HAL
archives-ouvertes.fr

Finding Trusted Publish/Subscribe Trees

Stephen Naicken, Ian Wakeman, Dan Chalmers

▶ **To cite this version:**

Stephen Naicken, Ian Wakeman, Dan Chalmers. Finding Trusted Publish/Subscribe Trees. 6th International Conference on Trust Management (TM), May 2012, Surat, India. pp.174-190, 10.1007/978-3-642-29852-3_12 . hal-01517643

HAL Id: hal-01517643
https://hal.inria.fr/hal-01517643

Submitted on 3 May 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution| 4.0 International License

Finding Trusted Publish/Subscribe Trees

Stephen Naicken, Ian Wakeman, and Dan Chalmers

Department of Informatics,
University of Sussex,
Brighton,
UK
Initial.Lastname@sussex.ac.uk

Abstract Publish/Subscribe systems assume that clients and brokers abide by the matching and forwarding protocols. Such an assumption implies implicit trust between all components of the system and has led to security issues being largely ignored. As publish/subscribe is increasingly used in applications where implicit trust can not be assumed, an approach is required to mitigate misbehaviour. We propose the construction and reconfiguration of the event forwarding topology, the publish/subscribe tree (PST), with respect to the trust requirements of the clients. The principal contribution of this paper is a trust metric for PSTs, which aggregates each client's trust evaluation of a PST to give a socially acceptable trust evaluation and allows for the ordering of PSTs. Additionally, we define the PST trust maximisation problem with overhead budget, which is solved by the PST that maximises trust within an overhead budget for a given advertisement. A tabu search based algorithm for this problem is presented and is shown to scale to large problem instances and give good approximations of the optimal solutions.

1 Introduction

Publish/Subscribe systems assume that brokers are implicitly trusted to correctly implement the matching and routing functions that are essential for the delivery of events from publishers to subscribers. Trust between clients (publishers and subscribers) and brokers is ensured by the presence of trusted administrative entities responsible for the event notification service (ENS) – the network of brokers responsible for propagation of events from publishers to subscribers – and external contracts between clients and administrative entities. Typically, publish/subscribe is used in application contexts where these mechanisms exist, for example news distribution services that are restricted to fee paying customers, but increasingly publish/subscribe is being utilised for applications where administrative entities and external contracts may not be present. In publish/subscribe based inter-networking, the scale and dynamicity of the network prohibit the use of external contracts, while mobile ad-hoc network publish/subscribe has the additional issue of the absence of an ENS under the aegis of trusted entities. These

applications are vulnerable to misbehaviour that can disrupt communications, as mechanisms to ensure trust are absent.

Motivated by research on the use of trust and reputation systems to safeguard Peer-to-Peer networks, this paper proposes a mechanism by which the publish subscribe tree (PST) that is used to distribute events from publishers and subscribers can be constructed and reconfigured to maximise the clients' trust of the PST. A trust metric for PSTs is defined that aggregates each and every client's trust evaluation of a PST in an equitable manner. Following from Rawls' difference principle, given a set of PSTs, the metric deems the most trusted PST to be the one that maximises the lowest trust opinion held by any client. The PST trust metric is used to define the PST Trust Maximisation problem with overhead budgets, which is solved by the PST that maximises trust within a prescribed overhead budget. An exhaustive search and tabu search algorithm to solve this problem are presented and evaluated. By maintaining, a PST that maximises trust given the clients' trust opinions, the PST is less vulnerable to misbehaviour and consequently service disruption. Brokers that are deemed by clients to be untrustworthy will be not be included in the initial PST construction or will be likely ejected upon PST reconfiguration.

The remainder of this paper is structured as follows. In section 2, the PST is defined and a PST overhead metric is presented. Section 3 details the trust metric for PSTs. The aggregation is underpinned by Rawls' difference principle to ensure that the aggregation is equitable to all clients. The definition of the PST trust maximisation problem with overhead budget (MTPSTO) is given in section 4. In section 5, an algorithm using the tabu search metaheuristic to solve the MTPSTO problem is presented. Finally, section 6 describes the evaluation of the tabu search algorithm with particular emphasis on the comparison of the results with those of an exhaustive search algorithm.

2 Publish/Subscribe Trees

2.1 Definition of Publish/Subscribe Trees

In publish/subscribe systems, publishers issue advertisements to the ENS that express their intent to publish events consisting of particular content (e.g. Java tutorial books for sale that are less than ten pounds). Subscribers submit subscriptions (e.g. Java tutorial books for sale) that express their interest to receive specific events to the ENS. Should a subscription match an advertisement owned by a given publisher, the ENS must ensure that matching events issued by this publisher are delivered to the interested subscriber.

