Trust-Based Service Management for Service-Oriented Mobile Ad Hoc Networks and Its Application to Service Composition and Task Assignment with Multi-Objective Optimization Goals

Yating Wang

Dissertation submitted to the faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
In
Computer Science and Applications

Committee:
Ing-Ray Chen (Chair)
Chang-Tien Lu
Wenjing Lou
Jin-Hee Cho
Ananthram Swami

April 25, 2016
Falls Church, Virginia

Keywords: Trust, logistic regression, service management, mobile ad hoc networks, multi-objective optimization, service composition, task assignment.

Trust-Based Service Management for Service-Oriented Mobile Ad Hoc Networks and Its Application to Service Composition and Task Assignment with Multi-Objective Optimization Goals

Yating Wang

ABSTRACT

With the proliferation of fairly powerful mobile devices and ubiquitous wireless technology, traditional mobile ad hoc networks (MANETs) now migrate into a new era of service-oriented MANETs wherein a node can provide and receive service from other nodes it encounters and interacts with. This dissertation research concerns trust management and its applications for service-oriented MANETs to answer the challenges of MANET environments, including no centralized authority, dynamically changing topology, limited bandwidth and battery power, limited observations, unreliable communication, and the presence of malicious nodes who act to break the system functionality as well as selfish nodes who act to maximize their own gain.

We propose a context-aware trust management model called CATrust for service-oriented ad hoc networks. The novelty of our design lies in the use of logit regression to dynamically estimate trustworthiness of a service provider based on its service behavior patterns in a context environment, treating channel conditions, node status, service payoff, and social disposition as "context" information. We develop a recommendation filtering mechanism to effectively screen out false recommendations even in extremely hostile environments in which the majority recommenders are malicious. We demonstrate desirable convergence, accuracy, and resiliency properties of CATrust. We also demonstrate that CATrust outperforms contemporary peer-to-peer and Internet of Things trust models in terms of service trust prediction accuracy against collusion recommendation attacks.

We validate the design of trust-based service management based on CATrust with a node-to-service composition and binding MANET application and a node-to-task assignment MANET application with multi-objective optimization (MOO)

requirements. For either application, we propose a trust-based algorithm to effectively filter out malicious nodes exhibiting various attack behaviors by penalizing them with trust loss, which ultimately leads to high user satisfaction. Our trust-based algorithm is efficient with polynomial runtime complexity while achieving a close-to-optimal solution. We demonstrate that our trust-based algorithm built on CATrust outperforms a non-trust-based counterpart using blacklisting techniques and trust-based counterparts built on contemporary peer-to-peer trust protocols. We also develop a dynamic table-lookup method to apply the best trust model parameter settings upon detection of rapid MANET environment changes to maximize MOO performance.

Trust-Based Service Management for Service-Oriented Mobile Ad Hoc Networks and Its Application to Service Composition and Task Assignment with Multi-Objective Optimization Goals

Yating Wang

GENERAL AUDIENCE ABSTRACT

With the proliferation of fairly powerful mobile devices and ubiquitous wireless technology, traditional mobile ad hoc networks (MANETs) now migrate into a new era of service-oriented MANETs wherein a node can act as a service provider and as a service requester with other nodes it encounters and interacts with. This dissertation research concerns trust management and its applications for service-oriented MANETs.

The dissertation research has the following unique contributions: (a) we develop a context-aware trust protocol called CATrust for effective and efficient trust management in service-oriented MANETs, treating channel conditions, node status, service payoff, and social disposition as "context" information. We demonstrate that our trust protocol is highly resilient toward collusion recommendation attacks and can significantly outperform contemporary trust protocols in terms of accuracy and resiliency against recommendation attacks; (b) we apply CATrust to a service composition and binding MANET application with multi-objective optimization (MOO) goals, treating MOO as service quality and considering the effect of context on MOO treated as service quality; we demonstrate that CATrust outperforms conventional single-trust and multi-trust protocols; and (c) we apply CATrust to a task assignment MANET application with MOO goals and demonstrate that CATrust outperforms a conventional trust protocol because it considers the association between context and MOO treated as service quality for this task assignment MANET application. We develop a table-lookup method to apply the best trust model parameter settings

upon detection of dynamically changing environmental conditions to maximize CATrust performance.

Acknowledgements

This dissertation would not have even been possible without the support and help of those people who have been with me during the years. With them, I obtained such a meaningful journey for my life. I would love to express my appreciation here.

First of all, I sincerely appreciate my advisor, Dr. Ing-Ray Chen. Because of your persistent patience and support, it helps me pass every milestone for finishing PhD study from a student without engineering background. Thank you for always giving me opportunities to improve myself.

I would also like to thank Dr. Jin-Hee Cho and Dr. Ananthram Swami. Although we did not see each other very often, you share your knowledge with me and always give me valuable suggestions whenever there is a chance.

Last but not least, to my greatest family, I am so grateful for your endless love, support and encouragement.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Trust-Based Service Management in Service-Oriented Mobile Ad Hoc Networks

With the proliferation of fairly powerful mobile devices and ubiquitous wireless technology, traditional mobile ad hoc networks (MANETs) now migrate into a new era of service-oriented MANETs wherein a mobile device can provide and receive service from other mobile devices it encounters and interacts with.

A service-oriented mobile ad hoc network (MANET) is populated with service providers (SPs) and service requesters (SRs). A realization of service-oriented MANETs is a peer-to-peer service system with SPs providing web services and SRs requesting services, each requiring dynamic service composition and binding [71]. Unlike a traditional web service system in which nodes are connected to the Internet, nodes in service-oriented MANETs are mobile and an SR will need to request services from available SPs it encounters and with which it interacts dynamically. One can view a service-oriented MANET as an instance of Internet of Things (IoT) systems [16] with a wide range of mobile applications including smart city, smart tourism, smart car, smart environmental monitoring, and healthcare [21]. It is particularly suitable to military MANET applications where nodes are mobile with multi-hop communication.

The primary concern of a service-oriented MANET is the existence of malicious nodes (acting on behalf of malicious owners) who may collude to maximize their own gain and even monopoly service. Our approach is to use trust [16] [17] [29] [30] [47] [55] [69] [70] for trust-based service management of service-oriented MANET applications. An SR can assess the trust levels of the SPs with which it interacts, and propagate its observations of service

performance as recommendations to other nodes, so that a well-behaving SP is more likely to be selected to provide services and low trustworthy SPs can be detected and isolated from the network. This in turn encourages node cooperation because SPs deemed trustworthy are allowed to access network resources.

1.2 Research Challenges and Our Solutions

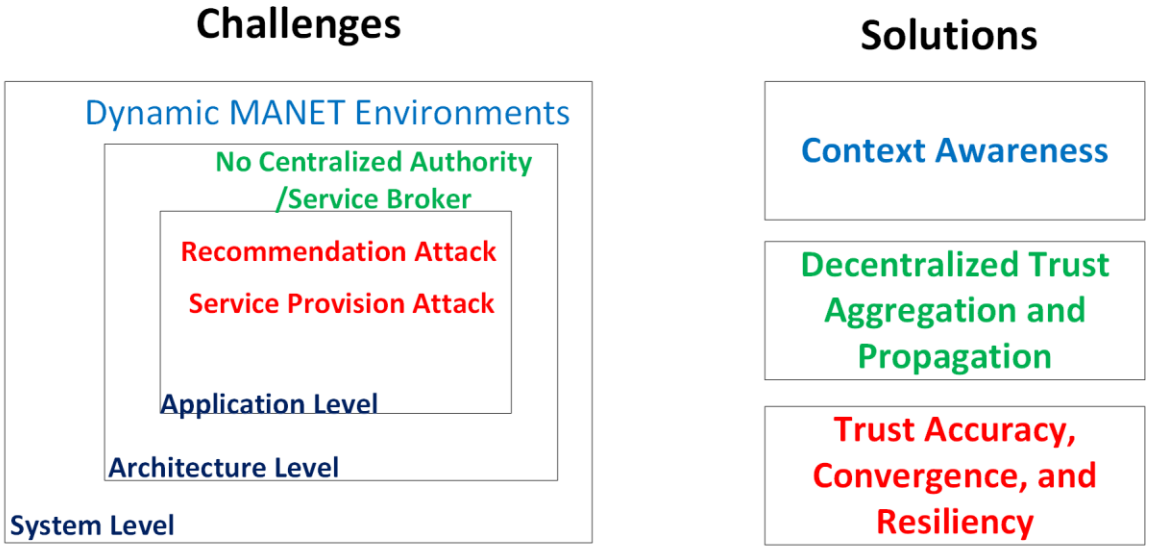


Figure 1.1: Research Challenges and Our Solutions.

Figure 1.1 depicts the main challenges of service-oriented MANET applications at the system, architecture, and application levels. At the system level, the MANET environment is highly dynamic due to node mobility and intermittent connection. Our design uses context-aware trust computation to take into consideration of the association between service quality and environment context. At the architecture level, there is no centralized authority coordinating SRs to reliable SPs. Our design uses distributed trust aggregation and propagation so each SR assesses service trust of SPs in a decentralized manner. At the application level, there are malicious nodes that can launch service and recommendation attacks. To tolerate these attacks, our design is to incorporate robustness into our trust computation model to achieve trust accuracy, convergence, and resiliency.

Below we elaborate more on these challenges and our solutions.

A major challenge in trust-based service management in service-oriented MANETs is to reliably estimate the trust levels of SPs in a fully distributed

manner. A node can simply observe direct evidence for direct trust assessment and propagate its observations to other nodes as recommendations for indirect trust assessment. However, a malicious node may violate this protocol. Sun et al. [122] described several attacks to trust management in distributed networks, including bad-mouthing, random, conflicting behavior, Sybil, and newcomer attacks. Among these, Sybil and newcomer attacks depend on intrusion detection mechanisms (running in parallel with trust mechanisms) for detecting identity attacks. Bad-mouthing, random, and conflicting behavior attacks require an effective trust management system for discerning misbehaving nodes from behaved nodes. Coping with such attacks is particularly challenging in service-oriented MANET environments since an SR may have little or limited evidence about an SP or a recommender because of logical or physical separation created by node mobility. Our solution is to incorporate robustness into our trust computation model to achieve trust accuracy, convergence, and resiliency despite malicious attacks.

Another major challenge is that the quality of service (QoS) received by an SR from an SP depends on the operational and environmental conditions of the SP. This blurs the SR's view on the service quality of the SP. For example, a service-oriented MANET may adopt IEEE 802.11 [146] in distributed coordination function (DCF) mode, so it is likely that the waiting time to access channel may cause a significant delay when many nodes are in the environment competing for the use of the channel. In the literature [4] [88] [151], this issue is tackled by breaking QoS into multiple QoS trust components, each being assessed separately, and then integrated together into an overall trust value. However, the way to select multiple trust components to form into a single trust value is often devised in an ad hoc manner and remains an open issue. Our solution is context-aware trust computation considering the association between service quality and environment context.

Real world service-oriented MANET applications often have multiple objectives [45] [136] and most of the time conflicting goals. For example, a user might want to have a high quality of information (QoI) via lossless information compression, but it also wants a low delay. In this case, *multi-objective optimization* (MOO) is considered a design goal. This process is especially complicated in service-oriented MANETs because of the space-time complexity of mobile devices (space is related to mobility and time is related to dynamic status change), and no existence of a trusted third party for centralized control. This issue is further compounded by the fact that nodes may exhibit malicious behavior and the information received is often erroneous, uncertain and

incomplete in MANET environments [47]. Our solution is to identify essential trust dimensions for a specific service-oriented MANET application and discover the best way to design the trust-based service management algorithm that can maximize the application performance in terms of MOO compared with non-trust-based baseline schemes.

1.3 Research Contributions

Our main contributions lie in answering the above challenges for designing an effective and efficient trust management protocol for service-oriented MANETs and validating trust-based service management by real-world service-oriented MANET applications with MOO requirements. More specifically, the dissertation research has the following unique contributions:

1. We take a context-aware approach for trust management in service-oriented MANETs, treating channel conditions, node status, service payoff, and social disposition as "context" information. We view trust as the probability that an SP will provide a satisfactory service in a particular context environment, as expected by an SR. The novelty of our work lies in the use of a robust inference model based on logit regression to robustly yet accurately predict how an SP's service quality will be in response to context changes. This allows us to reason about a node's service behavior pattern, given the operational and environmental context information as input. We name our context-aware trust management protocol "CATrust" (see Chapter 4) with the following contributions:
    a. To the best of our knowledge, we are the first to propose a context-aware trust management for service-oriented MANETs. Similar to existing P2P or IoT trust protocols, we also predict an SP's service trust based on the SP's behavior. The difference lies in our ability to associate an SP's service behavior with context information. The end result is that there is one service trust value for each context environment, instead of one service trust value for all context environments (as in existing protocols). This context-aware design greatly improves service trust prediction accuracy.
    b. We propose a novel recommendation filtering mechanism to effectively screen out false recommendations even when most of the recommendations are malicious.

c. We demonstrate that CATrust is highly resilient toward collusion recommendation attacks. CATrust significantly outperforms contemporary P2P and IoT trust protocols in terms of accuracy and resiliency against recommendation attacks.

2. We develop a trust-based service composition and binding algorithm with MOO in service-oriented MANETs (see Chapter 5) with the following contributions:

   a. Our work is the first to propose a dynamic trust-based service composition and binding algorithm for MOO in service-oriented MANETs. Our proposed scheme has only linear runtime complexity for solution search but approaches the ideal performance obtainable by the Integer Linear Programming (ILP) solution which has exponential runtime complexity, and is thus applicable only to small-sized problems.

   b. We are the first to conduct a comparative performance analysis of non-trust vs. CATrust, single-trust, and multi-trust protocols for peer-to-peer trust evaluation in service-oriented MANETs. Trust-based service composition and binding has been studied in the web services domain [15] [50] [135], but only a single trust score on service quality was considered, although the single trust score may derive from multiple service quality metrics such as response time, throughput, availability, etc. This largely ignores the multidimensional concept of trust. Identifying proper trust components and forming the overall trust out of multiple trust components is critical to maximize application performance. We consider two key trust dimensions in service request execution, namely, competence and integrity, as the building blocks of a composite trust metric.

   c. We use trust to effectively prevent malicious nodes from disrupting the operation of service-oriented MANETs. We conduct a detailed performance analysis and demonstrate that our trust-based algorithm can effectively filter out malicious nodes, which ultimately leads to high user satisfaction.

   d. We identify a key component for evaluating service quality of service-oriented applications with MOO goals, that is, service quality should not link to conventional performance metrics such as response time, delay, or energy consumption, but should link to application objectives because ultimately MOO is the goal of the application. We are the first

to apply a context-aware trust protocol (CATrust) to a service-oriented MANET application with a MOO goal, treating MOO as service quality and considering the effect of context on MOO treated as service quality. We demonstrate that CATrust outperforms conventional single-trust and multi-trust protocols.

3. We develop a trust-based task assignment algorithm with MOO in service-oriented MANETs (see Chapter 6) with the following contributions:
    a. To the best of our knowledge, this work is the first to solve a multi-objective optimization (MOO) problem dealing with multiple, concurrent and dynamic task assignments with conflicting goals using trust in service-oriented MANETs. Our trust-based heuristic algorithm has a polynomial runtime complexity, thus allowing dynamic node-to-task assignment to be performed at runtime. Our heuristic design can only produce a suboptimal solution. However, our heuristic design is able to achieve a solution quality approaching that of the ideal solution which has perfect knowledge of node status.
    b. This work proposes and analyzes a new design concept of trust-based MOO based on assessed trust levels to screen task team members for dynamic node-to-task assignment. Fourth, we conduct a comparative analysis of our proposed trust-based heuristic member selection algorithm against the ideal solution with perfect knowledge of node status, demonstrating that our trust-based solution achieves solution efficiency without compromising solution optimality. In particular, we demonstrate that CATrust outperforms a conventional trust protocol because it considers the association between context and MOO treated as service quality for this task assignment MANET application.
    c. We develop a table-lookup method to apply the best trust model parameter settings upon detection of dynamically changing environmental conditions to maximize protocol performance.

1.4 Thesis Organization

The remaining part of the dissertation is organized as follows. Chapter 2 surveys related work in trust management for MANETs (including MANET variants such as coalition networks and tactical networks). It also surveys the current-state-of-the-art in solving trust-based service composition MOO problems and trust-based task assignment MOO problems. Chapter 3 presents

the system model including the network model and the attack model. In Chapter 4, we propose and analyze CATrust, a logit regression-based context-aware trust model, for service-oriented MANETs. In Chapter 5, we validate the design of trust-based service management based on CATrust with a node-to-service composition and binding MANET application with MOO. In Chapter 6, we validate trust-based service management based on CATrust with a node-to-task assignment MANET application with MOO. Chapter 7 summarizes research accomplishments and future research directions extended from the dissertation research.

Chapter 2

Related Work

2.1 Concept of Trust

Trust has been extensively studied in economics, politics, philosophy, sociology etc. as one of the significant factors determining human's decision, while it is highly advocated to be leveraged into network security to cope with insider attacks unsettled under traditional cryptography methods [5] [12] [19] [22] [27] [51] [72] [74] [95] [104] [106] [115] [117] [150]. The logic behind is that trust is crucial for the environment with uncertainty, risk or interdependence [91].

Nevertheless, the definition of trust varies in the scientific community and many existing works define the essential factors for trust differently. For instance, McKnight and Chervany [90] deemed five elements - potential negative consequences, dependence, feelings of security, a situation-specific context, and lack of reliance on control – for trusting intentions; and four dimensions – *benevolence*, *honesty*, *competence*, and *predictability* – for trusting beliefs. Whetter [145] embodied *What*, *How*, *When* and *Why* for constructing a trust model. Jøsang [70] captures the meaning of trust in two parts – *reliability trust* and *decision trust*. Govindan and Mohapatra [59] considered that trust reveals the evaluator's (trustor's) belief/confidence/expectation on the service provider's (trustee's) future endeavor in terms of honesty, integrity, ability, availability and quality of service, which represents the trustee's trustworthiness. In the context of service-oriented MANETs, trust should also reflect a service provider's capability to provide reliable service under specific context environments.

In this dissertation, we follow Govindan and Mohapatra's trust definition for MANETs. Particularly, we define trust as the probability that a service requester will receive a satisfactory service in terms of QoS from a specific service provider.

2.2 Trust Management Protocol Design for Service-Oriented MANETs

2.2.1 Trust Computation Models

We survey existing works in trust computation models for service-oriented MANETs including P2P networks and IoT networks, and discuss how they deal with trust-related attacks. We compare and contrast our context-aware trust management protocol (CATrust) proposed in this dissertation research with existing works.

Unlike an e-commerce system with a centralized authority for trust management, trust management in MANETs is decentralized [2] [20] [79]. Trust management is challenging in MANETs [133] because there is no infrastructure connecting nodes together so that nodes must individually build up trust relations with other through interaction experiences upon encountering or recommendations. Because of node mobility, a MANET is characterized by rapidly changing topologies which hinder communication reliability and introduce high uncertainty in evidence gathering and trust evaluation. Since trust management is decentralized, each node can store a trust evidence table towards other acquaintances [26] [95], or can aggregate evidence itself with feedbacks from other nodes [103] [105].

Many existing trust models for predicting trust are based on Bayesian inference as [69]. In the context of MANETs, Yu et al. [154] applied Bayesian inference to measure the reputation of a node assuming that a node's behavior in each observation period is identically and independently distributed, and follows the binomial distribution. Therefore, a node's reputation is determined by the numbers of positive and negative samples observed. Sun et al. [123] utilized Bayesian inference for trust bootstrapping, using accumulated evidence of a node's performance. Buchegger and Le Boudec [25] adopted a modified Bayesian approach by assigning more weights to current evidence and reducing weights on past evidence. Similar approaches were adopted in several works for modeling dynamic trust in MANETs by considering trust decay over time [37] [48] [49]. In the context of IoT networks, Chen et al. [31] also considered a Bayesian framework as the underlying model for evaluating direct trust from user satisfaction experiences. A shortcoming of the above models based on Bayesian inference is that the trust value is not associated with context since it is just based on user satisfaction experiences.

Belief theory or subjective logic trust models [67] [70] introduce uncertainty into trust calculation. Balakrishnan et al. [13] developed a subjective logic based model for a node to evaluate its opinion towards another node using evidence-to-opinion mapping operators. Twigg [129] developed a subjective logic based trust model for selecting trustworthy nodes for secure routing in P2P networks. Similar to existing trust models based on Bayesian inference, these works only considered user satisfaction experience for trust computation.

Fuzzy logic based trust models are also well studied in the literature [141] [148]. Instead of using a binary set, a membership function is defined indicating the degree to which a node is considered trustworthy. Wang and Huang [141] computed the fuzzy trust values of candidate paths based on node reputation, bandwidth, and hop count for route selection. Xia et al. [148] applied fuzzy inference rules for trust prediction, considering past and current service experiences for predicting the service capability of a transmitter node. One drawback of fuzzy logic-based trust prediction is that it requires domain experts to do parameter tuning and set the fuzzy rules incorporating the knowledge of the causal relationship between the input and output parameters.

Relative to the works cited above based on Bayesian inference, belief theory, and fuzzy logic, we take an entirely different approach. We develop CATrust, a regression-based trust model utilizing logit regression, to estimate the trustworthiness of an SP in response to context environment changes. There is little work in the literature on applying regression for trust computation. To date, it has been used only by [84] [131] for finding the best weights to assign to observations or trust components. Specifically, Li et al. [84] proposed an auto-regression-based technique to learn the weights from historical observations to predict future outcomes. Venkataraman et al. [131] developed a regression-based trust model to learn the optimal weights of multiple trust metrics, where each trust metric is assessed separately using Bayesian inference. Unlike [84] [131], we do not use regression to learn weights to be applied to observations or trust properties. Instead, we apply logit regression analysis to learn the service behavior patterns of an SP in response to context environment changes and consequently predict the SP's dynamic trust in terms of its service quality trust. For this reason, we do not consider [84] [131] as baseline cases for performance comparison. We consider Beta Reputation [69] and Adaptive Trust [31] (both measuring service quality trust) as baseline cases for performance comparison in this paper.

Beta Reputation is the most popular trust protocol for P2P systems to-date. It applies the concept of belief discounting for recommendation filtering. Adaptive Trust [31] is a very recent IoT/P2P trust protocol proven to outperform other contemporary IoT/P2P protocols. It applies the concept of collaborative filtering for recommendation filtering and the concept of adaptive filtering for dynamically adjusting the weights of direct trust obtained through self-observations and indirect trust obtained through recommendations to maximize protocol performance.

