish/subscribe systems. In *Proceedings of the 2007 ACM Symposium on Applied Computing*, SAC '07, pages 543–550, New York, NY, USA. ACM. Cited on 120

Jayakumar, R., Thulasiraman, K., and Swamy, M. (1984). Complexity of computation of a spanning tree enumeration algorithm. *IEEE Transactions on Circuits and Systems*, 31(10):853–860. Cited on 54, 55, 56, 57

Kamvar, S. D., Schlosser, M. T., and Garcia-Molina, H. (2003). The Eigentrust algorithm for reputation management in P2P networks. In *Proceedings of the 12th international conference on World Wide Web*, WWW '03, pages 640–651, New York, NY, USA. ACM. Cited on 2, 41, 48

Klemm, K. and Eguiluz, V. (2002). Growing scale-free networks with small-world behavior. *Physical Review E*, 65(5):57102. Cited on 85

Knuth, D. E. (2011). *The Art of Computer Programming, Volume 4A: Combinatorial Algorithms, Part 1.* Addison Wesley Professional. Cited on 120

Kulturel-Konak, S., Norman, B. A., Coit, D. W., and Smith, A. E. (2004). Exploiting tabu search memory in constrained problems. *INFORMS Journal on Computing*, 16(3):241–254. Cited on 68, 69, 89, 95

Li, J., Lu, C., and Shi, W. (2004). An Efficient Scheme for Preserving Confidentiality in Content-Based Publish-Subscribe Systems. Technical report, Georgia Institute of Technology. Cited on 4, 18, 23

Marti, S., Ganesan, P., and Garcia-Molina, H. (2005). SPROUT: P2P routing with social networks. In *Current Trends in Database Technology - EDBT 2004 Workshops*, volume 3268 of *Lecture Notes in Computer Science*, pages 511–512. Springer Berlin/Heidelberg. Cited on 41, 43

Marti, S. and Garcia-Molina, H. (2006). Taxonomy of trust: Categorizing P2P reputation systems. *Computer Networks*, 50(4):472 – 484. Cited on 1, 119

Meier, R. and Cahill, V. (2002). Steam: Event-based middleware for wireless ad-hoc networks. In *Proceedings of the 22nd International Conference on Distributed Computing Systems Workshops*, pages 639–644. IEEE. Cited on 9

Migliavacca, M. and Cugola, G. (2007). Adapting publish-subscribe routing to traffic demands. In *Proceedings of the 2007 Inaugural International Conference on Distributed event-based systems*, DEBS '07, pages 91–96, New York, NY, USA. ACM. Cited on 120

Miklós, Z. (2002). Towards an access control mechanism for wide-area publish/subscribe systems. In *Proceedings of the 22nd International Conference on Distributed Computing Systems Workshops*, pages 516–521. IEEE. Cited on 24

Mill, J. S. (1863). *Utilitarianism*. Parker, Son and Bourne. Cited on 32

Minty, G. (1965). A simple algorithm for listing all the trees of a graph. *IEEE Transactions on Circuit Theory*, 12(1):120–120. Cited on 54

Mirkovic, J. and Reiher, P. (2004). A taxonomy of DDoS attack and DDoS defense mechanisms. *ACM SIGCOMM Computer Communication Review*, 34(2):39. Cited on 18

Mühl, G. (2002). *Large-Scale Content-Based Publish/Subscribe Systems*. PhD thesis, Berlin Institute of Technology. Cited on 24

Muhl, G., Fiege, L., Gartner, F., and Buchmann, A. (2003). Evaluating advanced routing algorithms for content-based publish/subscribe systems. In *Proceedings of the 10th IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems*, MASCOTS 2002, pages 167–176. IEEE. Cited on 9

Mühl, G., Fiege, L., and Pietzuch, P. (2006). *Distributed Event-Based Systems*. Springer. Cited on 26, 27

Nechvatal, J., Barker, E., Bassham, L., Burr, W., Dworkin, M., Foti, J., and Roback, E. (2001). Report on the development of the Advanced Encryption Standard (AES). *Journal of Research – National Institute of Standards and Technology*, 106(3):511–576. Cited on 27

Nonobe, K. and Ibaraki, T. (1998). A tabu search approach to the constraint satisfaction problem as a general problem solver. *European Journal of Operational Research*, 106(23):599 – 623. Cited on 69

Pesonen, L. I. W., Eyers, D. M., and Bacon, J. (2006). A capability-based access control architecture for multi-domain publish/subscribe systems. In *Proceedings of the International Symposium on Applications on Internet*, SAINT 2006, pages 222–228. IEEE. Cited on 4, 27

Pesonen, L. I. W., Eyers, D. M., and Bacon, J. (2007). Encryption-enforced access control in dynamic multi-domain publish/subscribe networks. In *Proceedings of the 2007 Inaugural International Conference on Distributed event-based systems*, DEBS '07, pages 104–115, New York, NY, USA. ACM. Cited on 4, 27

Pietzuch, P. and Bacon, J. (2002). Hermes: a distributed event-based middleware architecture. In *Proceedings of the 22nd International Conference on Distributed Computing Systems Workshops*, pages 611–618. IEEE. Cited on 9, 26