As the ENS is a network of interconnected brokers, a forwarding topology is required that allows for the dissemination of events from publishers to subscribers. The topology is typically an acyclic graph, so it can be modelled as a tree. Definition 1 defines this tree as a publish/subscribe tree (PST). It exists in the context of an advertisement. For each advertisement, there is a PST that is rooted at publisher of the advertisement and spans a subset of brokers and all interested subscribers.

PSTs can also be used to model ad-hoc network publish/subscribe where there is no ENS and both brokers and subscribers may be responsible for the matching and forwarding of events. In this context, the set of internal nodes of a PST can include subscribers.

Definition 1 (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 Publish/Subscribe Overheads

PST construction with respect to overhead costs was first considered by Huang and Garcia-Molina in the context of publish/subscribe in wireless ad-hoc networks [8]. They define three types of subscription: inherent subscription; effective subscription; proxied subscription. The inherent subscription s_i of a subscriber i is given by its subscription function sf_i. The effective subscription S_i of a subscriber i is given by the disjunction of its inherent subscription s_i and its proxied subscription $s_i^{'}$, $S_i = s_i \vee s_i^{'}$. 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. The overhead metric is defined with respect to these subscription types in definition 2.

Definition 2 (Publish/Subscribe Tree Overhead). *The overhead of a PST T, $C_T(E)$ is $O_T(E) = \sum_i O_{T_i}(E)$ where E is the set of events to be published and $O_{T_i}(E)$ is the overhead of receiving, processing and forwarding the events in E at node i of T. The overhead O_{T_i} at a node i, is given by $O_{T_i}(E) = (r + f) \cdot \Phi E(\neg s_i \wedge s_i^{'}) + f \cdot \Phi E(s_i \wedge s_i^{'})$ where s_i is the subscription function at node i, $s_i^{'}$ is the proxied subscription of i, and $\Phi E(\alpha)$ gives the number of events from the set E that match the subscription function α.*

3 Theory

The aim of this section is to show how PSTs can be compared in terms of the trust imbued in the tree by different participants. A generalised trust metric that combines trust values of the nodes on a path, and allows for discrimination of the trust of two disparate paths, is given. Subsequently, the relationships between the participants in a PST are identified, leading to a natural formulation of the problem as an analysis of the trust within and between different paths for a given individual participant. Having defined a means for participants to order their preferences for different PSTs, the problem of how to aggregate these orderings is addressed by the use of social choice theory, albeit with a caveat about the comparability of trust functions.

3.1 Definition of Trust

Trust is "the firm belief in the competence of an entity to act dependably, securely and reliably within a specified context" [7]. The competence of a trustee is dependent upon a variety of trust sources with their importance to trust evaluation dependent upon the trustor. Vector-based trust models have been proposed that aggregate a vector of trust sources to give a single trust opinion of an entity [19]. A generalisation of this model is defined as follows and is used in this work.

Definition 3. ***(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 4. ***(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.

3.2 Trustworthiness of Paths

Using previous work on semiring-based trust models [23], we define our trust algebra as the set S, with two binary operators, $\oplus$ and $\otimes$. S contains the individual entities, with the level of trust. The $\otimes$ operator combines the entities into a path and returns the level of trust of that path, whilst the $\oplus$ operator compares paths and picks out the path with the maximum trust. We assume that $\oplus$ is commutative, and that $\otimes$ is distributive over $\oplus$, and further that there is a partial order over the operators such that $a \leq a' \wedge b \leq b' \Rightarrow a \oplus b \leq a' \oplus b' \wedge a \otimes b \leq a' \otimes b'$.

Mathematically inclined readers may note that we have defined an ordered semiring. An example of such an algebra is the trusted path definitions adopted by Marti and Garcia-Molina in [12], an instantiation of which is given in definition 5. We will be using this instantiation, but results remain valid for any other valid instantiation.

Definition 5. ***(Trusted Path Semiring)*** *The trusted path semiring is a semiring,* $(S, \oplus, \otimes)$ *where* $S = [0, 1]$ *and* $\oplus$ *and* $\otimes$ *are defined as:*

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

Example 1. **(Example Use of Trusted Path Semiring)** Let σ_1 be a simple path, $\sigma_1 = (v_1, v_2), (v_2, v_3), \ldots, (v_{n-1}, v_n)$, where v_1 and v_{n+1} are the start and end vertex of the path, respectively. The trust v_1 has in σ_1 is given by $\tau(v_1, v_2) \otimes \tau(v_1, v_3) \otimes ... \otimes \tau(v_1, v_{n+1})$ 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 $\tau_{\sigma_1} \oplus \tau_{\sigma_2} \oplus ... \oplus \tau_{\sigma_p} = max(\tau_{\sigma_1}, \tau_{\sigma_2}, ..., \tau_{\sigma_p})$.