A significant amount of work has been done in the area of trust-based defenses against attacks in P2P, IoT and mobile ad hoc networks [31] [37] [48] [49] [72] [120] [140] [150]. In particular, Chen et al. [37] proposed the concept of trust bias minimization by dynamically adjusting the weights associated with direct trust (derived from direct

evidence such as local observations) and indirect trust (derived from indirect evidence such as recommendations) so as to minimize trust bias. Cho et al. [48] [49] proposed the use of trust thresholds to filter out untrusted recommendations. EigenTrust [72], PeerTrust [150], and ServiceTrust [120] considered various methods to aggregate recommender feedbacks weighted by the recommender's trustworthiness based on factors that affect a recommender's trustworthiness, including transaction context, community context, and credibility in terms of the trust and personalized similarity between the trustor and the recommender, etc. to filter out distrusted feedbacks.

A common challenge with the above approaches is that dynamically tuning trust parameters may perform poorly when malicious recommenders form a majority, especially if a node does not have enough service experiences with other nodes and must rely on recommendations for decision making. CATrust leverages a robust statistical kernel to tolerate false recommendations to effectively achieve resiliency against recommendation attacks. Furthermore, CATrust filters out false recommendations based on a novel threshold-based filtering mechanism such that if the difference between the predicted service quality trust and the recommended service quality trust under the same context environment is above a threshold, then the recommendation is filtered. We demonstrate that CATrust significantly outperforms contemporary P2P/IoT trust models including Beta Reputation and Adaptive Trust, especially when observations for recommenders are limited.

2.2.2 Context-Aware Trust Models

Context-aware trust models have existed in e-commerce applications [86] [159] and crowdsourcing applications [152]. Liu and Datta [86] applied a Hidden Markov Model (HMM) to model an agent behavior with the associated interaction contextual information, where the context information includes the trustee/service provider's features from its profile, service/product features, and social relationship between the trustor and trustee. Zhang et al. [159] proposed a trust vector approach to evaluate transaction context-aware trust in e-commence and e-service environments. It uses transaction attributes and service quality attributes to model a seller's trust. In the trust computation model, it considers similarity between transactions and uses a weighted-sum of ratings from similar transactions to predict service ratings. Service trustworthiness is represented by a vector of ratings and the corresponding context information under which the transaction is performed. On the other hand, Ye et al. [152] designed a context-ware trust model called CrowdTrust for worker selection in crowdsourcing environment with multiple objectives. It assumes that a worker has different trust levels in different context environments, where the context information

includes the task type and task reward amount. Their trust computation model uses a *sigmoid* function with a vector of weights associated with context variables, combining with a time decaying function, to estimate a worker's trust given the task type and reward as input. Wang et al. [142] proposed SocialTrust, a context-aware trust model which considers the effect of recommendations on service providers in social networks. The context information takes into account both SP and SR's characteristics and their social relations. The inference process is based on conditional probability calculation.

Ries [108] integrated context-dependent parameters into a Bayesian trust model. The intuition is that derived trust is influenced by context-dependent aspects such as the initial trust value, weight of the dispositional trust, aging factor, and maximal number of evidence. The design is to take a trustor's preference into consideration when rating a trustee. Hence, context is used to aggregate opinions/recommendations from multiple recommenders. This design might mitigate the effect of recommendation attacks. However, it does not consider trust accuracy.

Uddin et al. [130] designed a context-aware trust model called CAT that considers services context where the trust model involves a number of trust properties (e.g., context and risk awareness). Given service context as input, a context similarity parameter is computed to help user with decision making. Saied et al. [110] designed a context-aware and multi-service trust management system for IoT systems. An IoT node may switch from one context to another due to changes of available resources. When selecting service provider candidates, it considers service providers that match the service context of the service request. If no exact match is found, it selects service providers with high context similarity. Shankaran et al. [113] described a context-aware approach for assessing trustworthiness of participating nodes in a dynamic MANET environment. Trust is assessed from a node's functionality in the network. Specifically, trust is computed based on a node's capability, service type, trust table, and traffic class. Eventually, trust is computed by a weighted-sum of all trust dimensions. Li et al. [83] proposed a context-aware security and trust framework called CAST for MANETs. It considers multiple environmental context parameters, including communication channel status, battery status, and weather conditions, to assess the behavior of each mobile node. In the framework, a policy management module determines whether misbehavior is caused by environmental factors or deliberately conducted by adversaries.

In summary, existing context-aware trust models cited above adopted the following approaches: (a) they either computed context similarity for obtaining a trust value; (b) they computed a trust value for each context property and then integrated all trust values by weighted-sum into one trust value; (c) they simply used a policy-based rule

set based on context information to determine the trust value. Unlike existing context-aware trust models, CATrust uses direct observations and trust recommendations from other nodes, and applies an inference process to discover an SP's service behavior patterns in a context environment. The association between service quality and environmental context variables is learned by CATrust while it collects direct observations and indirect recommendations under different context environments. This association is used to predict an SP's delivered service quality in the future given a context environment as input.

2.3 Trust-based Service Composition with MOO in Service-Oriented MANETs

Service composition has been widely studied in Internet-based web services [28] [71] [73] [107] [111] [118] [121] [126] [158]. Service composition and binding comes in two forms: (a) goal-oriented composition where a goal and a set of available services are given and the system completes the goal by planning and service binding; and (b) workflow-based composition where the workflow with constraints is given as input [111]. We take the latter approach, with service composition being formulated as a workflow problem (based on the user's location and the availability of SPs an SR encounters), and service binding being formulated as a node-to-service assignment problem.

2.3.1 Service Composition in MANETs

Service composition in MANETs is still in its infancy. Existing work only focused on mechanisms for enabling service composition without considering MOO. Sheu et al. [114] designed a tactical pervasive service collaboration platform to enable service composition considering dynamic task arrival, data synchronization, and task failure. Johnsen et al. [64] [65] suggested a semantic service discovery solution with a rich and expressive service description provided to facilitate service composition. Wright et al. [147] proposed the use of UML 2.0 Activity Diagrams as a workflow specification language for describing service construction and composition. Suri [124] studied the deployment issues of service-oriented architectures and developed a middleware to facilitate dynamic service composition to cope with those issues. Compared with the above cited work, our work is the first to propose a dynamic service composition and binding algorithm using multi-trust protocols for MOO in service-oriented MANET.

Singh [116] indicated that trust is an important factor in service-oriented computing where user experience is the main factor for trust establishment. Bansal et al. [15] and Dai et al. [50] proposed trust-based web service composition, but only a single dimension of trust was considered. Relative to the above work, we consider multi-trust, recognizing multi-dimensional trust assessment is critical to decision making. We demonstrate that our multi-trust-based algorithm outperforms single-trust-based Beta Reputation [55] [69].

Wahab et al. [135] provided an excellent review of trust and reputation models for Web services. Mehdi et al. [92] [93] [94] considered multiple QoS metrics (e.g., response time, throughput, availability, etc.) for assessing the service quality of a web service. A trust score is derived from combining multiple QoS metrics to assess the trustworthiness of a web service. This single trust score considered in [92] [93] [94] in effect corresponds to the competence trust score considered in our work. In addition, we also consider the integrity trust score (the other metric of our multi-trust design) for measuring the degree to which a node complies with the prescribed service protocol. Khosravifar et al. [75] considered multiple reputation factors (user satisfaction and popularity) and analyzed their relationships. We do not consider popularity as a trust metric. Instead, we consider integrity as a trust metric to cope with malicious attacks. Hang et al. [63] modeled a composite service as a statistical mixture, and showed that their approach can dynamically punish or reward the constituents of composite services while making only partial observations. Hang et al. [62] later developed a probabilistic trust model considering not only the trust level, but also the amount of evidence supporting the trust level. They showed that their trust model yields higher prediction accuracy than traditional approaches suffering from situations in which witnesses are unreachable or are reachable only by untrustworthy referrals.

In our work, we consider confidence in the context of trust formation. That is, integrity trust is used as confidence to assess the validity of competence trust based on the rationale that competence trust ultimately ensures service success. This is discussed in more detail in Chapter 5.

2.3.2 Multi-Objective Optimization in Service Composition

Multi-objective optimization (MOO) has also been extensively studied in Internet-based web services [119]. Wagner et al. [134] proposed a planning assistant that achieves MOO for three objectives including price, response time, and reliability by approximating Pareto-optimal solutions. Yu and Lin [155] studied MOO with end-to-end QoS objectives, including response time, service cost, availability and reliability.

Alrifai and Risse [10] considered end-to-end QoS objectives during the runtime service binding process. Both [10] and [155] used a multi-choice multidimensional knapsack problem to formulate the MOO problem.

Weighted-sum is a common approach used in service composition with MOO. Yu et al. [156] addressed a service selection problem by aggregating multiple QoS objectives into a weighted utility function to be maximized subject to QoS resource needs with each weight representing the importance of each QoS attribute. Zeng et al. [158] considered five objectives, namely, price, duration, reliability, availability and reputation and formulated the MOO problem as a single objective problem using weighted sum. Similar to [156] [158], we adopt weighted sum to formulate our MOO problem for its simplicity and proven effectiveness.

The above cited work had a common drawback. Namely, their solutions have exponential time complexity because the MOO problem to be solved is NP-hard [58]. The contribution of our work remedies this problem by devising trust-based heuristic solutions that incur only linear runtime complexity, and verifying that the performance of our trust-based solution approaches the ideal performance obtainable by ILP solution.

2.4 Trust-Based Task Assignment with MOO in Service-Oriented MANETs

Existing work on task assignment MOO can be categorized into two classes, depending on whether the work deals with system objectives for global welfare and/or individual objectives for individual welfare. Class 1 represents the case in which there are multiple system objectives for global welfare, but there are no individual objectives. Class 2 represents the case in which there are multiple system objectives for global welfare but there are also individual objectives for individual welfare which may induce or hinder global welfare. Our work falls under Class 2 as we consider the presence of malicious nodes performing malicious attacks and colluding to monopoly service as the individual objectives for individual welfare.

Class 1 concerns solving a task assignment MOO problem to maximize global welfare but individual nodes do not have individual objectives. Balicki [14] studied a task assignment problem in a distributed environment based on a multi-objective quantum-based algorithm. The objectives are to maximize system reliability while minimizing workload and communication cost, all are global welfare. Chen et al. [42] solved a task assignment problem in a multi-robot environment consisting of

heterogeneous mobile robots, given resource constraints associated with tasks. They proposed a heuristic leader-follower structure that identifies optimal solutions of the task allocation problems. Guo et al. [61] examined a task assignment problem using a particle swarm optimization technique that minimizes task execution time and cost for data transfer between processors in cloud computing environments. Shen et al. [114] develop a trust-based solution for task assignment in grid management with multiple system objectives including security, reliability, load balance, and throughput. Solutions fall under Class 1 assume no malicious entity in the system, which is not a valid assumption in a service-oriented MANET environment which very likely will be populated with malicious nodes acting for own interest and colluding for individual welfare. Xie and Qin [149] proposed an energy-efficient task assignment protocol based on the tradeoff between energy and delay to execute a task for collaborative networked embedded systems to minimize the length of schedules of task allocation and energy consumption.

Relative to Class 1 solutions, we develop a trust-based service management protocol specifically for autonomous service-oriented MANET applications, and demonstrate the resiliency and convergence properties of our trust protocol design for service-oriented MANET in the presence of malicious nodes performing bad-mouthing, ballot-stuffing, opportunistic service, and self-promotion attacks.

Class 2 concerns solving a task assignment MOO problem to maximize global welfare but nodes may have separate individual objectives for individual welfare. Anagnostopoulos et al. [11] explored a solution for a task assignment problem by matching a set of skilled people to each task. Their solution aims to minimize the communication overhead while balancing workloads by solving a single objective problem that considers multiple objectives. Edalat et al. [52] proposed an auction-based task assignment solution in wireless sensor networks with two global objectives: maximizing the overall network lifetime while satisfying application deadlines. An individual entity seeks to maximize its payoff by bidding on a task with low workload so as to consume less energy but have a high chance of being assigned to the task. Szabo and Kroeger [125] examined a task allocation problem in cloud computing using evolutionary genetic algorithms. This work has the system goals to minimize workflows, delay introduced by the task completion, and communication cost while each individual user wants to minimize cost and service delay. Tolmidis and Petrou [128] proposed an auction theoretic approach to solve a task allocation problem in multi-robot systems where each robot is able to perform several functions. An individual robot has the goals to minimize energy consumption and delay in task completion while maximizing the degree of relevancy and priority level to an assigned

task. Similarly, the system aims to maximize the number of completed tasks and minimize delay introduced due to task assignment and completion. Wang et al. [143] proposed a trust-based task assignment technique for mobile ad hoc grid computing environments for maximizing mission completion ratio based on required levels of security and reliability in task assignment and minimizing delay to mission completion.

The main drawback of the existing work cited above is that the worst case runtime complexity is exponential because the MOO problem to be solved is NP-complete [57], making it unsuitable to be deployed at runtime. Our contribution is to remedy this problem by devising trust-based heuristic solutions that incur only polynomial runtime complexity, and verifying that the performance of our trust-based solution approaches the ideal MOO performance with perfect knowledge of node status.

Chapter 3

System Model

In this chapter, we discuss the system model including the network model discussed in Section 3.1 and the threat model discussed in Section 3.2.

3.1 Network Model

We consider a service-oriented MANET in which a node has two roles: (1) a service provider (SP) to provide service; and (2) a service requestor (SR) to request a service. The service requested may be a composite service which requires dynamic service composition and binding. We consider location-based service requests. That is, an SR requests a service, and SPs in the same location (within radio range) respond to the request. The SR selects the most trustworthy SPs for providing service. The SR can issue a sequence of service requests as it moves from one location to another. An example is a user in a smart city who first issues a service request "take me to a nice Thai restaurant nearby with drunken noodle on its menu" with a service quality specified in terms of QoI, QoS, and cost for the overall service request (e.g., the cost and duration of travel), as well as for individual abstract services (e.g., cost of drunken noodle). Once she finishes her meal, she issues another service request "take me to a nice night club in town" again with a minimum service quality specified in terms of, for example, QoI, QoS, and cost. Each of these service requests involves a service composition phase to compose a service plan out of the transportation services (e.g., taxi, bus, subway, etc.) and Thai food/night club services available to the user, followed by a service binding phase to select the best SPs out of all SPs available to the user at the time the service request is issued.

Nodes may be heterogeneous with vastly different functionalities and natures. For example, the entities may be sensors, robots, unmanned vehicles or other devices, dismounted soldiers or first response personnel carrying sensors or handheld devices, and manned vehicles with various types of equipment. Nodes can directly

communicate with each other if they are within wireless radio range or through multi-hop routing.

Mobility introduces dynamic topology changes and affects the reliability of packet routing over multiple hops from a source to a destination. In particular, it affects the success probability of recommendation packet delivery which in turn affects trust protocol performance. There are two forms of trust recommendation propagation. One form is broadcasting-based by which a recommendation is propagated to all nodes through multi-hop routing whenever a trustor has new observations. The recommendation packet is lost when there is no route to reach a destination node because of topology changes, when there is a channel error, or when any intermediate node maliciously performs packet dropping attacks. Another form is encounter-based by which a recommendation is provided when two nodes encounter each other. We test the resiliency of our trust-based algorithm design and the sensitivity of our analysis results with respect to two mobility models, i.e., Random Waypoint mobility (RWM) model [66] and Small World in Motion (SWIM) [76] [77]. We select SWIM because it captures key properties of human mobility in social network settings.

3.2 Attack Model

Just like Internet-based web services, in a service-oriented MANET there are malicious SPs acting for their own gain or individual welfare. The common goal of malicious nodes is to increase their chance of being selected while answering a service request. Malicious nodes can collude to achieve this common goal. Here, a malicious node's motivation is application-specific. In profit driven service-oriented MANETs (Chapter 5), we define malicious nodes as self-interest (or selfish) entities that will take advantage of the system to maximize their own benefits. In mission driven service-oriented MANETs (Chapter 6), we define malicious nodes as adversary with intent to disrupt the mission by failing as many tasks as possible. We assume that malicious nodes know each other and will collude to monopoly service for the worse case analysis. For convenience, we will use the word "bad" interchangeably with "malicious," and the word "good" interchangeably with "non-malicious."

Let $M(x)$ be a predicate evaluated to true if x is malicious, and $G(x, y)$ be a predicate evaluated to true if both x and y are malicious. Also let $RA(x, p)$ be a predicate evaluated to true if x attacks with probability p.

We assume that a malicious node exhibits the following behaviors:

1. *Self-promotion attacks*: A malicious node can promote its importance by reporting false service quality information so as to increase its chance to be selected as the good SP, but then provide inferior service or opportunistic service.
2. *Bad-mouthing attacks*: A malicious recommender can collude with other malicious nodes to ruin the trust of a good SP by providing bad recommendations against the good SP so as to decrease the chance of the good SP being selected to provide services. A bad-mouthing attack is performed by x (the malicious recommender) to z (the SR) about y (the good SP) when $\neg G(x, y) \wedge M(x)$ is evaluated true.
3. *Ballot-stuffing attacks*: A malicious recommender can collude with other malicious nodes to boost the trust of a bad SP by providing good recommendations for the bad SP so as to increase the chance of the bad SP being selected to provide services. A ballot-stuffing attack is performed by x (the malicious recommender) to z (the SR) about y (the malicious SP) when $G(x, y) \wedge M(x)$ is evaluated true.
4. *Conflicting-behavior attacks* (or *opportunistic service attacks*): A malicious SP can selectively provide satisfactory service within its service capability for some SRs while providing unsatisfactory service for others. With conflicting behavior attacks, x (the malicious SP) typically delivers unsatisfactory service to z (the SR) when $\neg G(x, z) \wedge M(x)$ is evaluated true, but delivers satisfactory service within its service capability to z (the SR) when $G(x, z) \wedge M(x)$ is evaluated true. When providing unsatisfactory service, a malicious node may be penalized with trust loss. A smart malicious node thus can provide "opportunistic" service to meet the minimum quality service requirement and user satisfaction expectation to improve the chance of the service request being completed successfully for it to gain good trust.
5. *Random attacks*: While performing conflicting behavior attacks, a malicious SP can perform *random attacks*, i.e., providing unsatisfactory service to non-malicious SRs only randomly, so as to avoid being labeled as a low service trust node and risk itself not being selected by non-malicious SRs in the future. With random attacks, x (the malicious SP) will deliver unsatisfactory service to z (the SR) when $\neg G(x, z) \wedge M(x) \wedge RA(x, p)$ is evaluated to true.
6. *Packet dropping attacks*: A malicious may drop packets passing through it during packet routing if the source node is a good node so as to launch a bad-mouthing attack against the source node.

Here, we note that our trust protocol design is to defend against inside attackers (who are legitimate members of the service-oriented MANET community), not outside attackers. We assume that a MANET member key, as a symmetric key, is used for communications among members to prevent outside attackers. A malicious node may

also perform data modification attacks to ruin the trust of a good node. PKI with assured digital signing [50] can be used to ensure data trustworthiness via source authenticity, integrity, and non-repudiation. PKI can also be used to uniquely identify each node and can be useful to defend against identity or Sybil attackers [101]. A malicious node may also perform denial of service (DoS) attacks to overwhelm an SP. Counter-DoS mechanisms [1] [4] [96] can be used to make DoS attacks largely ineffective to mitigate such attacks. Defending against communication-level attacks such as Denial-of-Service, identity or Sybil attacks is outside the scope of this dissertation research. We assume such behaviors are detected by intrusion detection mechanisms [9] [17] [44] [101] [161].

Chapter 4

CATrust: Context-Aware Trust Management for Service-Oriented Ad Hoc Networks

In this chapter we propose a context-aware trust management model called CATrust for service-oriented MANETs. The novelty of our design lies in the use of logit regression to dynamically estimate trustworthiness of a service provider based on its service behavior patterns in response to context environment changes. We develop a recommendation filtering mechanism to effectively screen out false recommendations even in extremely hostile environments in which the majority recommenders are malicious. We demonstrate desirable convergence, accuracy, and resiliency properties of CATrust. We also demonstrate that CATrust outperforms contemporary peer-to-peer and Internet of Things trust models in terms of service trust prediction accuracy against collusion recommendation attacks.

4.1 CATrust Design

4.1.1 Node Service Behavior Model

We consider the notion of context-sensitive service behavior as illustrated in Figure 4.1, i.e., an SP's service behavior may change dynamically, as the service-oriented MANET operational and environmental conditions change dynamically due to node mobility, channel contention (e.g., when local traffic is heavy), node status (e.g., energy status), and social disposition toward other nodes in the system. Trust therefore is dynamic because SPs are heterogeneous in terms of capability and attitude, and adapt to context changes. We call an operational or environmental condition that may affect an SP's service behavior a *context* variable. While CATrust can handle any context variable, we consider profit-awareness, capability-limitation, and energy-sensitivity as three distinct context variables for the following reasons: (a) a profit-aware SP is more likely to provide quality service when the SR offers a higher price [56] [157]; (b) an SP is

likely to provide inferior service when it is limited in resources and capability [89]; and (c) an SP is more likely to provide inferior service when the cost of servicing the task is high [87]. For example, in a congested environment the probability of wireless channel contention and signal interference will be high, so it will cost more for an SP to execute a service because the SP needs to consume more energy in listening to the channel and repeating packet transmission.

The service quality provided by an SP is determined by its service behavior in response to changes in the context environment. We call the resulting service quality "ground truth" service quality as it is intrinsically related to an SP's service behavior. A malicious node, however, may provide inferior service for self-interest even if it is capable of providing satisfactory service.

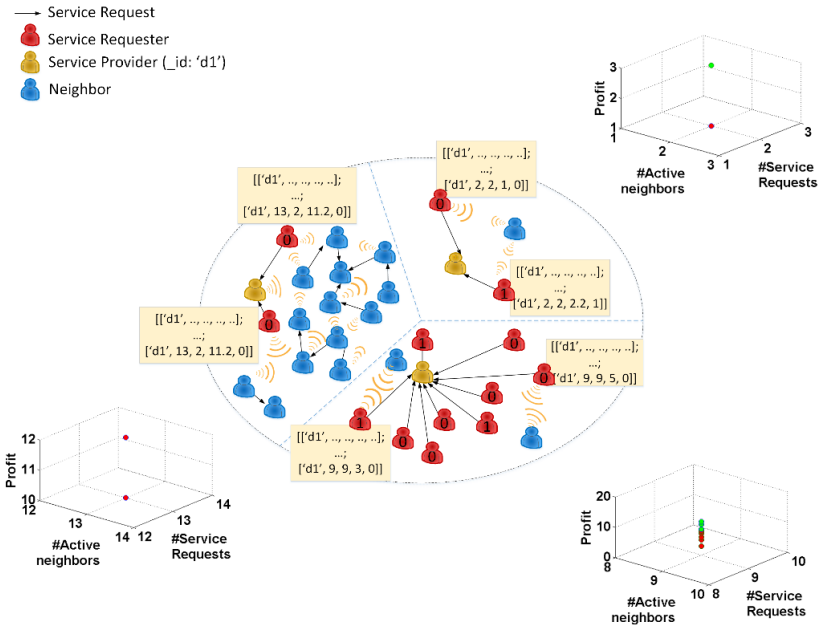


Figure 4.1: Context-Sensitive Service Behavior.