Raiciu, C. and Rosenblum, D. (2006). Enabling confidentiality in content-based publish/subscribe infrastructures. In *Proceedings of the Second IEEE/CreatNet International Conference on Security and Privacy in Communication Networks*, Securecomm '06, pages 1–11. IEEE. Cited on 4, 17, 21, 23

Rawls, J. (1971). *A Theory of Justice*. Belknap. Cited on 5, 32

Ray, I. and Chakraborty, S. (2004). A vector model of trust for developing trustworthy systems. In *Computer Security ESORICS 2004*, volume 3193 of *Lecture Notes in Computer Science*, pages 260–275. Springer Berlin/Heidelberg. Cited on 40

Resnick, P., Kuwabara, K., Zeckhauser, R., and Friedman, E. (2000). Reputation Systems. *Communications of the ACM*, 43(12):45–48. Cited on 1

Ribeiro, C. and De Souza, M. (2000). Tabu search for the Steiner problem in graphs. *Networks*, 36(2):138–146. Cited on 61, 63, 64

Robbins, L. C. (1935). *An Essay on the Nature and Significance of Economic Science*. Macmillan, London, Second edition. Cited on 33

Rosenblum, D. S. and Wolf, A. L. (1997). A design framework for Internet-scale event observation and notification. *ACM SIGSOFT Software Engineering Notes*, 22(6):344–360. Cited on 9

Sala, A., Zheng, H., Zhao, B. Y., Gaito, S., and Rossi, G. P. (2010). Brief announcement: revisiting the power-law degree distribution for social graph analysis. In *Proceedings of the 29th ACM SIGACT-SIGOPS Symposium on Principles of distributed computing*, PODC '10, pages 400–401, New York, NY, USA. ACM. Cited on 84

Sandhu, R., Coyne, E., Feinstein, H., and Youman, C. (1996). Role-based access control models. *Computer*, 29(2):38–47. Cited on 26

Sen, A. K. (1970). *Collective Choice and Social Welfare*. Holden-Day. Cited on 35, 36, 39

Sen, A. K. (1977). On weights and measures: Informational constraints in social welfare analysis. *Econometrica*, 45(7):pp. 1539–1572. Cited on 33, 35, 36

Sen, A. K. (1999). The possibility of social choice. *The American Economic Review*, 89(3):349–378. Cited on 33

Smith, M. J. (1997). Generating Spanning Trees. Master's thesis, University of Victoria. Cited on 120

Srivatsa, M. and Liu, L. (2005). Securing publish-subscribe overlay services with Eventguard. In *Proceedings of the 12th ACM conference on Computer and Communications Security*, CCS '05, pages 289–298, New York, NY, USA. ACM. Cited on 4, 20, 23

Takahashi, H. and Matsuyama, A. (1980). An approximate solution for the Steiner problem in graphs. *Mathematica Japonica*, 24(6):573–577. Cited on 61, 70

Talbi, E.-G. (2009). *Metaheuristics: From Design to Implementation*. John Wiley and Sons. Cited on 60

Tarkoma, S. (2006). Preventing spam in publish/subscribe. In *26th IEEE International Conference on Distributed Computing Systems Workshops*, ICDCSW 2006, pages 21–21. IEEE. Cited on 28

Theodorakopoulos, G. and Baras, J. (2006). On trust models and trust evaluation metrics for ad hoc networks. *IEEE Journal on Selected Areas in Communications*, 24(2):318–328. Cited on 41, 68

Triantafillou, P. and Aekaterinidis, I. (2004). Content-based publish-subscribe over structured P2P networks. *IEE Seminar Digests*, 2004(918):104–109. Cited on 9

Wang, C., Carzaniga, A., Evans, D., and Wolf, A. (2002). Security issues and requirements for internet-scale publish-subscribe systems. In *Proceedings of the 35th Annual Hawaii*

International Conference on System Sciences, HICSS '02, pages 3940 – 3947. IEEE. Cited on 4, 10, 15, 20

Wun, A., Cheung, A., and Jacobsen, H.-A. (2007). A taxonomy for denial of service attacks in content-based publish/subscribe systems. In *Proceedings of the 2007 Inaugural International Conference on Distributed event-based systems*, DEBS '07, pages 116–127, New York, NY, USA. ACM. Cited on 4, 18

Xiong, L. and Liu, L. (2004). Peertrust: Supporting reputation-based trust for peer-to-peer electronic communities. *Knowledge and Data Engineering, IEEE Transactions on*, 16(07):843–857. Cited on 2

Yager, R. (1997). On the analytic representation of the Leximin ordering and its application to flexible constraint propagation. *European Journal of Operational Research*, 102(1):176–192. Cited on 6, 38

Zhu, Y. and Hu, Y. (2005). Ferry: an architecture for content-based publish/subscribe services on P2P networks. In *Proceedings of the 2005 International Conference on Parallel Processing*, ICPP 2005. IEEE. Cited on 9

Zimmermann, P. R. (1995). *The official PGP user's guide*. MIT Press, Cambridge, MA, USA. Cited on 41