3.3 Individual Trust Evaluation Functions for PSTs

Using the individual trust function (def. 4) and the trusted path semiring (def. 5), the clients' trustworthiness of a PST can be defined. The relationships between publishers, internal subscribers and terminal subscribers are identified and evaluated using the aforementioned methods to give individual trust functions for these types of clients. The definitions below define how the trustworthiness of a PST is evaluated by the publisher, the internal subscribers and the terminal subscribers.

Publisher PST Trust Evaluation Function The publisher has a contract to send events to each subscriber in the PST and has no preference over these subscribers. To ensure delivery of these events, the trust function should maximise the trustworthiness of each path to the subscribers. The publisher trust function aggregates the trust of each path to each subscriber in the PST. Where the publisher is adjacent to a subscriber, the trust value of the path is 1, as all notifications are sent directly to the subscriber.

Definition 6. ***(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:*

$$\tau_p(\sigma_{p,s}) = \tau_p(\Lambda^{\eta}_{p,v_1}) \otimes \tau_p(\Lambda^{\eta}_{p,v_2}) \otimes \cdots \otimes \tau_p(\Lambda^{\eta}_{p,v_{|\sigma|-1}}) \tag{1}$$

The trust of T for p is a function of the trust of the paths to each subscriber and is given by:

$$\tau_p(T) = \alpha(\tau_p(\sigma_{p,s_1}), \tau_p(\sigma_{p,s_2}), \ldots, \tau_p(\sigma_{p,s_{|S|}})) \tag{2}$$

Terminal Subscriber PST Evaluation Function Terminal subscribers receive events forwarded on the path from the publisher. Their trust in the PST is determined exclusively by the trust of this path. If the subscriber is adjacent to the publisher, its trust value of the path is 1, as it trusts the publisher to receive its events.

Definition 7. ***(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 the trust of s in T is given by:*

$$\tau_s(T) = \tau_s(\Lambda^{\eta}_{s,v_1}) \otimes \tau_s(\Lambda^{\eta}_{s,v_2}) \otimes \cdots \otimes \tau_s(\Lambda^{\eta}_{s,v_{|\sigma_{s,p}|-1}}) \tag{3}$$

Internal Subscriber PST Evaluation Function An internal subscriber receives events on the path from the publisher and forwards them to subscribers in the sub-tree of which it is the root. The node holds both the roles of terminal subscriber and publisher, so its trust function is a function of the trust to the publisher and the trustworthiness of the paths to each subscribers in its sub-tree.

Definition 8. ***(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:*

$$\tau_s(\sigma_{s,p}) = \tau_s(\Lambda^{\eta}_{s,v_1}) \otimes \tau_s(\Lambda^{\eta}_{s,v_2}) \otimes \cdots \otimes \tau_s(\Lambda^{\eta}_{s,v_{|\sigma|-1}}) \tag{4}$$

Additionally, for each $s \in S$ such that s is an internal node, let $T_s = (V_s, E_s)$ be the sub-tree 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:

$$\tau_s(\sigma_{s,s'}) = \tau_s(\Lambda^{\eta}_{s,v_1}) \otimes \tau_s(\Lambda^{\eta}_{s,v_{|\sigma|-1}}) \otimes \tau_s(\Lambda^{\eta}_{s,s'}) \tag{5}$$

For each internal subscribe node s in a PST T, the trust of s in T is given by:

$$\tau_s(T) = \beta(\tau_s(\sigma_{s,p}), \tau_s(\sigma_{s,s'_1}), \ldots, \tau_s(\sigma_{s,s'_{d-1}})) \tag{6}$$

where $\beta : \mathbb{R}^d \longrightarrow \mathbb{R}$ is some aggregation function of trust values, and $d = |V_s \cap S| + 1$.

3.4 PST Trust Evaluation Function

Social Choice and Welfare Preliminaries 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. Here it is used to address the following problems: the aggregation of the trust evaluation of paths to give a client's trust evaluation of a PST; and the aggregation of the clients' PST trust evaluation functions to give the trustworthiness of a PST.