4.1.2 Problem Definition and Design Objective

The central idea of CATrust is that instead of directly predicting service quality, we predict the probability of delivering satisfactory service given service context as input.

For ease of discussion, we list the symbols used and their meanings in Table 4.1. A symbol may be associated with a special character to denote a special meaning, with "_" (underscore) denoting a vector/matrix, "^" (hat) denoting an inferred/predicted value, and "~" (tilde) denoting a set of self-observations and recommendations. We use *i* to refer to an SR, *j* to refer to an SP, and *k* to refer to a recommender. We assume

transportation/link failures can be identified by protocols in lower networking layers. The delivered service quality received by an SR is used by the SR to assess the service outcome. In order words, SR i would receive SP j's delivered service quality after context influence such as environment noise error which would contribute to the delivered service quality.

The problem at hand is for SR i to predict whether SP j will perform satisfactorily or not for a requested service in a particular context environment, given a history of evidence. The objective is to achieve high prediction accuracy in terms of correctly predicting bad service while not missing good service from an SP. Here we note that a node, malicious or not, can provide good service or bad service depending on the context environment.

Table 4.1: Notation.

Notation	Meaning
i	node i normally referring to a service requestor (SR)
k	node k normally referring to a recommender
j	node j normally referring to a service provider (SP)
T_j^t	j's service trustworthiness at time t
T_{ij}^t	j's service trustworthiness at time t as predicted by node i
x_m^t	m^{th} context variable value observed at time t
$\underline{x}^t$	$[x_0^t, \dots, x_M^t]$ denoting a set of context variable values observed at time t
s_j^t	actual service quality delivered by j at time t
$\hat{s}_{ij}^t$	s_j^t as predicted by i
s_{ij}^t	i's self-observation of service quality of j at time t
$\underline{\tilde{S}}_{ij}^t$	a set of self-observations and recommendations received by node i for service quality of j at time t
$\underline{\tilde{S}}_{ij}$	$\left[\underline{\tilde{S}}_{ij}^{t_0}, \dots, \underline{\tilde{S}}_{ij}^{t_n}\right]$ denoting cumulative evidence gathered by SR i regarding SP j's service quality over time
$\underline{X}$	$\left[\underline{x}^{t_0}, \dots, \underline{x}^{t_n}\right]$ denoting a matrix of context variable value sets over time corresponding to $\underline{\tilde{S}}_{ij}$
$\underline{\beta}_j$	$\left[\beta_j^0, \dots, \beta_j^M\right]$ denoting a set of regression coefficients
$\underline{\hat{\beta}}_{ij}$	i's estimate of $\underline{\beta}_j$

Within a specific type of service, SR i's observation s_{ij}^t at time t of the service quality received from SP j is either "satisfactory" or "unsatisfactory." If the service quality is satisfactory, then $s_{ij}^t = 1$ and SP j is considered *trustworthy;* otherwise, $s_{ij}^t = 0$ and SP j is considered *untrustworthy*. Let the operational and environmental conditions at time t be characterized by a set of distinct context variables $\underline{x}^t = [x_0^t, \dots, x_M^t]$. Then, SP

j's *service trust* is the probability that SP j is capable of providing satisfactory service given context variable $\underline{x}^t$, i.e., $T_j^t \triangleq \Pr(s_j^t = 1)$.

Let k ($k \neq i$) be a recommender who had prior service experience with SP j and is asked by SR i to provide its feedback regarding SP j. The recommendation from node k is in the form of $[\underline{x}^t, s_{kj}^t]$ specifying the context $\underline{x}^t$ under which the observation s_{kj}^t was made. Since k might launch recommendation attacks, it might report a false observation to i, in which case s_{kj}^t reported is $1 - s_{kj}^t$. Let $\underline{\tilde{S}}_{ij} = \left[\underline{\tilde{S}}_{ij}^{t_0}, \ldots, \underline{\tilde{S}}_{ij}^{t_n}\right]$, $i \neq j$, denote the cumulative evidence gathered by SR i over $[t_0, \ldots, t_n]$, including self-observations and recommendations. Also let $\underline{X} = \left[\underline{x}^{t_0}, \ldots, \underline{x}^{t_n}\right]$ denote the corresponding context matrix.

CATrust learns the service behavior pattern of SP j based on $\underline{\tilde{S}}_{ij}$ and $\underline{X}$, and predicts the probability that SP j is trustworthy at time t_{n+1}, given $\underline{x}^{t_{n+1}}$ as input. Suppose that node j follows service behavior pattern $\underline{\beta}_j$. Then, our prediction problem is to estimate $T_{ij}^{t_{n+1}} = \Pr\{\hat{s}_{ij}^{t_{n+1}} = 1 | \underline{x}^{t_{n+1}}, \underline{\hat{\beta}}_{ij}\}$, where $\underline{\hat{\beta}}_{ij}$ is node i's estimate of $\underline{\beta}_j$. Essentially, $T_{ij}^{t_{n+1}}$ obtained above is the service trust of SP j at time t_{n+1} from SR i's perspective.

4.1.3 Performance Metrics

The performance metrics for comparative performance analysis are false negative probability (P_{fn}) and false positive probability (P_{fp}) defined as follows:

- *False negative probability* (P_{fn}) is the misidentifying bad service probability. That is, it is the conditional probability that SR i will misidentify SP j as being able to provide satisfactory service, given that SP j actually provides unsatisfactory service. The term false negative is consistent with that used in intrusion detection [101], although the target subject in intrusion detection is the node itself (misidentifying a malicious node) rather than the service provided (misidentifying a bad service provided by a node). P_{fn} can be calculated by:

$$P_{fn} = \frac{\widehat{N}^c}{N^c} \tag{4.1}$$

  $\widehat{N}^c$ is the number of cases SR i believes SP j will provide satisfactory service while SP j actually provides unsatisfactory service, and N^c is the number of cases SP j will provide unsatisfactory services if selected. The lower the false negative rate the better the performance.
- *False positive probability* (P_{fp}) is the missing good service probability. That is, it is the conditional probability that SR i will misidentify SP j as being unable to provide satisfactory service, given that SP j actually provides satisfactory service. Again the term false positive is consistent with that used in intrusion detection [22], although

the target subject in intrusion detection is the node itself (misidentifying a non-malicious node) rather than the service provided (missing a good service provided by a node). P_{fp} can be calculated by:

$$P_{fp} = \frac{\widehat{N}^m}{N^m} \tag{4.2}$$

$\widehat{N}^m$ is the number of cases SR i believes SP j will not provide satisfactory service while SP j actually provides satisfactory service and N^m is the number of cases SP j will provide satisfactory service if selected. The lower the false positive rate the better the performance.

The basic difference of our definition of P_{fn} vs. the convention definition of P_{fn} in the intrusion detection field is that our P_{fn} is the probability of misidentifying a bad service given that the service delivered is bad, while the convention P_{fn} is the probability of misidentifying a bad node given that the node is bad. The difference in the definition of P_{fp} can be similarly derived.

4.1.4 Trust Computation

We utilize a sigmoid function to link the binary observation of service quality with context variables in a continuous range. More specifically, we utilize robust logistic regression [85] to analyze the relation between $\underline{\tilde{S}}_{ij}$ and $\underline{X}$. While many forms exist for relating $\underline{\tilde{S}}_{ij}$ and $\underline{X}$, we adopt a linear model for its simplicity and effectiveness, treating SP j's behavior pattern $\underline{\beta}_j = [\beta_j^0, \dots, \beta_j^M]$ essentially as a set of regression coefficients matching the context variable set $\underline{x}^t = [x_0^t, \dots, x_M^t]$. Later in Section 4.5, we discuss the feasibility of using other models.

We assume observations are mutually independent and that the order of observations can be changed. Following the linear model for a classical logistic regression problem, j's service trustworthiness at time t is modeled by:

$$T_j^t = \Pr\left\{s_j^t = 1 \middle| \underline{x}^t, \underline{\beta}_j\right\} = \left(1 + \exp(-(\underline{x}^t)^\top \underline{\beta}_j)\right)^{-1} \tag{4.3}$$

Or, equivalently, following the logit function definition, $\text{logit}(y) = \ln\left(\frac{y}{1-y}\right)$, we have:

$$\text{logit}(T_j^t) = (\underline{x}^t)^\top \underline{\beta}_j \tag{4.4}$$

With (4.3), i estimates j's trust T_j^t based on its estimated $\underline{\beta}_j$. To do so, i needs to estimate $\underline{\beta}_j$, but it only has noisy observations of the service history. We model this by:

$$z_{ij}^t = \text{logit}(T_{ij}^t) = (\underline{x}^t)^\top \underline{\beta}_j + \varepsilon_{ij}^t \tag{4.5}$$

where T_{ij}^t is j's service trustworthiness at time t as predicted by node i, and ε_{ij}^t is an independent error term following the logistic(0,1) distribution with the cumulative density function $\frac{1}{1+e^{-y}}$, $y \in (-\infty, \infty)$.

A malicious recommender k can modify all unsatisfactory observations on j to "1" in a ballot-stuffing attack, while reversing all satisfactory services to "0" in a bad-mouthing attack. Malicious behaviors result in "outliers" which will lead to inaccurate estimation. To tolerate recommendation attacks without overly sacrificing solution accuracy, we replace the error term ε_{ij}^t in (4.5) with a noise term that has the standard t-distribution with v degrees of freedom. The t-distribution has heavier tails when v is finite, which increases the ability to absorb outlier errors and provides robust estimates of $\underline{\beta}_j$.

After replacing ε_{ij} by a standard t-distribution random variable, denoting $\left(\underline{x}^t\right)^\top \underline{\beta}_j$ as u^t, and providing a value for the hyper parameter v, z_{ij}^t in (5) has the following density function:

$$f_v(z) = (\pi v)^{-\frac{1}{2}} \Gamma\left(\frac{v+1}{2}\right) \Gamma^{-1}\left(\frac{v}{2}\right) \left(1 + \frac{(z-u)^2}{v}\right)^{-\frac{v+1}{2}} \tag{4.6}$$

where Γ is the gamma function. We apply Bayesian inference based on the data augmentation algorithm with Markov Chain Monte Carlo (MCMC) [54] [109] to infer $\underline{\beta}_j$ given historic observations $\underline{\tilde{S}}_{ij}$, as follows:

$$p\left(\beta_j \middle| \underline{\tilde{S}}_{ij}\right) = \int p\left(\beta_j \middle| \underline{z}_{ij},\ \underline{\tilde{S}}_{ij}\right) p\left(\underline{z}_{ij} \middle| \beta_j, \underline{\tilde{S}}_{ij}\right) d\underline{z}_{ij} \tag{4.7}$$

where $\underline{z}_{ij} = [z_{ij}^{t_0}, \ldots, z_{ij}^{t_n}]$ is a latent variable set introduced by the data augmentation algorithm in order to construct known $p\left(\underline{\beta}_j \middle| \underline{z}_{ij},\ \underline{\tilde{S}}_{ij}\right)$. However, due to the t-distribution for ε_{ij}^t, there is no closed-form solution for $p\left(\underline{\beta}_j \middle| \underline{z}_{ij},\ \underline{\tilde{S}}_{ij}\right)$. To circumvent this, z_{ij}^t is approximated by a scale mixture of Gaussian distribution, i.e., $z_{ij}^t | \omega_{ij}^t \sim \mathcal{N}(u^t, \left(\omega_{ij}^t\right)^{-1})$ with $\omega_{ij}^t \sim \Gamma(v/2, v/2)$ where $\mathcal{N}(u^t, \left(\omega_{ij}^t\right)^{-1})$ is Gaussian distribution with mean u^t and variance $\left(\omega_{ij}^t\right)^{-1}$ and $\Gamma(v/2, v/2)$ is Gamma distribution with shape $v/2$ and scale $v/2$. Let $\underline{\omega}_{ij} = \left[\omega_{ij}^{t_0}, \ldots, \omega_{ij}^{t_n}\right]$. Then (4.7) can be rewritten as:

$$p\left(\underline{\beta}_j \middle| \underline{\tilde{S}}_{ij}\right) = \int \int p\left(\underline{\beta}_j \middle| \underline{z}_{ij}, \underline{\omega}_{ij}, \underline{\tilde{S}}_{ij}\right) p\left(\underline{z}_{ij}, \underline{\omega}_{ij} \middle| \underline{\beta}_j, \underline{\tilde{S}}_{ij}\right) d\underline{\omega}_{ij} d\underline{z}_{ij} \tag{4.8}$$

Assuming a Gaussian priori to $\underline{\beta}_j$ with known mean and variance defined above, the posterior $\underline{\beta}_j | \underline{z}_{ij}, \underline{\omega}_{ij}, \underline{\tilde{S}}_{ij}$ will follow a Gaussian distribution and the posterior

$\underline{\omega}_{ij}|\underline{z}_{ij}, \underline{\beta}_j$ will follow a Gamma distribution [54] [109]. Meanwhile, $\underline{z}_{ij}|\underline{\beta}_j, \underline{\tilde{S}}_{ij}$ is a truncated t-distribution, depending on the value of $\underline{\tilde{S}}_{ij}$. We then apply an iterative sampling procedure to first draw a new $(\underline{z}_{ij}, \underline{\omega}_{ij})$ and then produce a new $\underline{\beta}_j$ from $p\left(\underline{\beta}_j \middle| \underline{z}_{ij}, \underline{\omega}_{ij}, \ \underline{\tilde{S}}_{ij}\right)$. This process is repeated iteratively until the Markov Chain is stabilized. The final $\underline{\beta}_j$ obtained is node i's estimated $\underline{\beta}_j$, i.e., $\underline{\hat{\beta}}_{ij}$. Node i can then compute T_{ij}^{t+1} by (4.3), given $\underline{x}^{t+1}$ and $\underline{\hat{\beta}}_{ij}$ as input.

4.1.5 Recommendation Filtering

The logistic regression model with t-distribution error enables fairly robust learning of an SP's service behavior pattern based on a trustor's self-observations and the recommenders' history records. However, for any statistical model, there is a breakpoint. In our case, if the percentage of malicious recommenders is too high, it can hardly differentiate true from false recommendations, thus resulting in a low accuracy rate. One possible solution is to seek socially connected peers (friends) who are most likely to deliver true recommendations. However, the limitation is that it depends on the availability of friends in the neighborhood and also it requires friends to have direct service experiences with the targeted SP, a condition that may be difficult to meet in service-oriented MANET environments.

We propose a novel threshold-based recommendation filtering mechanism. The main idea is to compare the SR's prediction of the service trust toward the trustee with the recommender's report toward the same trustee under the same context environment. Specifically each time node k, serving as a recommender, propagates a recommendation about node j in the form of $[\underline{x}^t, \ s_{kj}^t]$, node i will apply the current predictor $\underline{\hat{\beta}}_{ij}$ with $\underline{x}^t$ given as input to compute T_{ij}^t. Ideally, if k delivers true s_{kj}^t, T_{ij}^t should be closer to s_{kj}^t than to $1 - s_{kj}^t$. To filter out malicious recommendations, i uses a threshold parameter T^{th}. If $|T_{ij}^t - s_{kj}^t|$ is larger than T^{th}, s_{kj}^t is considered modified and consequently $[\underline{x}^t, \ s_{kj}^t]$ is rejected by i; otherwise, it will be accepted by i and put in the training set for updating $\underline{\hat{\beta}}_{ij}$. In case a recommender provides several recommendations for several distinct context environments, the average difference can be first computed before applying threshold-based recommendation filtering. The threshold parameter T^{th} is an important parameter whose effect on protocol performance will be analyzed in Section 4.2.

4.1.6 Characteristics of Context Variables

Context variables must obey the property that in similar context, the service from an SP performs similarly. Therefore, context variables are inherently tied to an SP's service

behavior and the service quality criteria defined by an application. In our example SOANET application [137], service quality is defined by three criteria, namely, QoI, service delay, and service cost. Consequently, energy (which influences QoI [87]), local traffic (which influences service delay), and incentive (which influences service cost [56]) are natural choices for this application. As these context variables have clear physical meanings, the range of a context variable can be defined accordingly. For example, the energy context variable can be categorized as $[high, medium, low]$ denoting the energy status of an SP. The incentive context variable can be the price paid to an SP upon satisfactory completing of service in the range of $[minimum\ price, maximum\ price]$. The local traffic context variable can be the number of neighbors simultaneously transmitting packets with the range of $[0, maximum\ node\ density \times radio\ range\ area]$. A range with a finer granularity allows CATrust to more accurately learn an SP's service behavior pattern and service quality at the expense of computational complexity (see Section 4.5.1 for a discussion). Another property that must be satisfied is that a context variable must be measurable at runtime. For the example SOANET application, the energy status of SP j can be measured by the SR by counting the ratio of the number of acknowledgement packets received from SP j over the total number of transmitted packets from the SR to SP j during the encounter interval. The incentive to SP j is determined by the SR itself so it is easily measurable by the SR. The local traffic can be estimated by the SR based on the collision probability or the packet retransmission probability after transmitting a sequence of packets for initiating a service request.

4.1.7 Computational Procedure

Figure 4.2 illustrates how an SR learns and predicts the probability that a target SP will provide quality service (or trust) in a context environment based on the context-annotated service records collected. Lines 2-15 specify that SR would apply the recommendation filtering mechanism to accept or reject a new recommendation from a newly encountered recommender. If the recommendation is accepted, the SR would invoke the CATrust learning mechanism to learn the target SP's service behavior. Lines 16-23 specify the SR would invoke the CATrust prediction mechanism to predict the probability that a target SP will provide satisfactory service (or trust), given the current context as input. The SR then would decide if it should select the target SP for service execution depending on the comparison result between the predicted trust value provided by CATrust and the minimum trust threshold.

4.2 Analysis of Convergence, Accuracy, and Resiliency Properties of CATrust

```
# i: SR, k: SR, j: SP, i ≠ j, i ≠ k
1: initialize historical_observations ← ∅, behavior_patterns ← [[0, 0, ... ] for all SPs]

# Trust Aggregation
2: when encounter(i, k) do
3:     updated_SPs ← ∅
4:     for each observation in k's historical_observations do
5:         target_SP ← getTargetSP(observation)
6:         filtered ← recommendationFilter(observation, behavior_patterns[taget_SP])
7:         if not filtered then
8:             updated_SPs ≪ target_SP
9:             addObserveration(observation, historical_observertions)
10:        end
11:    end
12:    for each sp in updated_SPs do
13:        CATrustLearn(sp, behavior_patterns, historical_observations)
14:    end
15: end

#  Decision Making
16: when hasService(j, service) do
17:    trust ← CATrustPredict(context_vector, behavior_patterns[j] )
18:    if trust < trust_threshold then
19:        exploreAnotherSP()
20:    else
21:        requestServiceFrom(j)
22:    end
23: end
```

Figure 4.2: Computational Procedure for Trust Aggregation and Decision Making.

In this section, we analyze the convergence, accuracy, and resiliency properties of CATrust against collusion recommendation attacks. We first describe the environment setup and then we present the results.

4.2.1 Environment Setup

Table 4.2 lists a set of parameters and their values/ranges used in our analysis. We classify all parameters into *input*, *derived*, and *design*. Input parameters characterize the operational and environmental conditions, given as input. Derived parameters hold intermediate results derived from input parameters and are used for ease of referencing. Design parameters are protocol or algorithm variables which are to be optimized to maximize protocol or algorithm performance.

We simulate a service-oriented MANET with n_n nodes and the operational area being a rectangular area A, The radio transmission range is R, and the mobility model is RWM model without considering mobility dependency among nodes or geography

obstacles. Under RWM, every node moves randomly, with speed S, movement time W, and pause time P defining the movement pattern such that the average encounter rate between any two nodes is approximately α. We simulate encountering events (i.e., when nodes are in the same subarea within radio range) at which service requests are issued and service quality received are recorded, and recommendations are exchanged. Only one service type is considered for simplicity. A node acting as an SR has a service request rate of λ upon encountering a potential SP. A node can provide recommendations only to nodes with which it has had service experiences. The measurement interval for data collection is L. The trust update interval is t_{unit}.

Table 4.2: Parameters and Their Default Values.

Notation	Meaning	Default Value	Type
A	Operational area	800x800 m^2	Input
S	Speed	[1.0, 2.5] m/s	Input
P	Pause time	[0, 60] s	Input
W	Movement time	[5*60, 15*60] s	Input
R	Radio range	220 m	Input
α	Encountering rate	5/hr	Input
λ	Service request rate	5/encounter	Input
L	Length of measurement interval	24 hr	Input
n_n	Number of nodes	50	Input
p_{unit}	Unit price	1	Input
ν_0	Degree of freedom	7	Input
P_b	Percentage of bad nodes	[10%-70%]	Input
P_{ssr}	Satisfactory service ratio	60%	Input
n_r	Number of service records per node	$\alpha\lambda L$	Derived
n_c	Number of context variables	3	Design
t_{unit}	Trust update interval	7.2min	Design
T^{th}	Recommendation filtering threshold	[0-1]	Design

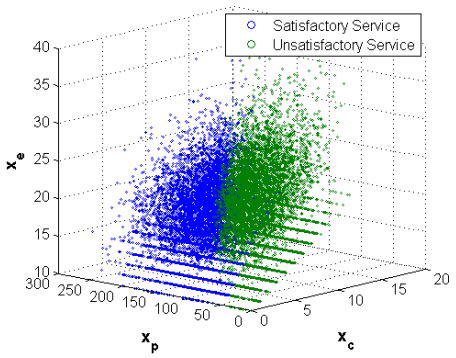


Figure 4.3: Visualization of Synthetic Data.

We consider three context variables in the experiment, namely, energy-sensitivity ($\underline{x}_e$), capability-limitation ($\underline{x}_c$), and profit-awareness ($\underline{x}_p$). The values of context variables are generated as follows: x_e^t is measured by the number of neighbors sharing the channel as more energy is consumed for channel contention and packet retransmission when there are more nodes sharing the channel. x_c^t is measured by the number of service requests to be processed in an SP's queue as high traffic to the SP hinders its processing capability. x_p^t is SP's potential gain upon satisfactory service completion. The potential gain consists of two parts: the asked price P_{ask} from an SP, calculated by multiplying the queue length with the unit price p_{unit}, and the overpaid price p_{over} by the SR that represents the overpaying incentive, modeled by a normal distribution with mean and variance being 50% and 12.5% of the asked price P_{ask}, respectively. Once $\left[x_e^t, x_c^t, x_p^t\right]$ is generated, we generate s_g^t (ground truth service satisfaction) such that the average satisfactory service ratio is P_{ssr}. An SP, whether malicious or not, has its own P_{ssr} and specific context environment instances under which it can provide satisfactory service within its capability. Figure 4.3 shows a snapshot of s_g^t (ground truth service satisfaction) vs. $\left[x_e^t, x_c^t, x_p^t\right]$ for an SP with $P_{ssr} = 60\%$.

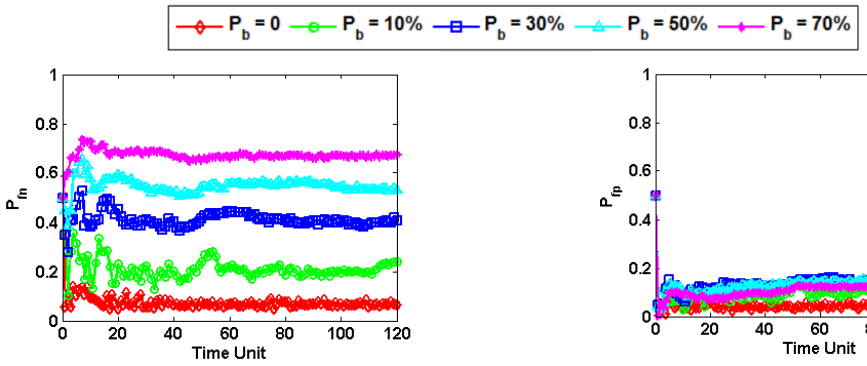
The hostility level is described by the percentage of malicious nodes (P_b) whose effect will be analyzed. Malicious nodes are randomly picked and will perform attacks as described in the threat model. For CATrust, the degree of freedom ν_0 is set to 7 (as in [109]) for approximating the original logistic regression. We analyze the effect of the recommendation filtering threshold parameter (T^{th}) on protocol performance.

4.2.2 Convergence, Accuracy and Resiliency Properties

We use MATLAB to implement the algorithm and collect numerical data for analyzing the convergence, accuracy, and resiliency properties of CATrust against collusion recommendation attacks. The performance metrics are false negative probability (P_{fn}) and false positive probability (P_{fp}) described earlier. Each data point reported is the average of 100 randomly generated test cases in a test data set $\left[\underline{x}_e, \underline{x}_c, \underline{x}_p, \underline{s}_g\right]$. Specifically, for the case in which s_g (ground truth) is 1 (satisfactory service) and the service trust predicted by CATrust is T_{ij}^t, then $P_{fp} = 1 - T_{ij}^t$ because $1 - T_{ij}^t$ is the belief that the service provided will be unsatisfactory so it is the missing good service probability. For the case in which s_g (ground truth) is 0 (unsatisfactory service) and the service trust predicted by CATrust is T_{ij}^t, then $P_{fn} = T_{ij}^t$ because T_{ij}^t is the belief that the service provided will be satisfactory so it is the misidentifying bad service probability.

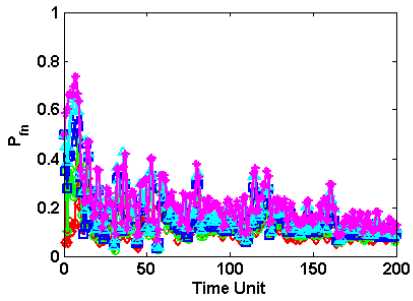
Figure 4.4 shows P_{fn}/P_{fp} vs. time as P_b (the percentage of malicious nodes) varies in the range of 0-70%, for a malicious trustee SP randomly picked (with $P_{ssr} = 60\%$). The

top (bottom) 2 graphs are without (with) recommendation filtering. With recommendation filtering, if the difference between the predicted service trust and the recommended service trust under the same context environment is above a threshold, then the recommendation is filtered. We intentionally used the same (and full) scale for all graphs, so we can visually see the sensitivity of P_{fn}/P_{fp} values with respect to P_b.

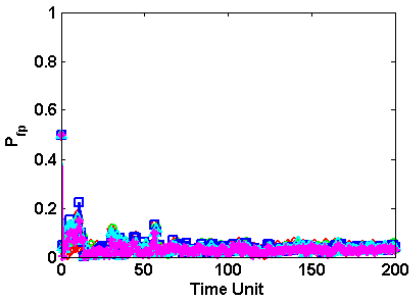


(a) P_{fn} vs. time without recommendation filtering

(b) P_{fp} vs. time without recommendation filtering



(c) P_{fn} vs. time with recommendation filtering



(d) P_{fp} vs. time with recommendation filtering

Figure 4.4: Convergence, Accuracy, and Resiliency Behavior of a Malicious Trustee SP. (a) and (b) are without Recommendation Filtering. (c) and (d) are with Recommendation Filtering.

First of all, we see fast convergence behavior in both cases without much sensitivity to P_b. However, without recommendation filtering, the prediction accuracy of P_{fn} (see Figure 4.4 (a)) is inversely related to P_b. The reason is that without recommendation filtering, as P_b increases, an SR will receive more and more high but false service trust recommendations from more malicious nodes performing ballot-stuffing attacks. These malicious trust recommendations cause the SR to misidentify bad service provided by the malicious node. Second, from Figure 4.4 (c), we observe that with recommendation filtering, CATrust is able to effectively filter out false recommendations and, as a result, converges to the same low P_{fn} value for high accuracy eventually. Note that the ideal P_{fn} is 0. Hence, low P_{fn} close to 0 after convergence means high accuracy. This

demonstrates that CATrust with recommendation filtering is resilient to ballot-stuffing attacks, even in extremely hostile environments. Last, from comparing Figure 4.4 (b) and (d), we observe that recommendation filtering has a relatively small effect on CATrust's high prediction accuracy of P_{fp}. This is because ballot-stuffing attacks can boost bad services but cannot further boost already good services provided by a malicious node.

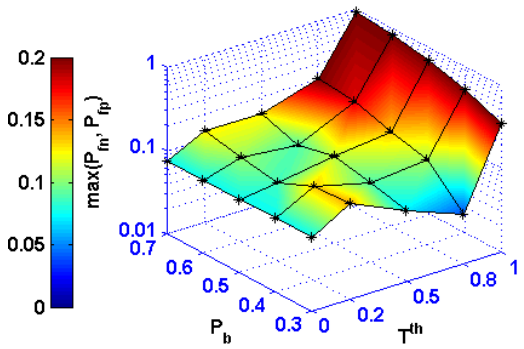
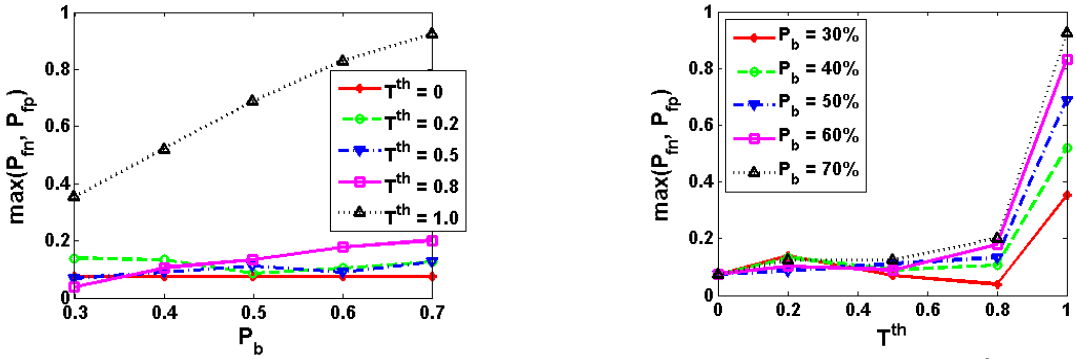


Figure 4.5: Effect of Recommendation Filtering Threshold T^{th} on max(P_{fn}, P_{fp}) for a Malicious SP.

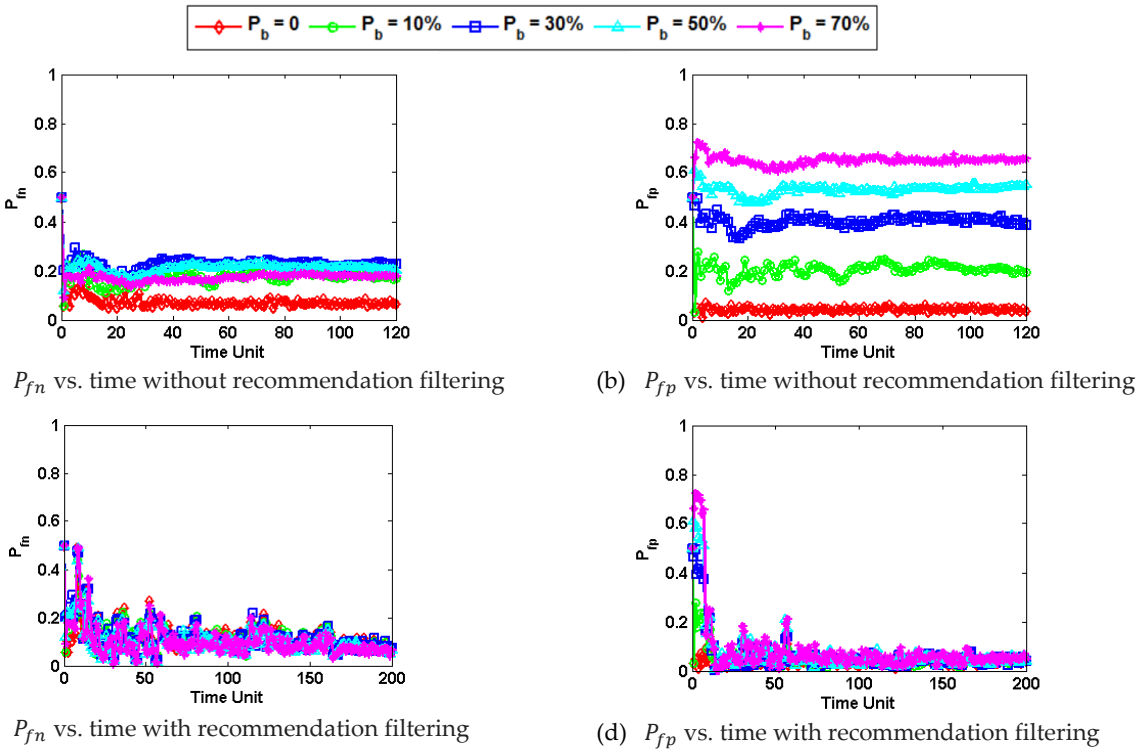


(a) max(P_{fn}, P_{fp}) vs. P_b (b) max(P_{fn}, P_{fp}) vs. T^{th}

Figure 4.6: Two-dimensional View of Figure 4.5.

Figure 4.5 analyzes the sensitivity of CATrust performance with respect to the recommendation filtering threshold T^{th}, a design parameter in our trust protocol design. Figure 4.6 shows the two-dimensional view of Figure 4.5. We use $\max(P_{fn}, P_{fp})$ as the performance metric to identify the best T^{th} for maximizing performance, because there is a tradeoff between P_{fn} and P_{fp}. That is, as the optimal trust threshold T^{th} increases, P_{fn} decreases while P_{fp} increases. Minimizing both is desirable. We observe from Figure 4.5 and Figure 4.6 that there exists an optimal T^{th} at which $\max(P_{fn}, P_{fp})$ is minimized, given P_b as input. For example when $P_b = 0.3$, the optimal T^{th} is 0.8, but when $P_b = 0.4$, the optimal T^{th} is 0.5. This result suggests that one should dynamically

adjust T^{th} to adapt to changes in hostility conditions in order to maximize application performance, i.e., minimizing both P_{fn} and P_{fp}.



(a) P_{fn} vs. time without recommendation filtering

(b) P_{fp} vs. time without recommendation filtering

(c) P_{fn} vs. time with recommendation filtering

(d) P_{fp} vs. time with recommendation filtering

Figure 4.7: Convergence, Accuracy and Resiliency Behavior of CATrust for a Non-malicious Trustee SP. (a) and (b) are without Recommendation Filtering. (c) and (d) are with Recommendation Filtering.

Correspondingly, Figure 4.7 shows P_{fn}/P_{fp} vs. time as P_b varies in the range of 0-70%, for a non-malicious trustee SP with $P_{ssr} = 60\%$. Here the trustee SP is non-malicious, so malicious nodes will perform bad-mouthing attacks to ruin its service trust, which, as opposite to ballot-stuffing attacks, will affect P_{fp} more than P_{fn}. We observe from Figure 4.7 (b) that without recommendation filtering, the prediction accuracy of P_{fp} is indeed inversely related to P_b. The reason is that without recommendation filtering, as P_b increases, an SR will receive more and more low service trust but false recommendations from more malicious nodes performing bad-mouthing attacks. These malicious recommendations cause the SR to misidentify good service provided by the non-malicious SP, which is downgraded due to bad-mouthing attacks. From Figure 4.7 (d), with recommendation filtering, CATrust is able to effectively filter out false recommendations and, as a result, converges to the same low P_{fp} value eventually as time progresses. This result demonstrates that CATrust with recommendation filtering is resilient to bad-mouthing attacks. From comparing Figure 4.7 (a) and (c), we observe that recommendation filtering has a relatively small effect

on CATrust's high prediction accuracy of P_{fn}. This is because bad-mouthing attacks (on a non-malicious node) can effectively downgrade good services but cannot downgrade already bad services provided by a non-malicious node.

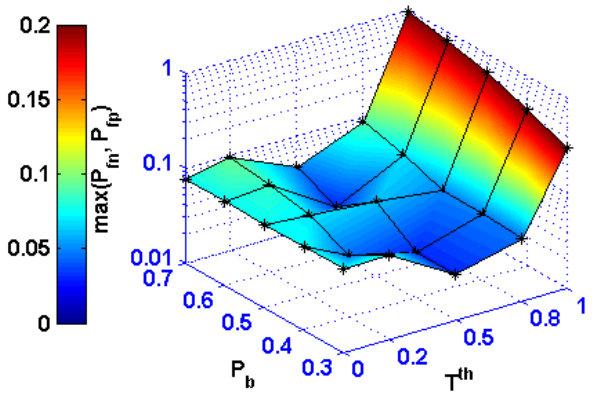
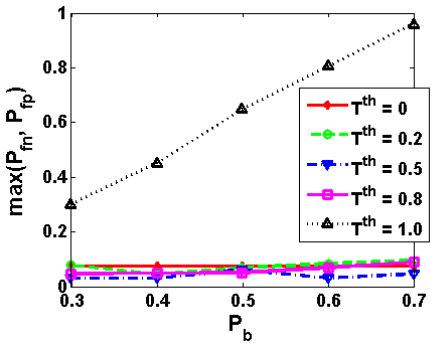
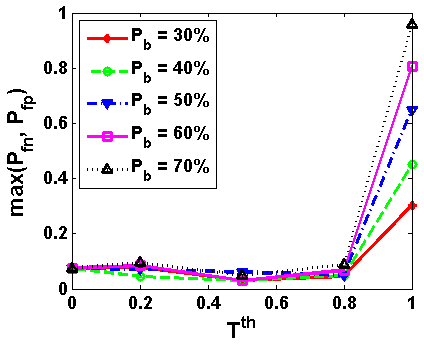


Figure 4.8: Effect of Recommendation Filtering Threshold T^{th} on max(P_{fn}, P_{fp}) for a Non-malicious SP.



(a) max(P_{fn}, P_{fp}) vs. P_b



(b) max(P_{fn}, P_{fp}) vs. T^{th}

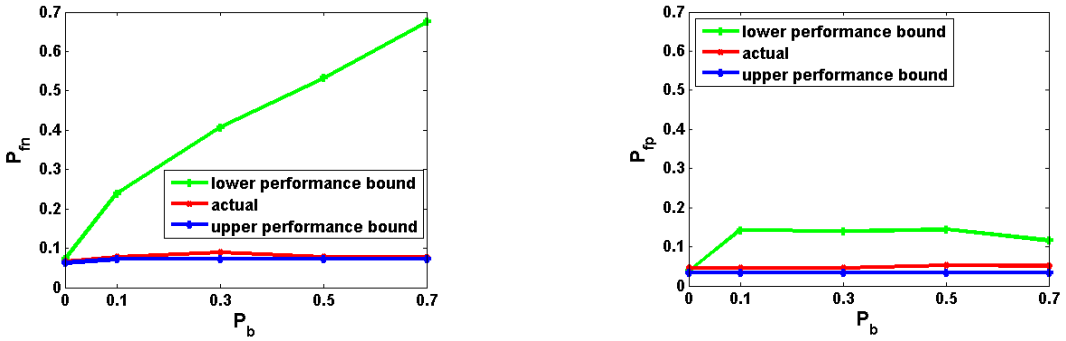
Figure 4.9: Two-dimensional View of Figure 4.8.

Figure 4.8 analyzes the sensitivity of CATrust performance with respect to the recommendation filtering threshold T^{th} when the trustee SP is non-malicious. Figure 4.9 shows the two-dimensional view of Figure 4.8. We again observe from Figure 4.8 and Figure 4.9 that there exists an optimal T^{th} at which $\max(P_{fn}, P_{fp})$ is minimized, given P_b as input. In conclusion, adjusting T^{th} dynamically to maximize application performance is a viable design. Our analysis paves the way for realizing adaptive control for protocol performance optimization.

4.2.3 Upper bound and Lower bound Performance

The upper bound performance of CATrust is obtained when an SR obtains only truthful service records from direct observations or recommenders. This will occur when (a) there is no malicious recommender (i.e., $P_b = 0$) and all recommendations are

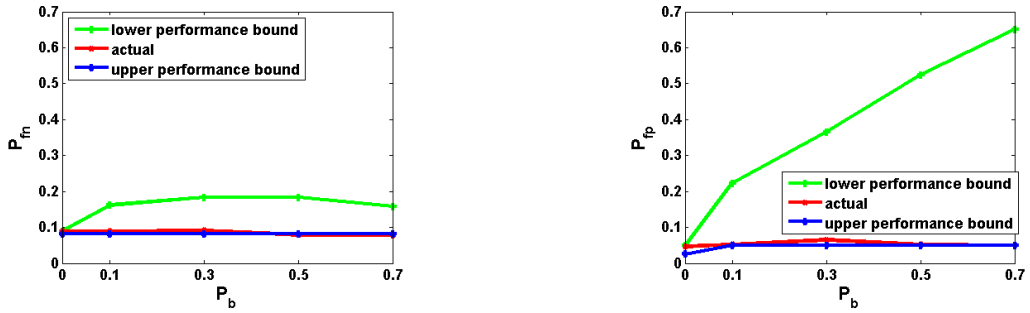
not filtered out by CATrust's filtering mechanism, or (b) when there are malicious recommenders but all malicious recommendations are filtered out by CATrust's filtering mechanism. On the other hand, the lower bound performance of CATrust is obtained when all malicious recommendations are not filtered out by CATrust's filtering mechanism.



(a) P_{fn} vs. P_b for a malicious trustee SP. (b) P_{fp} vs. P_b for a malicious trustee SP.

Figure 4.10: Performance Upper Bound and Lower Bound of CATrust for a Malicious Trustee SP.

Figure 4.10 shows the upper bound and lower bound performance of CATrust in terms of P_{fn} and P_{fp} (the lower the better) for a malicious trustee SP. We first observe that CATrust real performance (red curve) is close to the upper bound performance (blue curve), thus demonstrating the effectiveness of CATrust's filtering mechanism. Comparing Figure 4.10 (a) with Figure 4.10 (b), the effect of P_b is more pronounced on P_{fn} than on P_{fp}. This is because of ballot-stuffing attacks performed by malicious nodes, which causes the SR to misidentify bad service provided by the malicious trustee SP, especially for the lower bound performance case (green curve in Figure 4.10 (a)) when the filtering mechanism fails to filter out malicious recommendations.



(a) P_{fn} vs. P_b for a non-malicious trustee SP. (b) P_{fp} vs. P_b for a non-malicious trustee SP.

Figure 4.11: Performance Upper Bound and Lower Bound of CATrust for a Non-Malicious Trustee SP.

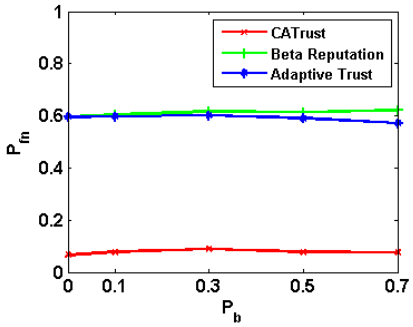
Figure 4.11 shows the upper bound and lower bound performance of CATrust for a non-malicious trustee SP. The results above show that CATrust's real performance is close to the upper bound performance, thus verifying the effectiveness of CATrust's filtering mechanism. Comparing Figure 4.11 (a) with Figure 4.11 (b), the effect of P_b is more pronounced on P_{fp} than on P_{fn} because of bad-mouthing attacks performed by malicious nodes. It causes the SR to misidentify good service provided by the non-malicious trustee SP, especially for the lower bound performance case (green curve in Figure 4.11 (b)) when the filtering mechanism fails.

4.3 Performance Comparison

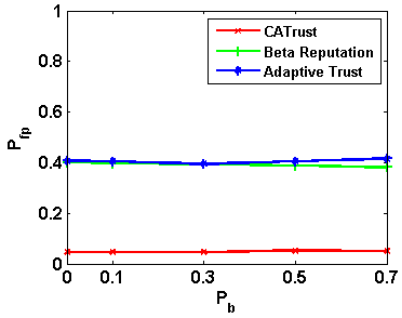
In this Section, we compare CATrust with Beta Reputation [69] and Adaptive Trust [31]. For fair comparison, we compare all three protocols at their optimizing conditions.

Figure 4.12 shows converged P_{fn}/P_{fp} of the three schemes for a malicious SP with $P_{ssr} = 60\%$, as P_b varies in the range of [0, 70%]. We observe that while all three protocols are resilient against collusion recommendation attacks, CATrust performs best by a wide margin. We notice that because the trustee SP is malicious, P_{fn} will be affected more than P_{fp} via ballot-stuffing attacks in this case.

Here we note that for a trustee SP, P_{fn} (misidentifying the node's bad service) tends to converge to the node's average service trust value, as evaluated by Beta Reputation and Adaptive Trust, because the node's average trust value maps to the trustee SP's satisfactory service ratio P_{ssr}. In the experiment setup, $P_{ssr} = 60\%$. Therefore, P_{fn} is close to 0.6 (as in Figure 4.12(a)). On the other hand, P_{fp} (missing the node's good service) tends to converge to $1 - P_{ssr}$. Consequently, P_{fp} is close to 0.4 (as in Figure 4.12(b)).

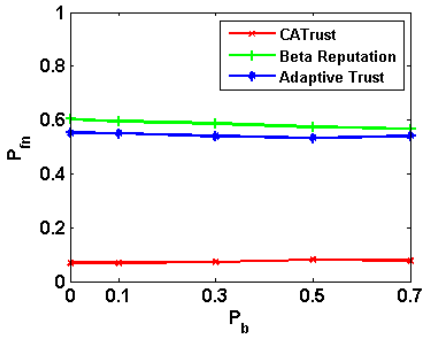


(a) P_{fn} vs. P_b for a malicious trustee SP.