Sen [21] shows that if we can assign utility values to the preferences, and that the utilities are comparable between individuals, then a choice function is usable. As in nearly all other work building on trust preferences, we must assume that the utility of our trust preferences can be compared, both in deciding which individual is worse off (*ordinal level comparability*), and how much one gains when another loses (*cardinal level comparability*).

Rather than adopt a utilitarian approach to determining the most trusted PST, the PST that maximises the trustworthiness of the least well-off node is dominates the trust metric. This is motivated by Rawls' difference principle [18], which 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.

Definition 9. ***(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 10. ***(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 $|C| = i - 1$ individuals such that for all $c \in C$, $u_i(x) \geq u_c(x)$. 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. $u_{c(x)}(x) = u_{c(y)}(y)$ *where* $c \in \{n : n \in N \wedge u_i(x) \geq u_n(x)\}$.

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

Aggregation of Path Trust Evaluations in Individual PST Trust Functions The aggregation functions α and β in definitions 6 and 8, are the leximin aggregation function given in definition 11 and the minimum aggregation function, respectively. As d - the number of paths - is variable across PSTs, for internal subscribers, β can not make use of analytical leximin aggregation (definition 11). The motivation for the choice of these aggregations is to allow a client's trust in a PST to be dominated by the least trustworthy path. Consider an internal subscriber in a PST with paths of very high trust, except for a path of low trust to a terminal subscriber. There is more risk of malicious behaviour on this path and the terminal is undeservedly punished in favour of others, as all subscribers should be treated equitably. This is a scenario our metric attempts to avoid.

PST Trust Evaluation Function 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. Rather than implement the leximin social welfare function as one of pairwise comparisons, an analytical leximin function (def. 11) is used. This allows PSTs to be ordered by their trust values and a combinatorial optimisation problem that maximises the trust value of the PST for a given advertisement to be defined. 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.

Definition 11. ***(Analytical Leximin Aggregation)*** *The analytical leximin aggregation operator, $F_{leximin}$, is an ordered weighted average where each $a_i \in [0, 1]$ and 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* $|a-b| > \Delta$.

Definition 12. ***(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* $F_{leximin}(\tau_{i_1}(T), \tau_{i_2}(T), \ldots, \tau_{i_{|S\cup\{p\}|}}(T))$.

4 The Problem

4.1 The PST Trust Maximisation Problem with Overhead Budget

Given this definition of aggregated social trust, we are now in a position to formally define the problem of maximising the trust of a Publish/Subscribe Tree that meets an overhead budget in terms of the cost of its links - the PST Trust Maximisation Problem with Overhead Budget (MTPSTO):

Problem 1 (The MTPSTO problem). 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 $O_T(Ev) \leq B$.

The MTPSTO decision problem is shown to be in NP-hard by a polynomial time reduction from the Minimum Overhead PST Problem [2] and in NP by a polynomial time verification algorithm. We omit the proof due to space constraints.

5 The Algorithm

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 an advertisement A_p. The set of all PSTs for A_p is a subset of the set of all Steiner trees in G_c that span p and S. Using this property and the fact that the set of all Steiner trees in G_c is given by the enumeration of all spanning trees in G_c and all its sub-graphs, the algorithm must find all the spanning trees in G_c and all its sub-graphs that are also feasible PSTs. Note that graphs and sub-graphs with router vertices with only one adjacent edge are ignored, as all spanning trees found will not be PSTs. The router will be a terminal node in every PST that spans the graph and this contradicts the definition of a PST.

5.1 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 have $\mathcal{O}(m+n+mt)$ [14] and $\mathcal{O}(m+n+nt)$ [5] complexity for undirected graphs, where t is the number of spanning trees. Prior to the these techniques, Char [3] proposed an algorithm that lexicographically tests sub-graphs to determine if each is a spanning tree, and although a complexity analysis was not given, it was later shown to be of $\mathcal{O}(m + n + n(t + t_0))$ complexity where t_0 is the number of sub-graphs found that are not spanning trees [9]. Char's algorithm is shown to be more suitable for enumerating PSTs, as the spanning tree test of a sub-graph can be modified to determine if it is a PST.

5.2 Approximation through Tabu Search

A number of approaches to the Steiner problem in graphs that use the tabu search metaheuristic have been proposed [20] [6]. The Ribeiro and De Souza [20] approach finds solutions that are better than the Takahashi-Matsuyama heuristic [22] and F-tabu [6]. Given the relationship between Steiner trees and PSTs, and the successful use of tabu search to solve the Steiner problem, this metaheuristic is explored as means to solve the MTPSTO problem.