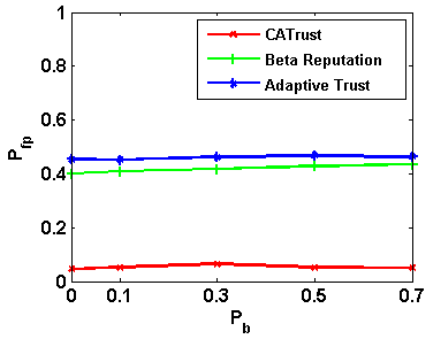


(b) P_{fp} vs. P_b for a malicious trustee SP.

Figure 4.12: Performance Comparison of CATrust vs. Beta Reputation and Adaptive Trust for a Malicious Trustee SP.



(a) P_{fn} vs. P_b for a non-malicious trustee SP.



(b) P_{fp} vs. P_b for a non-malicious trustee SP.

Figure 4.13: Performance Comparison of CATrust vs. Beta Reputation and Adaptive Trust for a Non-malicious Trustee SP.

Correspondingly Figure 4.13 compares protocol performance in terms of converged P_{fn}/P_{fp} values for a non-malicious SP with $P_{ssr} = 60\%$, as P_b varies in the range of [0, 70%]. We again observe that CATrust outperforms the other two by a wide margin. We notice that because the trustee SP is non-malicious, P_{fp} will be affected more than P_{fn}via bad-mouthing attacks in this case.

The superiority of CATrust over Beta Reputation and Adaptive Trust as demonstrated in Figure 4.12 and Figure 4.13 is attributed to the fundamental difference in trust protocol design logic. CATrust infers a service trust value for *each* context environment based on the trustee node's predicted service behavior in that context environment, while Beta Reputation or Adaptive Trust just maintains one service trust variable across all context environments. Consequently, for a malicious trustee SP (as in Figure 4.12), P_{fn} tends to converge to the malicious node's average service trust value which is equivalent to the malicious node's satisfactory service ratio $P_{ssr} = 60\%$. For a non-malicious trustee SP (as in Figure 4.13), P_{fp} tends to converge to $1 - P_{ssr} = 1 - 60\% = 40\%$. In contrast, our CATrust protocol is not bound by the satisfactory service ratio. Rather, by learning the trustee node's service behavior, CATrust infers a service trust value as close to the ground truth service satisfaction as possible in a particular context environment. The association of service trust with context results in high prediction accuracy, simply because the trust value inferred is tied to a specific context environment. Beta Reputation and Adaptive Trust, on the other hand, can only infer the average trust value across all context environments as context information is not taken into consideration in their trust protocol design.

4.4 Validation

We have performed a preliminary validation experiment using a synthesized trace composed from web service and mobility traces. The synthesized trace is composed by a two-step process: (a) the user satisfaction levels of service invocations are extracted from a real web service dataset; (b) the user satisfaction levels then are adjusted under the influences of "traffic load" (which depends on the current location of the service requester from the service provider at time t), "energy" (which depends on the distance separating an SR and an SP), and "price" (which varies from one service request to another). The adjustment is linear depending on the distance between the maximum and the minimum context variable values.

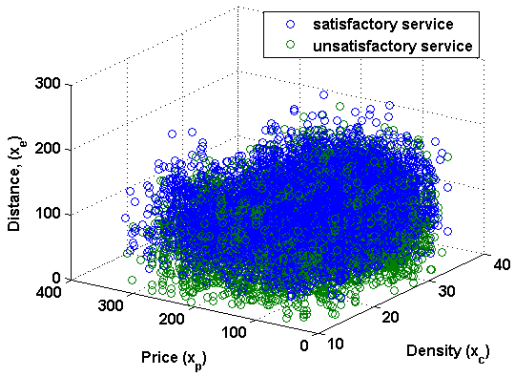
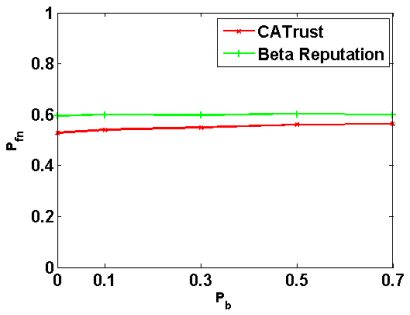


Figure 4.14: Visualization of Satisfactory vs. Unsatisfactory Service vs. $[x_e^t, x_c^t, x_p^t]$ for an SP with P_{ssr}= 60% in the Synthesized Trace.

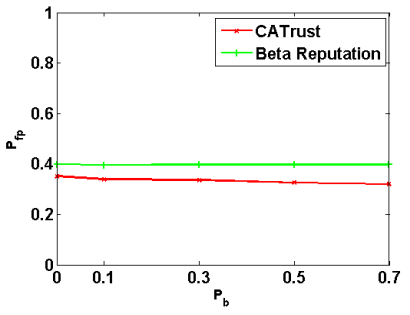
The web service request traces capture 150 SRs issuing service requests on 100 distinct SPs providing web services [160]. The traces record the data size, round trip time (RTT), response HTTP code, and response HTTP message. To tailor the trace data for a service-oriented MANET, we pick one web service as the target SP and all service users as the SRs, and combine the web service traces with mobility traces based on RWM according to the experimental setting listed in Table 4.2. The RTT is taken as the service quality metric which is affected by three context variables: traffic (measured by node density within radio range around the target SP), energy (measured by Euclidean distance between the SP and the SR), and price (measured by randomly generated payment to go with a service request). We classify a service as satisfactory (1) or unsatisfactory (0) by comparing the adjusted RTT with a minimum service quality threshold so that the average service satisfaction ratio P_{ssr} is 60%. Figure 4.14 visualizes satisfactory and unsatisfactory service vs. $[x_e^t, x_c^t, x_p^t]$ for an SP with $P_{ssr} = 60\%$ in the synthesized trace. We note that different from Figure 4.3, the data model is non-linear.

In Figure 4.15, the performance of CATrust is degraded significantly compared with that in Figure 4.12 and Figure 4.13. A possible explanation is that when classifying

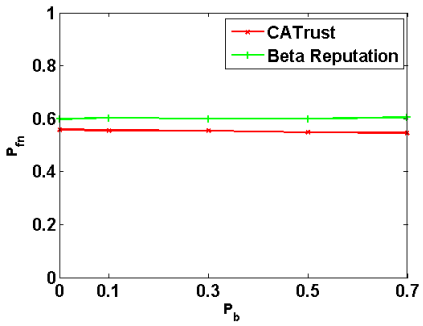
service, CATrust assumes a linear relationship between service quality and context for learning. However while synthesizing the trace data, we compare the adjusted RTT with a minimum service quality threshold to determine if service is satisfactory or not, and, as a result, the linear relationship between context and service quality no longer holds. This influences trust accuracy of CATrust. Although CATrust's performance degrades significantly as it uses a linear model to fit a nonlinear data model, it still outperforms Beta Reputation. The result suggests that a nonlinear model be used to improve CATrust performance if the data model is non-linear. This remains to be further investigated. The analysis of linear vs non-linear models will be discussed later in Section 4.5.3.



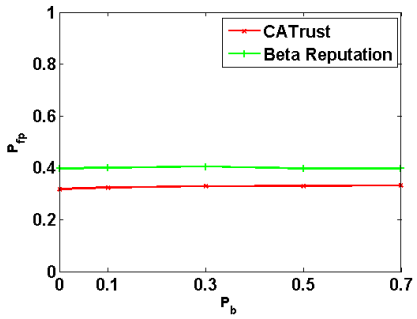
(a) P_{fn} vs. P_b for a malicious trustee SP.



(b) P_{fp} vs. P_b for a malicious trustee SP.



(c) P_{fn} vs. P_b for a non-malicious trustee SP.



(d) P_{fp} vs. P_b for a non-malicious trustee SP.

Figure 4.15: Validation of CATrust Performance using the Synthesized Trace. (a) and (b) are for a Malicious Trustee SP. (c) and (d) are for a Non-malicious Trustee SP.

4.5 Discussion

4.5.1 Computation Feasibility

In this subsection, we discuss the computational feasibility for a SOANET node to execute CATrust to learn the behavior patterns of other nodes ($\underline{\beta_j}$'s for individual SPs)

at runtime. Based on our trust propagation and aggregation protocol design, an SR stores a new service record of length $n_c + 1$ (for n_c context variables and user satisfaction) toward an SP after having a direct service experience with the SP. Two nodes encountering each other exchange their past service records toward all other nodes in the system. The memory complexity per node is $O(n_n n_c n_r)$, where n_n is the number of nodes, n_c is the number of context variables, and n_r is the number of service records (as defined in Table 4.2), because every node needs to store n_r service records (each of size n_c) for each of the other $n_n - 1$ nodes. The communication cost complexity per node is $O(n_n n_c \alpha L)$ where α is the encountering rate and L is the length of the measurement interval, because every node potentially can provide $n_n - 2$ service recommendation records (each of size n_c) toward the other $n_n - 2$ nodes whenever it encounters another node. Last, the computational complexity is $O(n_n k \max(n_c, n_r))$, where k is the number of iterations needed for reaching convergence, because every node needs to update n_r latent variables corresponding to the $n_r$service records and $\underline{\beta_j}$ of size n_c for node j in each iteration and this computational procedure is applied to each of the other $n_n - 1$ nodes in the system. In general, $n_c \ll n_r$ so the computational complexity is $O(k n_n n_r)$. Further, the magnitude of k largely depends on the granularity of context variable values, e.g., a range of (high, medium, low) for energy is of low granularity, while a range of $[0 - 10]$ joule is of high granularity. By controlling data granularity, k is a small constant relative to n_n or n_r, so in practice the computational complexity of CATrust is just $O(n_n n_r)$.

As a comparison, the memory complexity, message complexity, and computational complexity for both Beta Reputation and Adaptive Trust are $O(C n_n)$, $O(\alpha L C n_n)$, and $O(n_n n_r)$, respectively, where $C = 2$ for Beta Reputation (2 positive/negative service counts) and $C = 5$ for Adaptive Trust (2 positive/negative service counts and 3 social similarity lists). We first observe that CATrust has the same order of computational complexity $O(n_n n_r)$ as Beta Reputation and Adaptive Trust. With $n_c \approx C$ (that is, the number of context variables is between 2 to 5), CATrust, Beta Reputation and Adaptive Trust have comparable communication overhead. Last, CATrust has a higher memory overhead by a factor of n_r. In practice, the memory overhead is lower because one may be interested in only the most recent n_r service records (e.g., in the past hour or day). This memory requirement can still be excessive for SOANET nodes with limited memory space. We refer the readers to a caching design [31] as a possible solution to mitigate this problem. For the experimental setting specified in Table 4.2, a node with a 2.4 GHz i7 CPU with 8GB RAM took 2.63s real time to learn an SP's behavior pattern and predict its trust. For a less powerful node, it may take minutes rather than seconds to compute the result. Fortunately, the computational procedure needs to be executed

only periodically in the background by an SR after new observations are collected. Before the next trust update time arrives, an SR can simply use learned behavior patterns ($\underline{\beta}_j$'s for individual SPs) for decision making.

4.5.2 Dealing with Conflicting Behavior and Random Attacks

In this subsection, we discuss the applicability of CATrust in environments with conflicting behavior, random attacks, and opportunistic attacks.

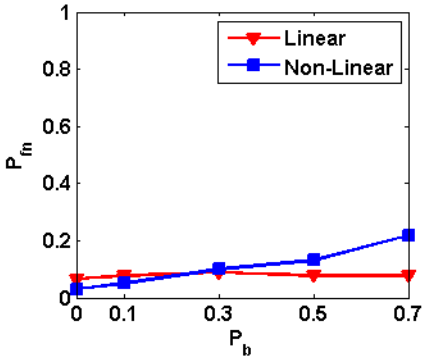
With conflicting behavior attacks, a malicious SP can selectively provide satisfactory service for some SRs while providing unsatisfactory service for others. In general, the relationship between an SR and an SP determines the SP's service behavior toward the SR. This is naturally solved by CATrust since it is based on SR-SP pairing. More specifically, if SP j who is capable of providing good service in a context environment provides bad service to SR i, then SR i will consider SP j's bad service as SP j's service behavior in this context environment. In effect, from the perspective of SR i, SP j's s_g (ground truth service satisfaction) is changed from 1 (satisfactory service) to 0 (unsatisfactory) which is learned by logit regression. As a result, SR i will predict unsatisfactory service being provided by SP j even though SP j is capable of providing satisfactory service in the same context environment.

With random attacks, a malicious node will provide bad service only randomly so as not to risk itself being labeled as a node providing bad service and not being selected for service. Again random attack can be naturally covered by CATrust since it is based on SR-SP pairing. From the perspective of SR i who is under random attacks by SP j, SP j's s_g (ground truth service satisfaction) is sometimes 1 (satisfactory service) and sometimes 0 (unsatisfactory), which is learned by logit regression. As a result, SR i will predict sometimes satisfactory service and sometimes unsatisfactory service being provided by SP j even though SP j is capable of providing satisfactory service in the same context environment. Consequently, the degree to which the malicious node can disguise itself as an SP providing good service is simply proportional to 1 − random attack probability. As long as SP j's random attack probability is not zero, the random attack behavior will be learned by SR i.

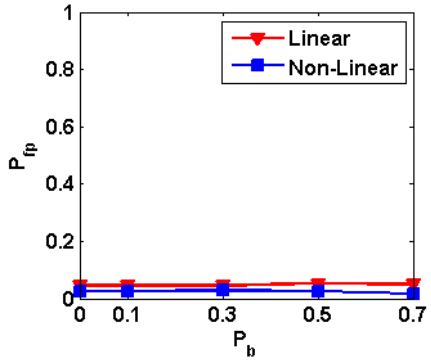
With opportunistic service attacks, a malicious node may not perform persistent attacks all the time but rather can attack opportunistically. To cope with opportunistic service attacks, the condition under which an opportunistic service attack or a time-varying attack will perform may be represented as a context-service quality relationship for CATrust to learn dynamically. Such opportunistic attack thus can be remembered by CATrust when the matching context environment appears again.

4.5.3 Linear Model vs. Non-linear Model Comparison

In many applications, context variables may not be independent, including covariate relationship between service observations and correlation between context variables. The results which we have reported above are based on a simple linear model with computational complexity of $O(n_n n_r)$ (see Section 4.5.1) to model the relation between context variables and observations. In this subsection, we conduct a comparative analysis to test if the prediction accuracy may improve further with a non-linear model at the expense of added computational complexity. The non-linear model implemented is the multi-layer feedforward neural network (FNN) with n_c nodes in the input layer and n_c^2 nodes in the hidden layer, resulting in computational complexity of $O(n_n n_r n_c^3)$ to process n_r service records for each of the other $n_n - 1$ nodes.

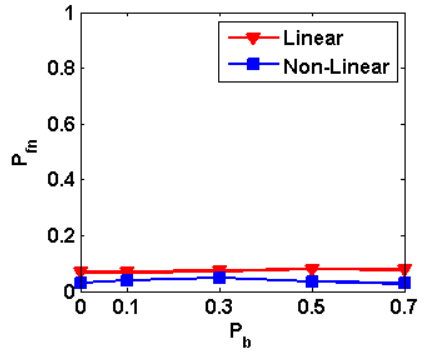


(a) P_{fn} vs. P_b for a non-malicious trustee SP.

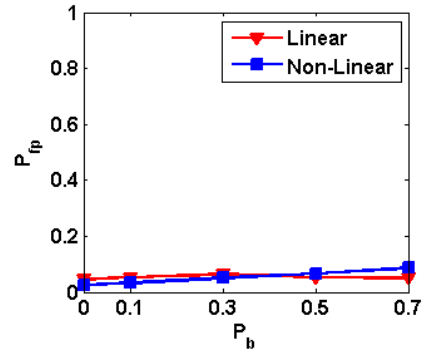


(b) P_{fp} vs. P_b for a non-malicious trustee SP.

Figure 4.16: Performance Comparison of Linear vs. Non-Linear CATrust for a Malicious Trustee SP.



(a) P_{fn} vs. P_b for a non-malicious trustee SP.



(b) P_{fp} vs. P_b for a non-malicious trustee SP.

Figure 4.17: Performance Comparison of Linear vs. Non-Linear CATrust for a Non-Malicoius Trustee SP.

Figure 4.16 and Figure 4.17 compare linear vs. non-linear model performance in terms of converged P_{fn}/P_{fp} values for a malicious trustee SP and a non-malicious trustee SP, respectively, with $P_{ssr} = 60\%$, as P_b varies in the range [0, 70%]. Note that the setup is the same as in Figure 4.12 and Figure 4.13. Also note that regardless of node type (malicious or non-malicious), the probabilities of misidentifying a node's bad service and good service are measured by P_{fn} and P_{fp}, respectively. As shown in Figure 4.16 (for a malicious trustee SP) as P_b increases, linear CATrust performs better than non-linear CATrust in P_{fn}, while non-linear CATrust performs better than linear CATrust in P_{fp}. Because the trustee SP is malicious in this case, P_{fn} (the probability of the SP's bad service being missed) increases as P_b increases via ballot-stuffing attacks. We observe that non-linear CATrust is less resilient to ballot-stuffing attacks than linear CATrust. The reason is that FNN uses mean square error as the objective function known to be sensitive to contaminated data [53]. On the other hand in Figure 4.17 (for a non-malicious trustee SP) as P_b increases, non-linear CATrust performs better than linear CATrust in P_{fn}, while linear CATrust performs better than non-linear CATrust in P_{fp}. Because the trustee SP is non-malicious in this case, P_{fp} (the probability of the SP's good service being missed) increases as P_b increases via bad-mouthing attacks. We again observe that non-linear CATrust is less resilient to bad-mouthing attacks than linear CATrust. On the whole there is a virtual tie between the linear and non-linear models. However, the much higher P_{fn} for a malicious trustee SP as P_b increases (see Figure 4.16 (a)) and the much higher computation complexity make the non-linear model an undesirable choice for runtime execution.

4.6 Summary

In this chapter, we proposed a novel regression-based trust model, CATrust, for evaluating service trust in service-oriented ad hoc networks. CATrust assesses each SP in terms of its service behavior patterns in response to context environment changes. The net effect is that we are able to learn and then predict its service behavior in a particular context environment, instead of judging its service trust from satisfactory/unsatisfactory service history across all context environments. We also built a novel threshold-based recommendation filtering mechanism to effectively filter out false recommendations. A salient feature of our model is that it can accommodate all context environment variables deemed critical to an SP's service behavior.

We analyzed convergence, accuracy and resiliency properties of CATrust and validated the theory via simulation. We conducted sensitivity analysis of CATrust performance with respect to key design parameters. We also conducted a comparative

analysis of CATrust with Beta Reputation with belief discounting and Adaptive Trust with collaborative filtering. Our results validated by simulation demonstrate that CATrust outperforms these existing approaches in both the missing good service and misidentifying bad service probabilities. Finally, we discussed applicability of our CATrust model in terms of computational efficiency, dealing with conflicting behavior attacks and random attacks, performance characteristics of CATrust implemented with the linear model vs. the non-linear model.

Chapter 5

Trust-based Service Composition and Binding with Multiple Objective Optimization in Service-Oriented Mobile Ad Hoc Networks

In this chapter, we validate the concept of trust-based service management with a service composition and binding application with MOO requirements in service-oriented MANETs. The MOO problem is an SP-to-service assignment problem. We propose a trust-based algorithm to solve the problem. We carry out an extensive suite of simulations to test the relative performance of a single-trust protocol, two multi-trust protocols, and CATrust (developed in Chapter 4), as the underlying trust protocol for executing the proposed trust-based algorithm, against a non-trust-based counterpart. Our proposed trust-based algorithm effectively filters out malicious nodes exhibiting various attack behaviors by penalizing them with trust loss, which ultimately leads to high user satisfaction. Further, our proposed trust-based algorithm is efficient with linear runtime complexity while achieving a close-to-optimal solution. In particular, CATrust outperforms single-trust and multi-trust protocols because CATrust considers the association between context and MOO treated as the service quality metric for the service composition and binding MANET application.

5.1 Service Composition and Binding

We consider a service-oriented MANET environment with $|\mathcal{N}|$ nodes moving according to the SWIM mobility model [77]. It introduces dynamic topology and affects the reliability of packet routing over multiple hops from a source to a destination. In particular, it affects the success probability of recommendation packet delivery which in turn affects trust accuracy. To conserve resources, we assume that only a single copy of the recommendation about a target node (node j) is transmitted from the recommender node (node k) to the trustor (node i). Then, the recommendation packet from node k is

lost when there is no route to reach node i from any intermediate node because of topology changes, when there is a channel error with probability $p_{e,}$ or when any intermediate node maliciously performs packet dropping attacks.

5.1.1 Service Quality Criteria

Without loss of generality, we consider three service quality criteria: QoI, service delay (as a QoS attribute), and cost. We denote them by Q, D, and C which may be measured after service invocations are performed. While D and C are easily measureable physical quantities, Q is specific to the application domain. For example, in environment monitoring service, Q is measured by the extent to which the output contributes to the ground truth data [127]. In sensing service, Q is measured by the extent to which the sensing data contributes to the ground truth picture.

We first scale our service quality metrics, Q, D, and C , to the range [0, 1] so that the higher the value, the better the quality [158], as follows:

$$\bar{Q} = \frac{Q - Q_{min}}{Q_{max} - Q_{min}}; \tag{5.1}$$

$$\bar{D} = \frac{D_{max} - D}{D_{max} - D_{min}};\ \bar{C} = \frac{C_{max} - C}{C_{max} - C_{min}}$$

Here Q_{max} and Q_{min} , D_{max} and D_{min} , and C_{max} and C_{min} are the maximum and minimum possible values of Q, D, and C, respectively,. They are known a priori. With this normalization we transform MOO into multi-objective maximization, i.e., from maximizing Q and minimizing D and C, into maximizing $\bar{Q}$, $\bar{D}$ and $\bar{C}$. From a pragmatic perspective, scaling facilitates a fair quantitative comparison of different service quality criteria, as each service quality criterion is in the range of [0, 1] with a higher value representing a higher service quality.

5.1.2 Service Advertisement

A node as an SP advertises its service availability when a peer node (i.e., an SR) shows interest [71] [73]. An SP responds with an advertisement message only if it is capable of providing the requested services. Specifically, it responds with an advertisement message $\mathrm{Ad_{SP}}$ comprising four-tuple records, one for each abstract service S_k it can provide, as follows:

$$\mathrm{Ad_{SP}}\text{: } [k, Q_k, D_k, C_k] \text{ for } S_k \tag{5.2}$$

Here k indicates the index of the service S_k; Q_k the level of QoI the SP can provide; D_k the level of service delay (for QoS); and C_k the service cost. However, a malicious node can promote its importance by advertising false service quality information in Q, D and

C, so as to increase its chance to be selected as the SP, but then performs conflicting attack.

5.1.3 Dynamic Service Composition

For convenience, we use m to index service requests, k to index services, and i, j, or r to index nodes. We also use the notation O_m to refer to service request m, and the notation SR_m to refer to the SR who issues m. A service request (e.g., take me to a nice Thai restaurant nearby with drunken noodle on its menu) requires a number of abstract services S_k's (e.g., transportation service, food service, etc.). For each service request in hand, the SR broadcasts the set of abstract services needed to which all qualified SPs respond with the 4-tuple records in (5.2). Based on the responses received, the SR then constructs a service composition specification (SCS) to specify the service plan for satisfying the service request. An example SCS is:

$$\mathrm{SCS}_m = \langle [S_0], [S_2, S_4], [S_3], [S_7], [S_4, S_8], [S_2] \rangle \tag{5.3}$$

where $[S_2, S_4]$ specifies that S_2 and S_4 are to be executed concurrently; $[S_3], [S_7]$ specifies that S_3 and S_7 are to be executed sequentially. The user also specifies a minimum service quality requirement at the service request level and at the abstract service level as follows:

$$\begin{aligned} \mathrm{SCS}_m^{THRES} &= (Q_m^{THRES}, D_m^{THRES}, C_m^{THRES}) \\ S_k^{THRES} &= (Q_k^{THRES}, D_k^{THRES}, C_k^{THRES}) \end{aligned} \tag{5.4}$$

The SR then decides the best SPs among all responders to execute SCS_m, while meeting the minimum service quality levels at both the service request level and the abstract service level. If the minimum service quality constraint is not satisfied, then it means that there are not enough qualified SPs available to provide service and O_m is considered a failure.

5.1.4 Service Binding

An SP capable of providing multiple abstract services can be selected to execute multiple abstract services in a service request. However, to avoid schedule conflicts among concurrent service requests (issued by multiple SRs) and to avoid degrading an SP's service quality due to heavy workloads, the SP can only commit to one service request. That is, the SP can only participate in one service request at a time to ensure its availability and commitment to a single service request.

5.2 Problem Definition and Performance Metrics

5.2.1 Problem Definition

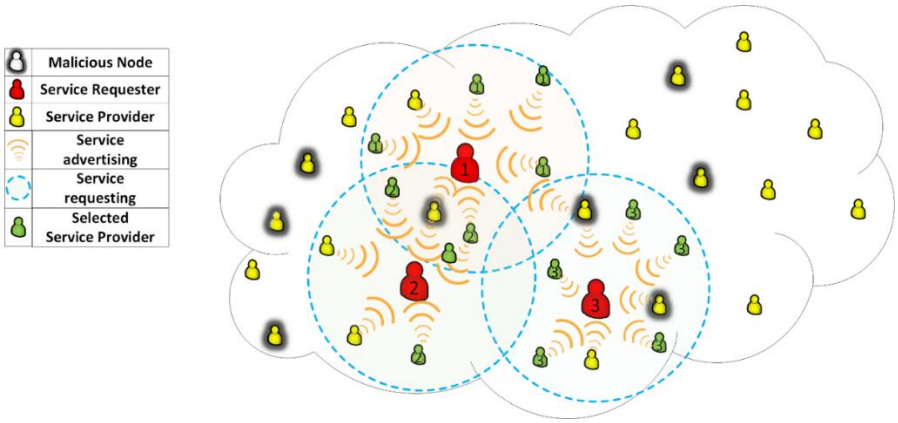


Figure 5.1: Service Requesting, Advertisement and Dynamic Binding.

Figure 5.1 above illustrates the service requesting, advertisement, and dynamic binding process. Given that multiple SPs may meet the service threshold criteria in (5.4), an SR must choose SPs so as to maximize the aggregate $\bar{Q}$, $\bar{D}$ and $\bar{C}$. An SCS for serving a service request is essentially a flow structure consisting of series or parallel substructures. For the SCS in (5.3), there is one series structure consisting of 6 substructures, $[S_0], [S_2, S_4], [S_3], [S_7], [S_4, S_8]$, and $[S_2]$, at the top level, and there are two parallel substructures,$[S_2, S_4]$ and $[S_4, S_8]$, at the bottom level. Let $\bar{Q}_m$, $\bar{D}_m$ and $\bar{C}_m$ be the scaled Q, D and C scores of O_m, and $\bar{Q}_{m,S}$, $\bar{D}_{m,S}$ and $\bar{C}_{m,S}$ be the scaled Q, D and C scores of substructure S. The service quality of O_m measured by $\bar{Q}_m$, $\bar{D}_m$ and $\bar{C}_m$ (the larger the better) after service binding can be computed recursively as follows:

a) For a parallel structure S consisting of two concurrent substructures $[S_1, S_2]$, the maximum $\bar{Q}$ and $\bar{D}$ scores are limited by the minimum service quality score (which we want to avoid through node selection), and the maximum $\bar{C}$ score is bounded by the sum of $\bar{C}$ scores (since cost is additive), i.e.,

$$\begin{aligned} \bar{Q}_{m,S} &= min\left(\bar{Q}_{m,S_1}, \bar{Q}_{m,S_2}\right); \\ \bar{D}_{m,S} &= min\left(\bar{D}_{m,S_1}, \bar{D}_{m,S_2}\right); \\ \bar{C}_{m,S} &= \bar{C}_{m,S_1} + \bar{C}_{m,S_2}. \end{aligned} \tag{5.5}$$

When combining scaled Q or D scores of two concurrent substructures using the min operator, the minimum of scaled Q scores turns out to be the scaled minimum of the unscaled Q scores. That is, $min\left(\bar{Q}_{m,S_1}, \bar{Q}_{m,S_2}\right) = min\left(\frac{Q_{m,S_1} - Q_{min}}{Q_{max} - Q_{min}}, \frac{Q_{m,S_2} - Q_{min}}{Q_{max} - Q_{min}}\right) = \frac{min(Q_{m,S_1}, Q_{m,S_2}) - Q_{min}}{Q_{max} - Q_{min}}$. The combined scaled Q score stays in the range of [0, 1], if each substructure is a single abstract service at the bottom level of an SCS. When combining scaled C scores of two concurrent substructures, we use the addition

operator because each substructure will unavoidably incur a separate service cost which must be accounted for. The combined C score as a result of using the addition operator is no longer scaled in [0, 1].

b) For a series structure S consisting of two sequential substructures $[S_1], [S_2]$, the maximum score is limited by the sum of service quality scores, i.e.,

$$\begin{aligned} \bar{Q}_{m,S} &= \bar{Q}_{m,S_1} + \bar{Q}_{m,S_2}; \\ \bar{D}_{m,S} &= \bar{D}_{m,S_1} + \bar{D}_{m,S_2}; \\ \bar{C}_{m,S} &= \bar{C}_{m,S_1} + \bar{C}_{m,S_2}. \end{aligned} \tag{5.6}$$

When combining scaled Q, D and C scores of two sequential substructures, we use the addition operator because the two substructures will be sequentially executed and each score (which we want to maximize through node selection) must be separately accounted for. The combined score as a result of using the addition operator is no longer scaled in [0, 1].

Here we note that at the bottom level of an SCS, a substructure is only an abstract service. If node j is selected to bind to this abstract service then $\bar{Q}_{m,S} = \bar{Q}_{m,j}$, $\bar{D}_{m,S} = \bar{D}_{m,j}$ and $\bar{C}_{m,S} = \bar{C}_{m,j}$. The top level, on the other hand, is either a series substructure or a parallel substructure. Let θ be the top level substructure of this SCS for O_m. Then, the overall service quality score of O_m (after service binding) is given by:

$$\bar{Q}_m = \bar{Q}_{m,\theta};\ \bar{D}_m = \bar{D}_{m,\theta};\ \bar{C}_m = \bar{C}_{m,\theta} \tag{5.7}$$

5.2.2 MOO Problem Formulation

We use the weighted sum form [112] allowing a user to express its preferences regarding service quality criteria. Let $\omega_{Q,m}$, $\omega_{D,m}$ and $\omega_{C,m}$ be the weights associated with $\bar{Q}_m$, $\bar{D}_m$ and $\bar{C}_m$ for O_m issued by the user, with $\omega_{Q,m} + \omega_{D,m} + \omega_{C,m} = 1$. Another compelling justification of using weighted sum is that expressing the optimization criterion of a multi-objective problem by means of a weighted sum corresponds to a Lagrangian formulation [23] with multiple Lagrange multipliers, thereby effectively sweeping the lower convex envelope of the objective surface. With this simple additive weighting technique, we formulate our MOO problem at the service-request level as:

$$\text{Maximize } \text{MOO}_m = \omega_{Q,m}\bar{Q}_m + \omega_{D,m}\bar{D}_m + \omega_{C,m}\bar{C}_m \tag{5.8}$$

subject to the service request level constraint $\text{SCS}_m^{\text{THRES}}$ and the abstract service level constraint S_k^{THRES} specified in (5.4) by the user. As there may be multiple SRs issuing service requests and performing service composition and binding concurrently, we formulate our MOO problem at the system level as:

$$\text{Maximize MOO} = \sum_{m \in \mathcal{T}} (\omega_{Q,m}\bar{Q}_m + \omega_{D,m}\bar{D}_m + \omega_{C,m}\bar{C}_m) \tag{5.9}$$

where $\mathcal{T}$ is the set of concurrent service requests issued by multiple SRs who are competing for the use of SPs available to them. It is noteworthy that (5.8) and (5.9) solve the service binding problem, given a service composition specification (SCS) formulated as in (5.3).

5.2.3 MOO Value and User Satisfaction as Performance Metrics

While the final MOO value defined in (5.9) above can be used to measure MOO performance, user satisfaction ultimately determines if a service request is a success or a failure. The user satisfaction level of the SR toward SPs selected for executing O_m, denoted as US_m, can be measured by the ratio of the actual service quality received to the best service quality available among SPs for executing O_m. We allow a user to specify a user satisfaction threshold, denoted as UST_m, which specifies the minimum service quality the user can accept. This is to be compared against US_m to decide if the service experience of the user toward SPs selected for executing O_m is positive or negative. If O_m fails because of failing to satisfy the service request level constraint SCS_m^{THRES}, then US_m is zero. If the service experience is negative, culprit SPs are identified and penalized with trust loss. Conversely, if the service experience is positive, all constituent SPs are rewarded with trust gain based on US_m obtained. For notational convenience, let $\overline{SQ}_m^R = \omega_{Q,m}\bar{Q}_m^R + \omega_{D,m}\bar{D}_m^R + \omega_{C,m}\bar{C}_m^R$ denoting the actual service quality received after service binding and execution of O_m, $\overline{SQ}_m^{max} = \omega_{Q,m}\bar{Q}_m^{max} + \omega_{D,m}\bar{D}_m^{max} + \omega_{C,m}\bar{C}_m^{max}$ denoting the best service quality that can ever be achieved, and $\overline{SQ}_m^{min} = \omega_{Q,m}\bar{Q}_m^{THRES} + \omega_{D,m}\bar{D}_m^{THRES} + \omega_{C,m}\bar{C}_m^{THRES}$ denoting the minimum service quality that must be obtained in order to satisfy the service request level constraint $\text{SCS}_m^{\text{THRES}}$. Then, with score scaling, US_m can be computed as:

$$US_m = \begin{cases} \dfrac{\overline{SQ}_m^R - \overline{SQ}_m^{min}}{\overline{SQ}_m^{max} - \overline{SQ}_m^{min}} & \text{if } \overline{SQ}_m^R \geq \overline{SQ}_m^{min} \\ 0 & \text{otherwise} \end{cases} \tag{5.10}$$

Here $\overline{SQ}_m^R, \overline{SQ}_m^{max}$ and $\overline{SQ}_m^{min}$ are the received, maximum, and minimum service quality scores, respectively, for executing O_m, calculated based on $\bar{Q}_m$, $\bar{D}_m$ and $\bar{C}_m$ in (5.7). The second condition in (5.10) is for the case in which the received service quality is less than the required minimum service quality. Again we note that because of scaling, the large the $\bar{Q}$, $\bar{D}$ and $\bar{C}$ values, the better the service quality. Also note that $\overline{SQ}_m^R = \omega_{Q,m}\bar{Q}_m^R + \omega_{D,m}\bar{D}_m^R + \omega_{C,m}\bar{C}_m^R$, so maximizing MOO_m in (5.8) is equivalent to maximizing US_m in (5.10). Therefore, the MOO problem to solve is in effect a user satisfaction maximization problem.

5.3 Trust Management Protocol

In this section, we first discuss a well-known trust management scheme based on Beta Reputation System (BRS) [55] [69] as the baseline scheme against which our multi-trust protocol will be compared. We choose BRS because of its sound statistical basis compared to other schemes using intuitive and ad-hoc methods for measuring trust. In addition, it enables a trustor to ensure tractability of trust evidence over time. We note that a trust model based on Dirichlet distribution [68] [92], which is a generalization of BRS, can also be used as the single-trust baseline scheme for performance comparison. However, since a user can specify a minimum *user satisfaction threshold* to decide if a service experience is positive or negative (a binary classification), BRS suffices. We then describe two trust protocols (TRM and SRM described below) with multi-trust design. Finally we describe how one can apply CATrust developed in Chapter 4 to solving the service composition and binding MOO problem.

5.3.1 Single-trust Protocol Design

The baseline BRS protocol is based on Bayesian inference with the trust value modeled as a random variable in the range of [0, 1] following the Beta(α, β) distribution; the numbers of positive and negative experiences are modeled as binomial random variables. Since the beta-binomial is a conjugate pair, this leads to a posterior beta distribution with updated parameters. Here $\alpha/(\alpha + \beta)$ is the estimated mean of "direct" trust evidence of an SP where α is the number of positive interactions and β is the number of negative interactions. A positive evidence is observed when SR_m is satisfied. More specifically, when US_m exceeds UST_m, it is counted as positive evidence and all constituting SPs in O_m are rewarded. In the case of positive evidence, α is incremented by 1 for all SPs in O_m. On the other hand, when US_m is less than UST_m, SR_m identifies the culprits with low performance (i.e., the actual service quality is lower than the advertised service quality) and considers it as negative evidence against these culprits. In this case, β is increased by 1 for all identified culprits. SPs with expected performance (i.e., the actual service quality is about the same as the advertised service quality) are identified as benign and will not be penalized.

After a service request is completed, the SR propagates its updated trust of the SPs involved in the service request to other nodes in the system. A node receiving a trust update follows the propagation and aggregation protocol described below to update its (α, β) pair toward the SP. Trust propagation is done through recommendations received from 1-hop neighbors whom the trustor encounters dynamically. A node (trustor) will select n_{rec} recommenders whom it trusts most to provide trust

recommendations of an SP (trustee). A recommender should only pass its direct interaction experience with the trustee node in terms of (α, β) as a recommendation to avoid dependence and looping [70]. Let node i be the trustor, node j be the trustee, and node k be a recommender. Also let $(\alpha_{i,j}, \beta_{i,j})$ be the trustor's (α, β) toward the trustee, $(\alpha_{k,j}, \beta_{k,j})$ be the recommender's (α, β) toward the trustee and $(\alpha_{i,k}, \beta_{i,k})$ be the trustor's (α, β) toward the recommender. Based on belief discounting (see [69] for details), node i will compute its new $(\alpha_{i,j}^{new}, \beta_{i,j}^{new})$ as follows:

$$\alpha_{i,j}^{new} = \alpha_{i,j} + \frac{2\alpha_{i,k}\alpha_{k,j}}{[(\beta_{i,k}+2)(\alpha_{k,j}+\beta_{k,j}+2)]+2\alpha_{i,k}} \tag{5.11}$$

$$\beta_{i,j}^{new} = \beta_{i,j} + \frac{2\alpha_{i,k}\beta_{k,j}}{[(\beta_{i,k}+2)(\alpha_{k,j}+\beta_{k,j}+2)]+2\alpha_{i,k}} \tag{5.12}$$

The basic idea is that if node i does not trust k, it will discount the recommendation provided by node k, so $\alpha_{i,j}^{new} \sim \alpha_{i,j}$ and $\beta_{i,j}^{new} \sim \beta_{i,j}$ as if the recommendation from k does not have any effect. This can be derived from (5.11) and (5.12). First of all, if node i does not trust node k then $\alpha_{i,k} \ll \beta_{i,k}$. In case node k is performing a bad-mouthing attack on node j, then $\alpha_{k,j} \ll \beta_{k,j}$. Applying these two conditions to (5.11) and (5.12), one can easily verify $\alpha_{i,j}^{new} \sim \alpha_{i,j}$ and $\beta_{i,j}^{new} \sim \beta_{i,j}$. In case node k is performing a ballot-stuffing attack on node j, then $\alpha_{k,j} \gg \beta_{k,j}$ and again one can easily verify $\alpha_{i,j}^{new} \sim \alpha_{i,j}$ and $\beta_{i,j}^{new} \sim \beta_{i,j}$. After trust aggregation, the trustor's (or node i's) trust toward the trustee (or node j) is then computed as $T_{i,j} = \frac{\alpha_{i,j}^{new}}{\alpha_{i,j}^{new}+\beta_{i,j}^{new}}$.

5.3.2 Multi-trust Protocol Design

Our trust management protocol design centers on the concept of multi-trust. Multi-trust refers to the use of multidimensional trust for more accurately describing multiple and often distinct factors contributing to successful service execution. For service composition and binding, we choose two unique trust dimensions: competence and integrity. We chose these two dimensions because the capability for service provision (i.e., competence) and the compliance to the prescribed service protocol (i.e., integrity) are the key criteria of quality service provision for high user satisfaction. The two trust dimensions are:

- **Competence**: This refers to an SP's capability to satisfactorily serve the received request. This is largely determined by the intrinsic service capability of the SP, as modeled by the SP's "true" Q, D and C scores.

- **Integrity**: This refers to the degree to which a node complies with the prescribed protocol, including the service contract and the trust protocol. A node may violate its service contract when it performs self-promotion attacks. That is, a node lies about Q, D and C scores of its own capability so as to increase its chance to be included in executing service requests but then it fails to honor the service contract or just performs opportunistic service once it is selected for service request execution.

We denote node i's trust toward node j in X (i.e., C for competence, and I for integrity) as $T_{i,j}^{X}$. We adopt BRS to assess node i's mean direct trust toward node j in trust property X as $\alpha_{i,j}^{X}/(\alpha_{i,j}^{X}+\beta_{i,j}^{X})$ where $\alpha_{i,j}^{X}$ is the number of positive and $\beta_{i,j}^{X}$ is the number of negative experiences in trust property X, which are accumulated upon trust update. To update $(\alpha_{i,j}^{C}, \beta_{i,j}^{C})$ for competence trust, node i (acting as the SR) compares UST_m with US_m as described in the baseline BRS scheme. To update $(\alpha_{i,j}^{I}, \beta_{i,j}^{I})$ for integrity trust, node i considers it positive evidence if it sees node j's observed Q, D and C scores are close to node j's advertised scaled Q, D and C scores. Node i (as SR_m) assesses node j's compliance degree ($CD_{m,j}$) as:

$$CD_{m,j} = min\left(\frac{\bar{Q}_{m,j}^{observed}}{\bar{Q}_{m,j}^{advertised}}, \frac{\bar{D}_{m,j}^{observed}}{\bar{D}_{m,j}^{advertised}}, \frac{\bar{C}_{m,j}^{observed}}{\bar{C}_{m,j}^{advertised}}\right) \tag{5.13}$$

Here $\bar{Q}_{m,j}^{advertised}, \bar{D}_{m,j}^{advertised}$ and $\bar{C}_{m,j}^{advertised}$ are node j's advertised "scaled" Q, D and C scores, while $\bar{Q}_{m,j}^{observed}, \bar{D}_{m,j}^{observed}$ and $\bar{C}_{m,j}^{observed}$ are node j's "scaled" Q, D and C scores actually observed by node i (acting as SR_m) during O_m execution. Each user defines its minimum compliance degree threshold for service request m, denoted by CDT_m. If $CD_{m,j} \geq CDT_m$, then it is counted as a positive experience for node j for integrity and $\alpha_{i,j}^{I}$ is incremented by 1; otherwise, it is counted as a negative experience for node j and $\beta_{i,j}^{I}$ is incremented by 1. We note that this can effectively capture self-promotion attack behavior. Trust propagation and aggregation are again based on the concept of belief discounting, i.e., the $(\alpha_{i,j}^{X}, \beta_{i,j}^{X})$ pair of node i toward node j is merged with n_{rec} pairs of $(\alpha_{r,j}^{X}, \beta_{r,j}^{X})$ from n_{rec} recommenders whom node i trusts the most.

There are multiple ways to form the overall trust $T_{i,j}$ from $T_{i,j}^{C}$ and $T_{i,j}^{I}$. In this work, we explore the Trust + Confidence formation model by which integrity trust is used as confidence to assess the validity of competence trust assuming that competence trust ultimately ensures service success. If integrity trust falls below a threshold, competence trust is invalid or scaled down. We investigate two relationships under this model: drop-to-zero and scaling. In the threshold-based relationship model (TRM), if integrity trust falls below a threshold, $T_{i,j}^{I,THRES}$, competence trust drops to zero. In the scaling relationship model (SRM), competence trust scales up (to 1 maximum) or down (to 0

minimum), depending on whether integrity trust is higher or lower than the threshold. The integrity threshold $T_{i,j}^{I,THRES}$ may be individual-based (for node i toward node j) in service-oriented MANETs populated with human operators. More specifically, TRM computes the overall trust as:

$$T_{i,j} = T_{i,j}^{C} \text{ if } \quad T_{i,j}^{I} \geq T_{i,j}^{I,THRES}; \text{ 0 otherwise} \tag{5.14}$$

where $T_{i,j}^{I,THRES}$ is the minimum integrity trust threshold.

SRM computes the overall trust as:

$$T_{i,j} = min\left(1, T_{i,j}^{C} \times \frac{T_{i,j}^{I}}{T_{i,j}^{I,THRES}}\right) \tag{5.15}$$

5.3.3 CATrust Design

The basic idea of CATrust discussed in Chapter 4 is for SR i to predict whether SP j will perform satisfactorily or not for a requested abstract service in a particular context environment, given a history of evidence. The objective is to achieve high prediction accuracy in terms of correctly predicting bad service while not missing good service from an SP.