PST Tabu Move Selection and Evaluation Similar to the move structure defined in [20], a tabu search move is defined as the addition or removal of a broker 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 broker nodes is variable, hence the choice of move structure. For insertion moves, a broker node and its edges (from the connectivity graph), which are adjacent to nodes in the PST, are added to the PST. For removal moves, a broker is removed from the PST and every edge in the connectivity graph between pairs of nodes in the PST are added to the PST.

In tabu search, the application of a move to an current PST solution gives a new solution, however the application of a move to a PST gives a sub-graph of the connectivity graph. To address this, the modified spanning tree algorithm (section 5.1) is used to find all PST in the sub-graph. Each PST is evaluated for its trust and overhead value. The PST that maximises trust and is below budget is then selected as the tree that is derived as a result of the application of the move to the current PST solution. If no tree is under budget, the one with the highest trust value is chosen.

Penalty Function Tabu search is designed for minimisation and maximisation combinatorial problems without constraints, however the MTPSTO problem has an overhead budget constraint within which trust is maximised. Although a Near Feasibility Threshold (NFT) technique [11] was investigated, superior results

were obtained with a static penalty function. In this approach, all over-budget PSTs are penalised by increasing the trust value by 50%.

Diversification Strategy Takahashi and Matsuyama [22] present a Steiner tree heuristic that can easily be modified to find a Steiner tree that is a PST. The modifications required is to stipulate that the subscribers and the publisher are the Steiner nodes. 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.

Surrogate Objective Function Each move when applied to a PST gives a number new solutions that must be evaluated for their trust and overhead values (objective values). Evaluation of each move can become costly as the size of the problem instance increases. To address this, the use of a surrogate evaluation function is proposed that estimates the trust value of solutions derived from a move. Moves that are likely to not result in an improvement over the current solution are discarded. The solutions derived from the smaller moves set are then fully evaluated for exact objective values.

A greedy approach is adopted for the surrogate objective function, as it attempts 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, it is possible to to use the generic shortest distance algorithm algorithm defined in [23] to find the most trusted path between two nodes.

Tabu Search Algorithm for MTPSTO The tabu search algorithm for the MTPSTO begins by using the diversification method to find the initial PST solution and setting to the current solution. Its objective value is evaluated and the tabu search iterates until the maximum number of iterations without an improvement in the objective value is met. During each iteration, first the set of moves is established by determining the routers that can be added and removed from the PST. Moves that are in the tabu list are discarded. Using the surrogate objective function, the move set is further reduced to the best estimated insertion and removal moves. The modified Char spanning tree algorithm is used to enumerate all PSTs resulting from the application of the moves in the move set to the current solution. These PSTs are then evaluated for their trust and overhead values. The PST with the highest trust value and lowest overheads is selected as the new current solution. If it is also better than the existing best solution found by the algorithm so far, then it is set as the new best solution. The move that yields the new current solution is marked is places in the tabu list and will not be available for selection for a given number of iterations.

6 Evaluation

In this section, an evaluation of the exhaustive search and the tabu search algorithms for the MTPSTO problem are presented. The evaluation is concerned with two properties, the quality of the solutions found and the running times of the algorithm. Solution quality is given by the relative error of the trust and overhead values with respect to the optimal solution.

The algorithms were implemented using Java and are dependent upon two third-party libraries, the Java Universal Network/Graph Framework (JUNG) (ver. 2.01) and the OpenTS library (ver. 1.0-exp10). JUNG is a framework 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 and the running times given in the results tables are averages over these executions unless stated otherwise. Experiments were performed on Amazon EC2 using a High-Memory Extra Large instance (m2.xlarge). The instance has 17.1 Gb of RAM, two virtual cores with 3.25 EC2 Compute Units reported as two 2.67 GHz Intel Xeon X5550 CPUs by `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 and the Java runtime environment used was IcedTea6 1.9.1. 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 requirements of the exhaustive search algorithm. To ensure fair comparability of the running times, the same instance type was used for the tabu search algorithm, despite its lower memory usage.

6.1 Evaluation Test Data Sets

The test data sets are comprised of problem instances varying in $|R|$, 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 graph density of all problem instances is approximately equal to 0.5. By increasing the number of routers in each problem and maintaining constant graph density, 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.

Test data sets are made of subsets of five problems, each problem sharing identical parameters other than the value of the overhead budget, B. 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 and B), `<Subset Number>` indicates the value of $|R|$ for all problem instances 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). 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 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 $B = 5000$. The choice of $B = 2^{31} - 1$ is so that the algorithms can find the most trusted PST within the largest permitted integer overhead budget.