Within a specific type of service, SR i's observation s_{ij}^{t} at time t of the service quality received from SP j is either "satisfactory" or "unsatisfactory." If the service quality is satisfactory, then the service assessment $s_{ij}^{t} = 1$ and SP j is considered *trustworthy* in this context environment; otherwise, $s_{ij}^{t} = 0$ and SP j is considered *untrustworthy*. For the service composition and binding application in hand, if $US_m \geq UST_m$ and $CD_{m,j} \geq CDT_m$, then $s_{ij}^{t} = 1$; otherwise, $s_{ij}^{t} = 0$. Let the operational and environmental conditions at time t be characterized by a set of distinct context variables $\underline{x}^t = [x_0^t, \dots, x_M^t]$. Then, SP j's service trust is the probability that SP j is capable of providing satisfactory service given context variables $\underline{x}^t$, i.e., $T_j^t \triangleq \Pr(s_j^t = 1)$.

Let k ($k \neq i$) be a recommender who had prior service experience with SP j and is asked by SR i to provide its feedback regarding SP j. The recommendation from node k is in the form of $[\underline{x}^t, s_{kj}^t]$ specifying the context $\underline{x}^t$ under which the observation s_{kj}^t was made. Since k might launch recommendation attacks, it might report a false observation to i, in which case s_{kj}^t reported is $1 - s_{kj}^t$. Let $\underline{\tilde{S}}_{ij} = \left[\underline{\tilde{S}}_{ij}^{t_0}, \dots, \underline{\tilde{S}}_{ij}^{t_n}\right]$, $i \neq j$, denote the cumulative evidence gathered by SR i over $[t_0, \dots, t_n]$, including self-observations and recommendations. Also let $\underline{X} = \left[\underline{x}^{t_0}, \dots, \underline{x}^{t_n}\right]$ denote the corresponding context matrix.

To apply CATrust to solving the service composition and binding MOO problem, SR i would apply (4.3) to compute SP j's trust. Since service quality is defined by Q, D and

C, local traffic (which influences Q and D) and incentive (which influences C) are chosen as the context variables. The incentive to SP j is determined by SR i itself so it is easily measurable by SR i. The local traffic can be estimated by SR i based on the location information which determines collision probability or the packet retransmission probability after transmitting a sequence of packets for initiating a service request.

5.4 Service Composition and Binding Algorithm Description

In this section, we describe the three trust-based service composition and binding algorithms (based on CATrust, BRS, TRM, and SRM), and the non-trust-based counterpart. After an SR formulates an SCS (e.g., (5.3)) based on available SPs for executing O_m, multiple solutions may exist to meet the service requirements and constraints. The SR then chooses a solution among all candidate solutions to maximize MOO_m in (5.8). The four algorithms are:

- ***Non-trust-based***: While there is no trust in place, each SR keeps a blacklist of SPs with which it has negative interaction experience, i.e., $\mathrm{CD}_{m,j} < \mathrm{CDT}_m$. When selecting the best node-to-service assignment, it only considers SPs that are not blacklisted.
- ***Trust-based*** (CATrust, BRS, TRM and SRM): As illustrated in Figure 5.2 below, each SR selects the best node-to-service assignment that maximizes MOO_m in (5.8) with $\bar{Q}_{m,j}$, $\bar{D}_{m,j}$ and $\bar{C}_{m,j}$ (of node j at the bottom level of the SCS defined in (5.5) and (5.6)) multiplying by $T_{SR,j}$ which is the SR's overall trust toward node j obtained from running a trust protocol (BRS, TRM or SRM) as discussed in Section 5.5. The basic idea of trust-based service composition and binding is that an SP's advertised Q, D and C scores are discounted by the SR's trust towards the SP.

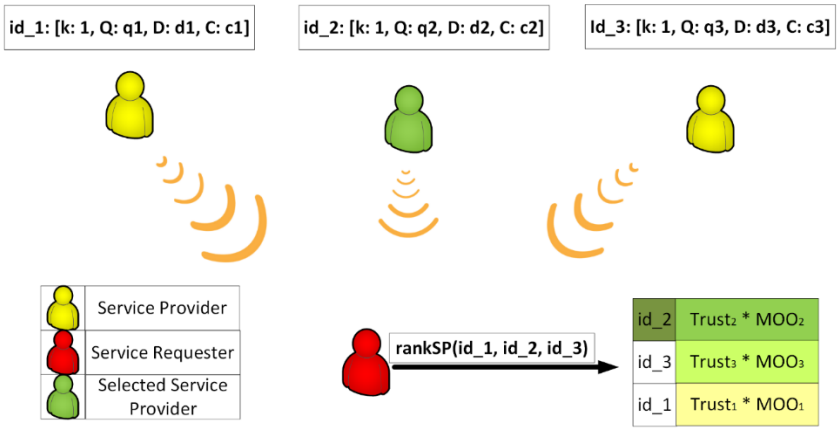


Figure 5.2: Trust-based Service Binding.

Each algorithm described above can be solved by ILP with exponential runtime complexity of $0(2^{|\mathcal{N}|})$ where $|\mathcal{N}|$ is the number of SPs, assuming that the total number of abstract services in a set of concurrent service requests is much smaller than the number of SPs available.

Below we provide implementation details of the ILP solution technique for optimally solving the node-to-service assignment problem for maximizing M00 in (5.9) for both trust-based and non-trust-based algorithms.

Table 5.1 defines the variables used in the ILP formulation. There is only one decision variable, namely, $w_{j,k,m}$ to be determined by the ILP, specifying if node j should be assigned to abstract service k of service request m. The ILP will search for an optimal solution of $w_{j,k,m}$ for all j's, k's and m's to maximize MOO in both trust-based design and non-trust-based design algorithms. The objective function MOO = $\sum_{m\in\mathcal{T}}(\omega_{Q,m}\bar{Q}_m + \omega_{D,m}\bar{D}_m + \omega_{C,m}\bar{C}_m)$ as defined by (5.9) can be computed as a linear function of $w_{j,k,m}$ (the only decision variable to be decided by the ILP).

Table 5.1: Variable Definitions for ILP.

Variable	Definition
$ov_{p,q}$	1 if service requests O_p and O_q are overlapping in time; 0 otherwise
$s_{j,k}$	1 if node j can provide abstract service S_k; 0 otherwise
$tt_{j,k}$	1 if advertised service quality of node j satisfies the abstract service level minimum threshold of S_k; 0 otherwise
$in_{k,m}$	1 if service request O_m requires abstract service S_k; 0, otherwise
$to_{j,k,m}$	$s_{j,k} \times tt_{j,k} \times in_{k,m}$
$w_{j,k,m}$	1 if node j is assigned to service S_k in service request O_m; 0 otherwise

The service-to-node assignment MOO problem is formulated as follows:

Given: $\mathcal{T}, \mathcal{S}_m, \mathcal{N}$

Calculate: $ov_{p,q}, s_{j,k}, tt_{j,k}, in_{k,m}$

Find: $w_{j,k,m}$

Maximize: $\sum_{\mathrm{m}\in\mathcal{T}}(\omega_{Q,m}\bar{Q}_m + \omega_{D,m}\bar{D}_m + \omega_{C,m}\bar{C}_m)$

Subject to: $\forall j\ \forall\{p,q\}\ ov_{p,q} \times (w_{j,k,p} + w_{j,k,q}) \leq 1;$

$$\sum_j w_{j,k,m} = in_{k,m};\ w_{j,k,m} \leq to_{j,k,m}$$

To circumvent high runtime complexity which renders it infeasible for runtime operations, we develop heuristic-based solutions with linear runtime complexity of $O(|\mathcal{N}|)$ for solution efficiency. For all algorithms, an SR simply ranks all eligible SPs for executing an abstract service specified in SCS_m by $\omega_{Q,m}\bar{Q}_m + \omega_{D,m}\bar{D}_m + \omega_{C,m}\bar{C}_m$ and selects the highest ranked SP as the winner for executing that particular abstract service. It examines all SPs which responded to its query in a single round and performs ranking and service binding for all abstract services defined in its SCS_m. Then, the SR notifies SPs that are selected without coordination with other concurrent SRs. As a result, an SP may receive multiple offers from multiple SRs executing concurrent service requests.

In the *non-trust-based* algorithm, an SP receiving multiple offers randomly selects one SR among all to serve. In the trust-based algorithm, an SP resolves the tie-breaker by selecting the SR for which it has the highest trust to ensure the highest success probability as it will increase its chance of gaining good trust. The other SRs not selected will be informed of the decision by the SP and will then select other SPs that are still available to provide the particular abstract service. The time to complete the node-to-service selection thus is linear to the number of eligible SPs multiplied by the number of concurrent service requests because each SR will only examine and rank the advertised service quality scores by all eligible SPs once to select a subset of SPs that maximizes its own ranking.

Here we note that the heuristics employed by non-trust-based and trust-based algorithms for service binding are not necessarily optimal because the resulting solution may not maximize MOO_m in (5.8) or MOO defined in (5.9). As we shall see later, our heuristic design is able to achieve a solution quality approaching that generated by ILP but with significantly lower complexity.

5.5 Results and Analysis

In this section, we evaluate performance of trust-based algorithms (with BRS, TRM or SRM for trust computation) vs. non-trust-based algorithms (with blacklisting).

5.5.1 Experiment Setup

Table 5.2 lists input parameters and their default values for performance analysis. Below we explain each parameter in the table. For the small, medium, and large-sized problems, we consider $|\mathcal{N}| = 60$, 120, and 240 SPs respectively. There are $|\mathcal{S}| = 9$ abstract services, S_0 to S_8, provided by these SPs. For simplicity, each SP is assumed to be specialized to one abstract service S_k which is randomly assigned

initially. All nodes follow the SWIM mobility model with p_e= 0.5%. We consider a scenario with 15 service requests ($|\mathcal{T}| = 15$) divided into 9 sequential chunks, i.e., {1, 2}, {3}, {4, 5}, {6, 7, 8}, {9}, {10}, {11, 12}, {13}, and {14, 15}, where a chunk is defined as a set of concurrent service requests overlapping in execution time. The performance outcome is about the same when the number of chunks is more than 9 or when the number of service requests is longer (30 and 60), so these scenarios are not presented here. For simplicity, each service request has only one SCS consisting of $|\mathcal{S}_m|$ abstract services connected in a series structure, with $|\mathcal{S}_m| = 4$, 8 and 16 for small, medium and large sized problems, respectively. In our case study, we consider only one SCS in each service request. The abstract services are randomly selected from S_0 to S_8 so that the demand to each node is roughly equal in these different sized problems. In case an SR cannot find enough SPs to satisfy the SCS, the service request fails and $US_m = 0$.

Table 5.2: Input Parameters and Default Values/Ranges.

Parameter	Value	Type
$\|\mathcal{T}\|$	15	Input
$\|\mathcal{N}\|$	60, 120, 240	Input
$\|\mathcal{S}_m\|$	4, 8, 16	Input
$\|\mathcal{S}\|$	9	Input
$(Q_k^{THRES}, D_k^{THRES}, C_k^{THRES})$	(1,5,5)	Input
$(Q_m^{THRES}, D_m^{THRES}, C_m^{THRES})$	(4,20,20)	Input
P_{bad}	10-50%	Input
p_e	5%	Input
Q_{bad}	[1-3]	Input
D_{bad}	[3-5]	Input
C_{bad}	[2-5]	Input
Q_{good}	[3-5]	Input
D_{good}	[1-4]	Input
C_{good}	[1-2]	Input
(α, β) for BMA	(1, 10)	Derived
(α, β) for BSA	(10, 1)	Derived
n_{rec}	3	Design
$T_{i,j}^{I,THRES}$	0.5	Design
CDT_m	50-100%	Design
UST_m	50-100%	Design

We model the hostility of the environment by the percentage of malicious nodes, P_{bad}, in the range of $[0 - 50\%]$ with the default value set at 30%. A malicious node

performs all attacks as described in the threat model. In particular, the self-promotion attack behavior is modeled by a risk parameter, P_{risk}, in the range of $[0-100\%]$ with the default set at 50%. A malicious node performs self-promotion attacks by boosting its advertised Q, D and C scores by multiplying its true Q, D and C scores by $(1+P_{risk})$, $(1-P_{risk})$, and $(1-P_{risk})$, respectively. Q_{bad}, D_{bad} and C_{bad} give the true Q, D and C scores for bad nodes, which can be boosted during service advertisement. Q_{good}, D_{good} and C_{good} give the true Q, D and C scores for good nodes, which will be reported by good nodes faithfully during service advertisement. The Q, D and C values of a node are generated from a uniform distribution at the beginning and are not changed during system operation. The default values are set for the case in which the service quality of bad nodes is inferior to that of good nodes to reveal interesting design tradeoffs. Later we will perform a sensitivity analysis of the effect of Q, D and C score distributions for good and bad nodes on performance.

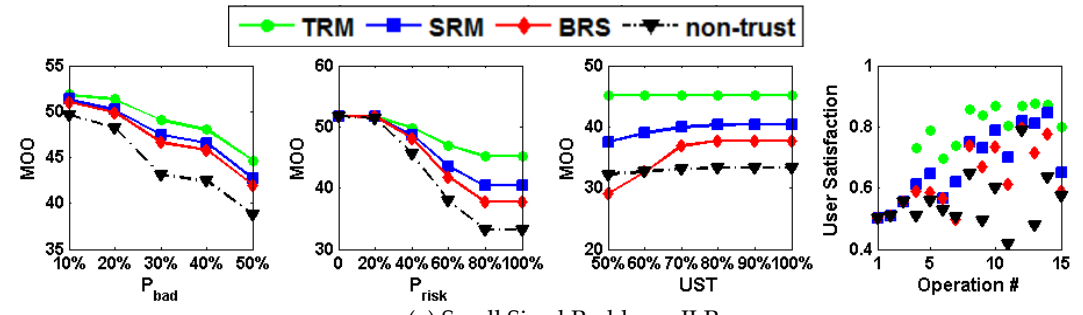
We initially set $(Q_k^{THRES}, D_k^{THRES}, C_k^{THRES})$ = (1, 5, 5) at the abstract service level and $(Q_m^{THRES}, D_m^{THRES}, C_m^{THRES})$= (4, 20, 20) at the service request level for the light minimum service quality constraint case. Later we will perform a sensitivity analysis of the effect of service quality constraints.

The initial trust values (for integrity and competence in the case of multi-trust) are set to 0.5 for all nodes meaning ignorance (no knowledge). The integrity trust threshold $T_{i,j}^{I,THRES}$ for TRM and SRM is set to 0.5 to punish a node with integrity trust less than ignorance trust as time progresses. The protocol compliance degree threshold CDT_m is set to 0.9 to accommodate a 10% maximum detection error for assessing protocol compliance behavior. For simplicity we set $\omega_{Q,m} = \omega_{D,m} = \omega_{C,m} = 1/3$ in (5.8). The number of recommenders n_{rec} is set to 3 so node i will only allow 3 nodes with the highest $T_{i,j}$ values as the recommenders during trust update. When a bad node is mistakenly chosen as a recommender, it can perform a bad-mouthing attack (BMA) on a good trustee node. This is done by providing a very low α, and a very high β, e.g., (α, β) = (1, 10), with α representing the number of positive and β the number of negative experiences, to ruin the trust of the good trustee node. A bad node serving as a recommender can also perform a ballot-stuffing attack (BSA) on a bad trustee node by providing a very high α, and a very low β, e.g., $(\alpha, \beta) = (10, 1)$, to boost the trust of the bad trustee node.

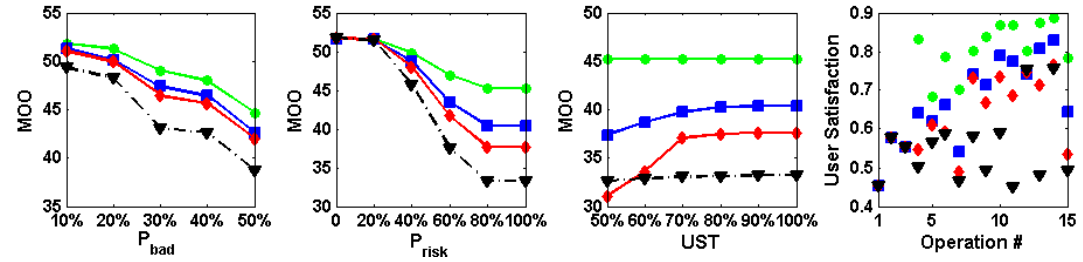
5.5.2 Comparative Performance Analysis

In this section, we compare MOO performance of trust-based and non-trust-based algorithms via MATLAB simulation. Specifically, we simulate non-trust-based and trust-based algorithms (both have linear runtime complexity) under the same

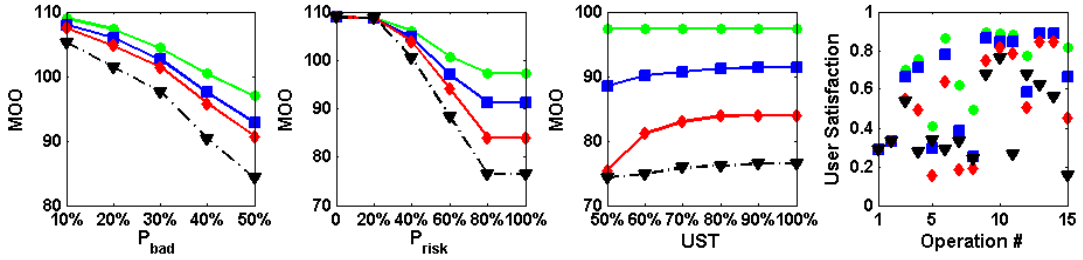
environment setting defined in Table I with $|\mathcal{N}| = 60$, 120 and 240, and correspondingly $|\mathcal{S}_m| = 4$, 8 and 16 for small, medium and large sized problems, respectively.



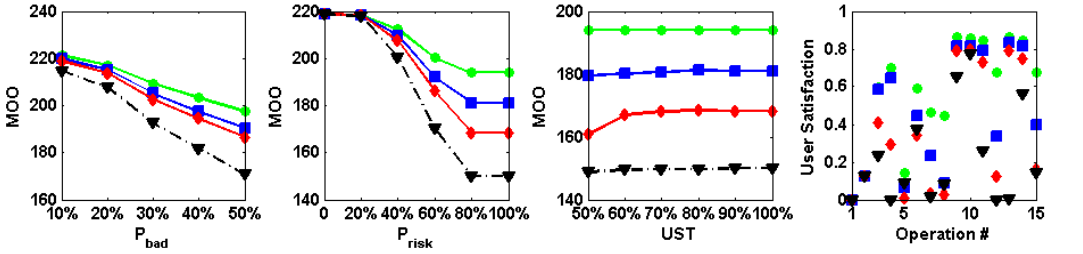
(a) Small Sized Problem - ILP.



(b) Small Sized Problem -Simulation.



(c) Medium Sized Problem - Simulation.



(d) Large Sized Problem -Simulation.

Figure 5.3: Performance Comparison for Small, Medium and Large Sized MOO Problems.

Figure 5.3 (a) shows the ILP results for the small model. Figure 5.3 (b)-(d) show the simulation results for the small, medium and large sized problems, respectively. The simulation results for the small sized problem in Figure 5.3 (b) are to be compared against the ILP analytical results in Figure 5.3 (a) to reveal the tradeoff between solution efficiency gained (because of linear runtime complexity) vs. solution optimality

sacrificed (because of the use of heuristics). Here we note that medium to large sized problems ($|\mathcal{N}| = 120$ and 240) can only be evaluated by simulation since ILP is not able to generate a solution due to the high computational complexity. We observe that for the small-sized problem, the simulation results based on heuristic designs are remarkably similar to the ideal performance results both in shape and value. This demonstrates that the heuristic design can achieve solution efficiency without sacrificing solution optimality too much.

The leftmost two graphs in each row of Figure 5.3 examine the negative impact of increasing P_{bad} and P_{risk} on MOO performance in (9). Compared with the non-trust-based algorithm, trust-based algorithms show high resilience against increased attack intensity with more malicious entities (P_{bad}) or higher self-promotion attack behavior (P_{risk}). TRM has the best MOO performance among all because the drop-to-zero trust based on trust threshold can effectively filter out bad nodes.

The 3rd graph of each row in Figure 5.3 examines the impact of UST_m on MOO performance. Recall that UST_m is to be compared with US_m to determine if a service experience is positive or negative for trust assessment for BRS and for competence trust assessment for SRM and TRM. We observe that as UST_m increases, MOO performance increases (and levels off). The reason is that a high UST_m has the effect of penalizing bad nodes with low trust and can effectively remove bad nodes from participating in future service requests. Another noteworthy observation is that unlike BRS and SRM, TRM is relatively insensitive to UST_m. We attribute the insensitivity to TRM severely punishing a bad node (i.e., dropping its trust to zero) once a bad node's integrity trust falls below the minimum ignorance trust of 0.5.

The rightmost graph of each row in Figure 5.3 compares US_m calculated from (5.10) as more service requests (labeled as "Operation #" on the x coordinate) are executed over time for the three trust-based algorithms against the non-trust-based algorithm. We consider a combination of $P_{risk} = 70\%$ and $P_{bad} = 30\%$ to reveal interesting trends. Because of high P_{risk}, even trust-based algorithms are fooled into selecting bad nodes in the first few service requests. So the first few service requests do not pass UST_m. As a result, bad nodes selected to provide services in the first few service requests are penalized with trust decrease and likely to be filtered out from later service requests. This is evidenced by the result that the last 8 service requests have high US_mvalues. In particular, for multi-trust TRM, US_m is above 90% for the last 8 service requests because mostly only good nodes are being selected by the trust-based algorithm due to dynamic trust update. On the contrary, the non-trust-based algorithm consistently yields a low US_m service request by service request because it has no effective way of filtering out bad nodes. This trend supports our claim that trust-based algorithms can effectively

achieve high user satisfaction despite the presence of bad nodes performing self-promotion attacks, especially after trust convergence occurs. In particular, we observe that US_m under TRM or SRM is consistently higher than that under BRS. We attribute this to the use of integrity trust as confidence for competence trust, thus providing a more accurate estimate of the trustworthy service quality of an SP. Furthermore, TRM consistently outperforms SRM because of its ability to discern bad nodes from good nodes. It is worth noting that one main reason for having a low US_m in a service request is that the SR does not have enough information about the trust of SPs bidding for services. Consequently, the SR must select SPs for service request execution based on their advertised scores. This is a serious problem for the non-trust-based algorithm because SRs do not share experiences through recommendations. As a result, we see that the non-trust-based algorithm has the most severe zigzag pattern of US_m among all. In particular, the zigzag pattern is most pronounced for service requests 5, 9, 11, 13 and 15 at which the responsible SRs have never issued a service request before.

As we go from small to large sized problems, we observe a remarkable similarity with TRM outperforming SRM and BRS. This demonstrates the scalability of our trust algorithm design. That is, our trust-based algorithm especially with TRM as the underlying trust protocol can find near optimal solutions with linear runtime complexity as the problem size increases from small ($|\mathcal{N}| = 60$ and $|\mathcal{S}_m| = 4$), to medium ($|\mathcal{N}| = 120$ and $|\mathcal{S}_m| = 8$) and large ($|\mathcal{N}| = 240$ and $|\mathcal{S}_m| = 16$).