Problem set A 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. Problem set B consists of problems where $20 \leq |R| \leq 100$. Although no exact solutions known for these problems, the results are useful for evaluating the scalability of the tabu search algorithm.

6.2 Results

Table 1 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 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.

Table 1. Execution Times of Exhaustive Search Results for 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	N/A

Tables 2 and 3 give the results for the tabu search algorithms for problem sets A and B, respectively. The running times for problem subsets A1 to A4 are

inferior to those of the exhaustive search. For problem subset A5 and above, the tabu search outperforms the exhaustive search algorithm with respect to execution time. For only seven problems in problem set A, the tabu search does not find the exact solution. Of these, four have negligible error to the optimal solution in the trust value of the PST. The average relative error in the overhead costs is 0.1148.

For problem set B, no exact solutions are available due to these problem instances being of too large for an exhaustive search. However, the results show that even for large problem instances, the tabu search algorithm is capable of finding solutions in running times that are considerably faster than those of the exhaustive search. The slowest time, 88.44s for problem B30-4, is some three times faster than that of an exhaustive search for problem instance consisting of six routers.

In conclusion, it has been shown that the tabu search scales to large problem instances with running times comparable to those of the exhaustive search algorithm for significantly smaller instances. The results for problem set A have demonstrated that the tabu search is capable of finding good approximation solutions.

7 Discussion

For a PST to form, the subscribers and routers must trust the publisher, so it is natural to devolve responsibility for the creation and selection of the PST to the publisher. Our protocol thus degenerates to the collection of trust vectors by the publisher for the candidate nodes in the tree, the execution of the tabu search algorithm, and finally the notification of the selected nodes of their roles and routing tables.

We have provided a formal definition of how trust can be used to evaluate the worth of a PST, dependent upon position, and under the assumption of full cardinal comparability of the trust metrics. Using this metric and existing work on PST overheads, we have shown how to derive optimal and near-optimal trees which maximise trust and meet a link cost budget. Under what circumstances does this assumption hold?

Trust can be formed using a number of trust sources, but typically in the literature, it is a function of reputation. 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 [10], 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 file from k and rates the transaction as 0, perhaps due to having a higher tolerance of malicious behaviour. When calculating trust values, it is therefore not possible to state that nodes i and j holding trust values of 0.7 in some entities are comparable, as their perceptions and understanding of trust and consequently their trust ratings of others differ.

Table 2. Solutions for Problem Set A using the Tabu Search algorithm

	PST		Rel. Error		
Pr	τ_T	O_T	η_τ	η_O	Sec
A1-2	0.0181	2398	-	-	3.01
A1-3	0.0181	2398	-	-	3.02
A1-4	0.0181	2398	-	-	3.01
A1-5	0.0181	2398	-	-	3.00
A2-1	0.0931	1850	-	-	8.44
A2-2	0.0931	1850	-	-	8.49
A2-3	0.0931	1850	-	-	8.40
A2-4	0.0931	1850	-	-	8.37
A2-5	0.0931	1850	-	-	8.36
A3-2	0.0224	2917	-	-	11.12
A3-3	0.0224	2917	-	-	11.12
A3-4	0.0224	2917	-	-	11.03
A3-5	0.0224	2917	-	-	11.06
A4-2	0.1855	2224	-	-	7.28
A4-3	0.1855	2224	-	-	7.21
A4-4	0.1855	2224	-	-	7.20
A4-5	0.1855	2224	-	-	7.21
A5-2	0.0542	2262	-	-	13.63
A5-3	0.0812	3580	-	0.1202	8.26
A5-4	0.0812	3580	-	0.1202	8.24
A5-5	0.0812	3580	-	0.1202	8.22
A6-3	0.0360	3846	-	-	139.19
A6-4	0.0360	3846	5×10^{-7}	0.1287	138.96
A6-5	0.0360	3846	5×10^{-7}	0.1287	127.22
A7-2	0.0692	3570	-	-	70.95
A7-3	0.0692	3570	-	-	72.92
A7-4	0.0692	3570	-	-	78.38
A8-3	0.0031	3657	-	-	9.77
A8-4	0.0031	3657	1×10^{-6}	0.0928	9.77
A8-5	0.0031	3657	1×10^{-6}	0.0928	9.82
A9-1	0.2184	1885	-	-	20.39
A9-2	0.2184	1885	-	-	14.69
A9-3	0.2184	1885	-	-	20.55
A9-4	0.2184	1885	-	-	20.49
A9-5	0.2184	1885	-	-	20.51

Table 3. Solutions for Problem Set B using the Tabu Search algorithm

	PST				PST		
Pr	τ_T	O_T	Sec	Pr	τ_T	O_T	Sec
B20-1	0.1210	2948	42.00	B30-1	0.1329	2234	57.19
B20-2	0.1210	2948	41.97	B30-2	0.1329	2234	61.82
B20-3	0.1210	3254	36.33	B30-3	0.1329	2234	72.58
B20-4	0.1210	3254	33.76	B30-4	0.1329	2234	88.44
B20-5	0.1210	3254	33.73	B30-5	0.1329	2234	84.46
B40-1	0.0245	2564	56.52	B50-1	0.0124	2224	18.96
B40-2	0.0245	2564	60.04	B50-2	0.0124	2224	18.87
B40-3	0.0245	2564	50.73	B50-3	0.0124	2224	18.70
B40-4	0.0245	2564	50.77	B50-4	0.0124	2224	19.70
B40-5	0.0245	2564	50.81	B50-5	0.0124	2224	19.96
B60-1	0.0661	1630	9.86	B70-1	0.0381	2838	30.00
B60-2	0.0661	1630	9.98	B70-2	0.0381	2838	29.99
B60-3	0.0661	1630	9.82	B70-3	0.0381	2838	46.44
B60-4	0.0661	1630	9.89	B70-4	0.0381	2838	46.77
B60-5	0.0661	1630	9.91	B70-5	0.0381	2838	45.85
B80-1	0.1320	1962	17.84	B90-1	0.0354	1282	11.56
B80-2	0.1320	1962	13.54	B90-2	0.0354	1282	11.59
B80-3	0.1320	1962	13.56	B90-3	0.0354	1282	11.59
B80-4	0.1320	1962	13.55	B90-4	0.0354	1282	11.57
B80-5	0.1320	1962	13.57	B90-5	0.0354	1282	11.57

Even when two entities have the same understanding of trust, they may assign different values to to trustees. This also applies to the trust valuations of alternatives under consideration. If the trust continuum is the unit interval, ordinal level comparability feasible, as an alternative can be rated as trusted, untrusted and indifferent or uncertain (a distinction can not be made), but the presence of an origin alone does not imply cardinal full comparability, a scale is required too. Trust can not, when represented using quantitatively, be interpersonally comparable, unless there is agreement about the meaning of the scale of trust metrics, i.e. is there must be an accepted definition of a unit of trust.

But we do not form a single PST. Instead we will repeat the collection of trust vectors and the evaluation of trees *ad infinitum*, as the publishers and subscribers in the network change. We postulate that the dominant strategy within this repeated game is to converge to a common understanding of the range of trust values, and to be truthful about the levels of trust, allowing for full cardinal comparability. We hope to demonstrate this to be true in future work.

8 Related Work

Wang [24] produced one of the earliest descriptions of the security issues in publish/subscribe networks, which was then built upon by Raicu in [17] to provide a formal definition of confidentiality in content based publish/subscribe. Miklós describes a method to define access control policies on clients' advertisement and subscription filters [13], which assumes that the infra-structure is trusted. Fiege et al. [4] attempted to address the level of trust between publishers, subscribers and infra-structure through the development of scopes of visibility. An implementation on the REBECA [15] shows how such a scoping approach might work. An alternative approach using Role Based Access Control (RBAC) is demonstrated by Belokosztolski et al. [1], building on the HERMES middleware [16]. Policies have been used to control the tree construction by Wun [25].

9 Conclusion

We have presented an algorithm for evaluating trust in publish / subscribe trees, where assumptions about the participants are weaker than in prior work. This algorithm is based on our own trust metric combined with an overhead metric from [8] in order to maximise the trust in the tree with respect to both producers and consumers with respect to a given budget. In addition we present a Tabu-based approximation which is significantly more efficient.

References

1. Belokosztolszki, A., Eyers, D., Pietzuch, P., Bacon, J., Moody, K.: Role-based access control for publish/subscribe middleware architectures. In: Proc. 2nd Intl. workshop on Distributed event-based systems. pp. 1–8. ACM (2003)

2. Cao, X., Shen, C.: Subscription-aware publish/subscribe tree construction in mobile ad hoc networks. In: Intl. Conf. on Parallel and Distributed Systems. vol. 2, pp. 1–9. IEEE (2009)
3. Char, J.: Generation of trees, two-trees, and storage of master forests. Circuit Theory, IEEE Transactions on 15(3), 228–238 (1968)
4. Fiege, L., Zeidler, A., Buchmann, A., Kilian-Kehr, R., Mühl, G.: Security aspects in publish/subscribe systems. In: In 3rd Intl. Workshop on Distributed Event-based Systems (DEBS04. pp. 44–49. Citeseer (2004)
5. Gabow, H.N., Myers, E.W.: Finding All Spanning Trees of Directed and Undirected Graphs. SIAM Journal on Computing 7(3), 280 (1978)
6. Gendreau, M., Larochelle, J., Sanso, B.: A tabu search heuristic for the Steiner tree problem. Networks 34(2), 162–172 (Sep 1999)
7. Grandison, T., Sloman, M.: A survey of trust in internet applications. IEEE Communications Surveys & Tutorials 3(4), 2–16 (2000)
8. Huang, Y., Garcia-Molina, H.: Publish/subscribe tree construction in wireless adhoc networks. In: Mobile Data Management. pp. 122–140. Springer (2003)
9. Jayakumar, R., Thulasiraman, K., Swamy, M.: Complexity of computation of a spanning tree enumeration algorithm. Circuits and Systems, IEEE Transactions on 31(10), 853–860 (Oct 1984)
10. Kamvar, S., Schlosser, M., Garcia-Molina, H.: The eigentrust algorithm for reputation management in P2P networks. In: Proc. of the 12th Intl. Conf. on World Wide Web. pp. 640–651. ACM (2003)
11. Kulturel-Konak, S., Norman, B., Coit, D.W., Smith, A.E.: Exploiting Tabu Search Memory in Constrained Problems. INFORMS Journal on Computing 16(3), 241–254 (Jun 2004)
12. Marti, S., Ganesan, P., Garcia-Molina, H.: SPROUT: P2P Routing with Social Networks. In: Current Trends in Database Technology-EDBT 2004 Workshops. pp. 511–512. Springer (2005)
13. Miklós, Z.: Towards an access control mechanism for wide-area publish/subscribe systems. In: Proc. 22nd Intl. Conf. on Distributed Computing Systems Workshops. pp. 516–521. IEEE (2002)
14. Minty, G.: A Simple Algorithm for Listing All the Trees of a Graph. Circuit Theory, IEEE Transactions on 12(1), 120–120 (1965)
15. Mühl, G.: Large-Scale Content-Based Publish/Subscribe Systems. Ph.D. thesis, Berlin Institute of Technology (2002)
16. Pietzuch, P., Bacon, J.: Hermes: a distributed event-based middleware architecture. Proc. 22nd Intl. Conf. on Distributed Computing Systems Workshops pp. 611–618 (2002)
17. Raiciu, C., Rosenblum, D.S.: Enabling Confidentiality in Content-Based Publish/Subscribe Infrastructures. In: 2006 Securecomm and Workshops. pp. 1–11. IEEE (Aug 2006)
18. Rawls, J.: A theory of justice. Oxford University Press, second edn. (1971)
19. Ray, I., Chakraborty, S.: A Vector Model of Trust for Developing Trustworthy Systems. Computer Security ESORICS 2004 pp. 260–275 (2004)
20. Ribeiro, C.C., De Souza, M.C.: Tabu search for the steiner tree problem in graphs. Networks 36, 138–146 (2000)
21. Sen, A.: Interpersonal aggregation and partial comparability. Econometrica: Journal of the Econometric Society 38(3), 393–409 (1970), http://www.jstor.org/stable/1909546
22. Takahashi, H., Matsuyama, A.: An approximate solution for the Steiner problem in graphs. Math. Japonica 24(6), 573–577 (1980)

23. Theodorakopoulos, G., Baras, J.: On trust models and trust evaluation metrics for ad hoc networks. Selected Areas in Communications, IEEE Journal on 24(2), 318–328 (Feb 2006)
24. Wang, C., Carzaniga, A., Evans, D., Wolf, A.: Security issues and requirements for Internet-scale publish-subscribe systems. In: Proc. of the 35th Annual Hawaii Intl. Conf. on System Sciences (HICSS). pp. 3940–3947. IEEE (2002)
25. Wun, A., Jacobsen, H.A.: A policy management framework for content-based publish/subscribe middleware. In: Cerqueira, R., Campbell, R. (eds.) Middleware 2007, LNCS, vol. 4834, pp. 368–388. Springer Berlin / Heidelberg (2007)