5.5.3 Effect of Service Quality Constraints and Opportunistic Service Attacks

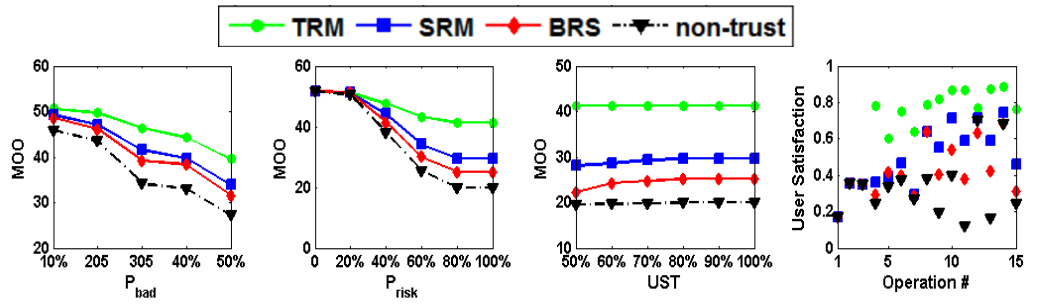
In this section, we perform a sensitivity analysis of the results with respect to the minimum service quality constraints including the minimum service quality requirement at the abstract service level $S_k^{\mathrm{THRES}} = (Q_k^{THRES}, D_k^{THRES}, C_k^{THRES})$ and the minimum service quality requirement at the service request level $\mathrm{SCS}_m^{THRES} = (Q_m^{THRES}, D_m^{THRES}, C_m^{THRES})$ specified by the user as in (5.4).

Table 5.3: Minimum Service Quality Constraints.

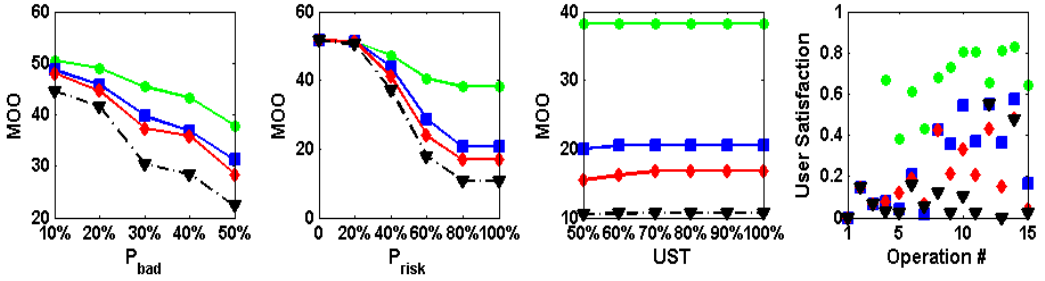
Case	$(\boldsymbol{Q_k^{THRES}}, \boldsymbol{D_k^{THRES}}, \boldsymbol{C_k^{THRES}})$	$(\boldsymbol{Q_m^{THRES}}, \boldsymbol{D_m^{THRES}}, \boldsymbol{C_m^{THRES}})$
Light	$(1, 5, 5)$	$(4, 20, 20)$
Medium	$(2, 4, 3)$	$(8, 16, 12)$
Tight	$(3, 3, 2)$	$(12, 12, 8)$

Table 5.3 lists three conditions: light, moderate, and tight. Service quality constraints induce "opportunistic service" attack behaviors in two ways.

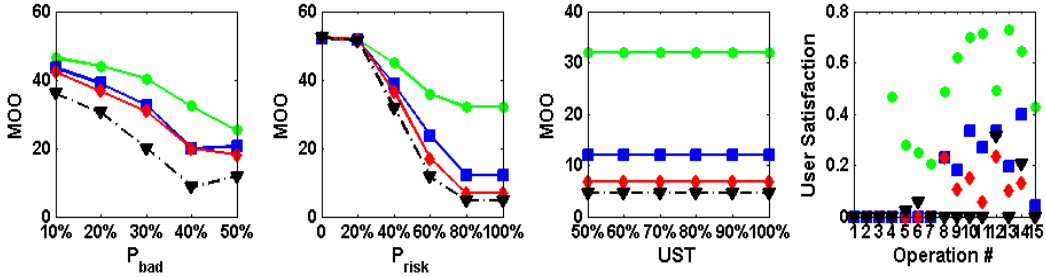
First, a bad node may want to lie about its service quality to at least pass the abstract service level Q, D and C constraints in order for it to have a chance to be selected for service request execution. Second, once a bad node is selected for a service request, it is strongly motivated to contribute at least a minimum effort to satisfy the service request level constraint; otherwise, the service request is considered a failure from the user's perspective and the malicious node will be penalized with a trust loss.



(a) Light Service Quality Constraints.



(b) Medium Service Quality Constraints.



(c) Tight Service Quality Constraints.

Figure 5.4: Effect of Service Quality Constraints and Opportunistic Service Attacks.

Figure 5.4 (a)-(c) show the simulation results for the light, medium, and tight minimum service quality constraint cases, respectively, for the small sized problem.

For comparison, the light minimum service quality constraint case shown in Figure 5.4 (a) corresponds to Figure 5.3 (b) except that a bad node will perform opportunistic service attacks. Even with this opportunistic service attack behavior, we observe from Figure 5.4 that the relative performance rank of TRM, SRM, BRS and non-trust-based design (in this order) remains the same. A major reason is that a bad node often must

perform self-promotion attacks first to increase its chance of being selected after which it can perform opportunistic service attacks. However, opportunistic service attacks cannot offset low user satisfaction toward the bad node resulting from self-promotion attacks. In Figure 5.4, TRM outperforms SRM and BRS because of its ability to discern bad nodes from good nodes even in the presence of opportunistic service attacks.

5.5.4 Effect of *Q*, *D*, *C* Score Distribution

Table 5.4: *Q*, *D*, *C* Score Distribution.

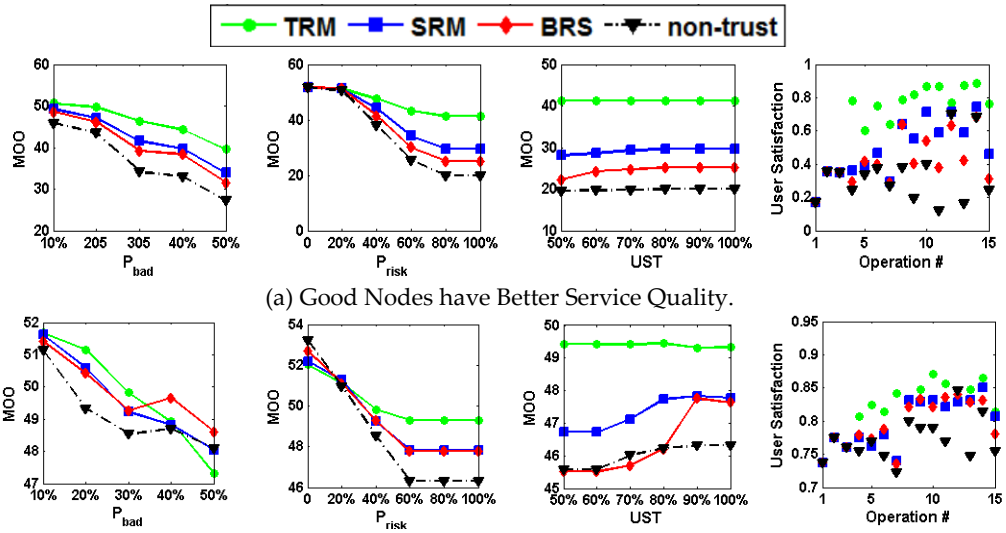
Case	$(\boldsymbol{Q_{bad}}, \boldsymbol{D_{bad}}, \boldsymbol{C_{bad}})$	$(\boldsymbol{Q_{good}}, \boldsymbol{D_{good}}, \boldsymbol{C_{good}})$
Good nodes have better service quality	$([1-3], [3-5], [2-5])$	$([3-5], [1-4], [1-2])$
Equal	$([3-5], [1-4], [1-2])$	$([3-5], [1-4], [1-2])$

In this section, we perform a sensitivity analysis of the results with respect to Q, D, C score distributions for bad nodes and good nodes in terms of $(Q_{bad}, D_{bad}, C_{bad})$ and $(Q_{good}, D_{good}, C_{good})$. Table 5.4 lists two test cases: "good nodes have better service quality" vs. "equal" (good nodes and bad nodes have equal service quality). We ignore the case in which bad nodes have better service quality than good nodes because bad nodes will prevail anyway even with trust-based service composition and binding. All other conditions remain the same as specified in Table 5.2. Note that the results presented so far are based on the "good nodes have better service quality" case.

Figure 5.5 (a)-(b) show the simulation results for the "good nodes have better service quality" and "equal" test cases, for the small sized problem. Figure 5.5 (a) is the same as Figure 5.3 (b) and is replicated here for ease of comparison. We see that in the case of equal service quality, i.e., Figure 5.5 (b), trust-based design still outperforms non-trust-based design with the same performance ranking preserved. However, we see that BRS is just as good as SRM, especially when UST is high and/or more service requests are executed. The reason is that both BRS and SRM in this equal service quality case tend to select moderate to high service quality nodes, regardless of whether they are good or bad nodes. Although SRM also applies scaling trust reward/penalty to good/bad nodes based on integrity trust information, this small differentiation does not prevent bad nodes with medium to high service quality from being selected. As a result, they both select moderate to high service quality nodes for service request execution.

On the other hand, TRM applies a rather strict trust penalty to bad nodes, so it essentially excludes bad nodes from selection and only good nodes with high service quality are selected for service request execution. The results exhibited in Figure 5.5 reveal conditions ("good nodes have better service quality" vs. "equal") under which our trust-based algorithm design is effective compared with the non-trust-based

algorithm design. More specifically, with the service quality about being equal for both good and bad nodes, TRM performs the best when there are more good nodes than bad nodes. This is so because of TRM's unique ability to discern bad nodes from good nodes and to select only good nodes with high service quality. When there are more bad nodes than good nodes, on the other hand, BRS performs the best especially when bad nodes do not need to lie about their service quality for them to be selected for service request execution (i.e., when P_{risk} is low) because in this case BRS, due to its inability to discern bad nodes from good nodes, tends to select high service quality nodes for service request execution even if they are bad nodes.



(a) Good Nodes have Better Service Quality.

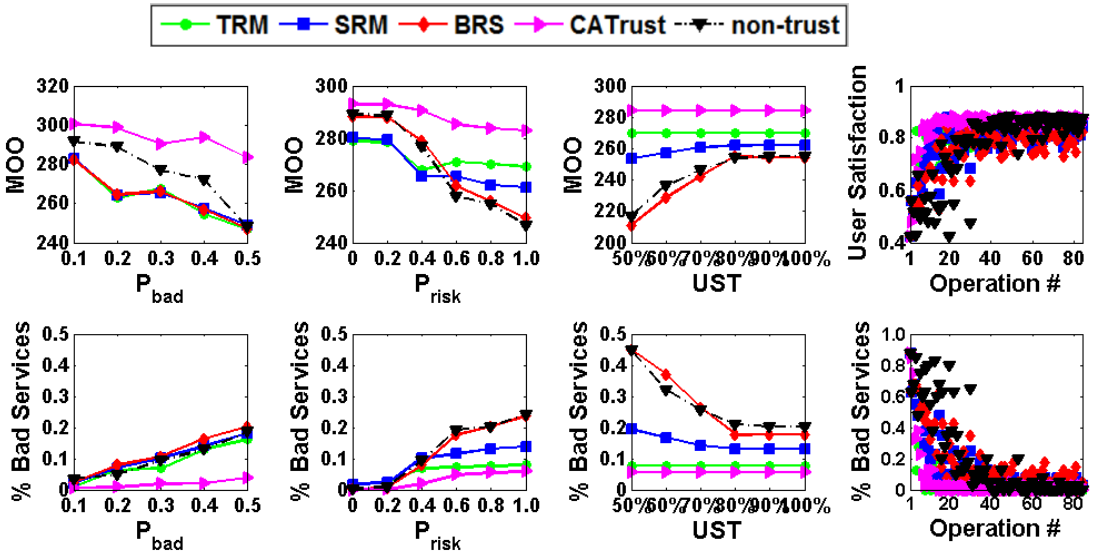
(b) Good Nodes and Bad Node have Equal Service Quality.

Figure 5.5: Effect of Q, D, and C Score Distributions for Good and Bad Nodes on Performance.

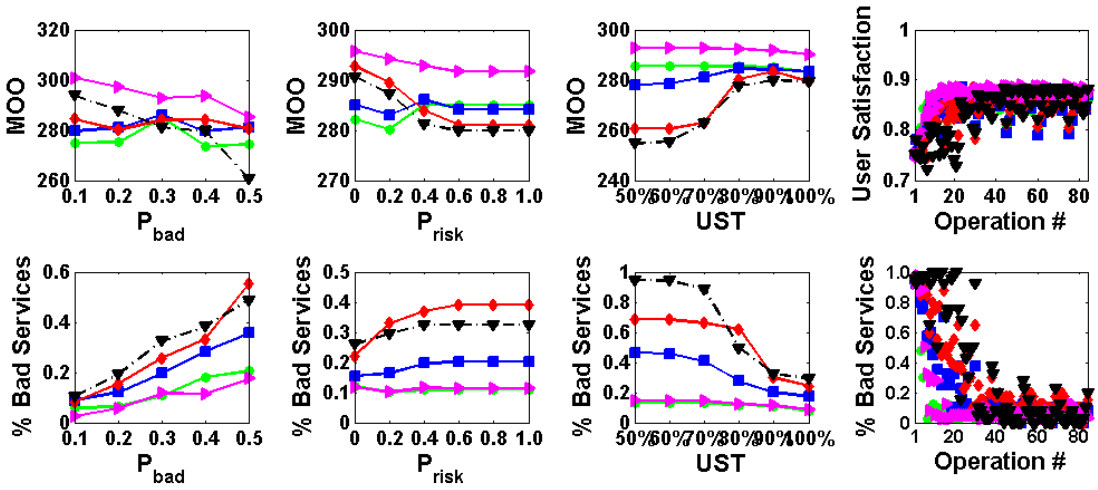
Another interesting result is that when bad nodes have about the same service quality as good nodes, there is an optimal *UST* level under which the performance of our trust-based algorithm is maximized. This is evident from the MOO vs. *UST* graph (the 2nd rightmost graph) in Figure 5.5 (b). The reason is that if *UST* is too high (e.g., 100%), $US < UST$ is true and bad nodes with high service quality originally selected for service request execution will be identified as culprits and will be penalized with trust degradation. This in effect will block bad nodes from participating in future service request executions. Consequently, the system will be forced to select only good nodes for servicing future service request. Since not all good nodes are of high service quality, inevitably the resulting MOO value is not as high as it would be if only high service quality nodes (good or bad) are selected for service request execution. In effect, *UST* can be used as a design parameter to maximize the MOO value.

5.5.5 Performance Comparison of CATrust vs. Single-Trust and Multi-Trust Protocols

In this section, we perform a comparative performance analysis of CATrust vs. single-trust and multi-trust protocols as the underlying trust protocol for the trust-based algorithm for solving the service composition and binding MOO problem.



(a) Good Nodes have Better Service Quality.



(b) Good Nodes and Bad Node have Equal Service Quality.

Figure 5.6: Performance Comparison of CATrust vs. Single-trust and Multi-trust Protocols.

Figure 5.6 (a)-(b) compare CATrust with BRS, TRM and SRM for the "good nodes have better service quality" and "equal" test cases (as in Table 5.4), respectively, for the small sized problem. The first row compares the MOO value and user satisfaction,

while the second row compares the percentage of malicious nodes selected for service execution in this service composition and binding application with MOO requirements. One can clearly see that CATrust outperforms the single-trust (BRS) and multi-trust (TRM, SRM) counterparts by a wide margin, especially when good nodes have better service quality as shown in Figure 5.6 (a). In particular, the percentage of malicious nodes selected for service execution under CATrust is significantly lower than that under BRS, TRM, or SRM. We attribute the superiority of CATrust to its ability to associate context to solution quality delivered by an SP. Because a service request issued by an SR is context dependent (location and incentive), CATrust is able to learn an SP's context-dependent service behavior and consequently the delivered service quality, given location and incentive as input. On the contrary, BRS, TRM, and SRM do not take context into consideration and merely predict an SP's delivered service quality across the location and incentive context space. We note that even for the case in which good nodes and bad nodes have about equal service quality, as shown in Figure 5.6 (b), CATrust still outperforms BRS, TRM, and SRM due to this context-aware ability. In this case, for the percentage of malicious nodes selected for service execution, CATrust performs comparably with TRM because bad nodes selected can also provide quality service to maximize the MOO value and user satisfaction.

5.6 Summary

In this chapter, we proposed a trust-based service composition and service binding algorithm with linear runtime complexity for MOO in service-oriented MANET characterized by space-time complexity of mobile devices wherein nodes may be malicious and/or information received is often erroneous, partly trusted, uncertain and incomplete.

We investigated single-trust and multi-trust as the building block for our trust-based algorithm design and demonstrated that the multi-trust-based algorithm outperforms the non-trust-based and single-trust-based counterparts, and approaches the ideal solution quality obtainable by the ILP solution. We also performed sensitivity analysis to identify conditions under which trust-based service composition is most effective compared with non-trust-based service composition. We discovered that our threshold based model (TRM) performs the best among all because TRM can best discern bad nodes from good nodes. Our analysis result backed by simulation reveals that in case good nodes have higher service quality than bad nodes, multi-trust protocols and in particular TRM, which severely penalizes malicious attack behavior, will perform the best. However, in case bad nodes have equal service quality as good nodes, TRM will perform the best only when there are more good nodes than bad nodes. Otherwise,

single-trust protocols such as the Beta Reputation System will perform the best, especially when bad nodes do not need to lie about their service quality for them to be selected for service request execution. In the latter case, there exists an optimal user satisfaction threshold in service quality under which the protocol performance is maximized.

Lastly we performed a comparative performance analysis of CATrust vs. single-trust and multi-trust protocols as the underlying trust protocol for executing the trust-based algorithm to solve the service composition and binding MOO problem. We demonstrated that CATrust outperforms the single-trust (BRS) and multi-trust (TRM, SRM) counterparts by a wide margin, especially when good nodes have better service quality. We attribute the superiority of CATrust to its ability to associate context (location and incentive) to solution quality delivered by an SP. Because a service request issued by an SR is context dependent, CATrust is able to learn an SP's context-dependent service behavior and, consequently, accurately predict the delivered service quality.

Chapter 6

Trust-based Task Assignment with Multi-Objective Optimization in Service-Oriented Ad Hoc Networks

In this chapter, we validate the concept of trust-based service management with a node-to-task assignment application in service-oriented MANETs. We devise a trust-based heuristic algorithm based on auctioning to solve this node-to-task assignment problem with multi-objective optimization (MOO) requirements. Our trust-based heuristic algorithm has a polynomial runtime complexity, rather than an exponential runtime complexity as in existing work, thus allowing dynamic node-to-task assignment to be performed at runtime. We perform a comparative analysis of our trust-based algorithm based on CATrust and Beta Reputation System (BRS) [69] against the ideal solution with perfect knowledge over node reliability as well as a non-trust baseline scheme and demonstrate that our trust-based algorithm outperforms the non-trust-based counterpart using blacklisting techniques and performs close to the ideal solution quality. We show that CATrust outperforms Beta Reputation because CATrust considers the association between context and MOO treated as the service quality metric for the task assignment application. We also develop a table-lookup method to apply the best trust model parameter settings upon detection of dynamically changing environmental conditions to maximize MOO performance.

Table 6.1 summarizes the acronyms and symbols used in the chapter.

6.1 Node-to-Task Assignment with MOO

We consider a mission-driven service-oriented MANET that must handle dynamically arriving tasks to achieve multiple system objectives. Each task requires an SR to be the task lead and assemble a team among SPs to accomplish the task. We

consider M ordered node types, NT_1,… NT_M, such that a higher node type has a higher capability than a lower node type. A node with a high node type also may involve a human operator and thus has additional trust dimensions pertaining to social trust [17] [29]. When mobile nodes are not involved in a task, they follow their routine-work mobility model.

Table 6.1: Acronyms and Symbols.

Acronym/Symbol	Meaning
CN	Commander node
TL	Task lead
NT	Node type
STO	Service trust only
SSTRT	Separating service trust from recommendation trust
ISTRT	Integrating service trust with recommendation trust
α, β	Amount of positive evidence, amount of negative evidence
$\mathcal{N}, \lvert\mathcal{N}\rvert$	Node set in the system, # of nodes in the system
$\mathcal{T}, \lvert\mathcal{T}\rvert$	Task set in the system, # of tasks in the system
R, U, D	Reliability, utilization variance, delay to task completion
$\bar{R}$, $\bar{U}$, $\bar{D}$	Scaled R, U, D
$\omega_R : \omega_U : \omega_D$	Weights associated with R, U, D for multi-objective optimization
P_{MOO}	$\omega_R \bar{R} + \omega_U \bar{U} + \omega_D \bar{D}$
$[I_m, NT_m, N_m, F_m, (t_m^{start}, t_m^{end})]$	Task m's specification: importance, node type, number of nodes needed for task execution, task flow, task start time and end time (deadline)
N_B	Number of bidders in response to task m's specification advertisement
$[U_j, D_j, NT_j]$	Node j's specification: utilization, execution time, node type
DT_m, $DT_{mission}$	Task m execution duration, mission execution duration
$T_{i,j}$	Trust of node i has toward node j
n_{rec}	Maximum number of recommenders for trust propagation
P_b	Percentage of malicious nodes

Figure 6.1 illustrates the system architecture of nodes and tasks. We consider a mission-driven service-oriented MANET that must handle dynamically arriving tasks in a mission setting to achieve multiple system objectives. A commander node (CN) governs the mission team. Under the CN, multiple task leads (TLs) lead task teams. The CN selects TLs at the beginning of network deployment based on the trustworthiness of nodes known to CN a priori and the TLs (acting as SRs) each recruit trustworthy SPs dynamically for executing the tasks assigned to them. Tasks that overlap in time are grouped into a chunk. In Figure 6.1, tasks 1-4 are grouped into chunk 1 and tasks 5-6 are

grouped into chunk 2. Tasks in the same chunk must compete with each other for services provided by qualified SPs as each SP can participate in only one task at a time. Here we make the assumption that the TLs selected by the CN are fully trusted. In military command and control environments, a hierarchical chain of command and control is a common network structure [6] wherein this assumption is justified. One example is a tactical convoy operation where a commander and multiple assistant commanders work together to control members to maximize communication and efficiency. Assistant commanders are selected a priori and the trust relationships between the commander and assistant commanders are assumed [3].

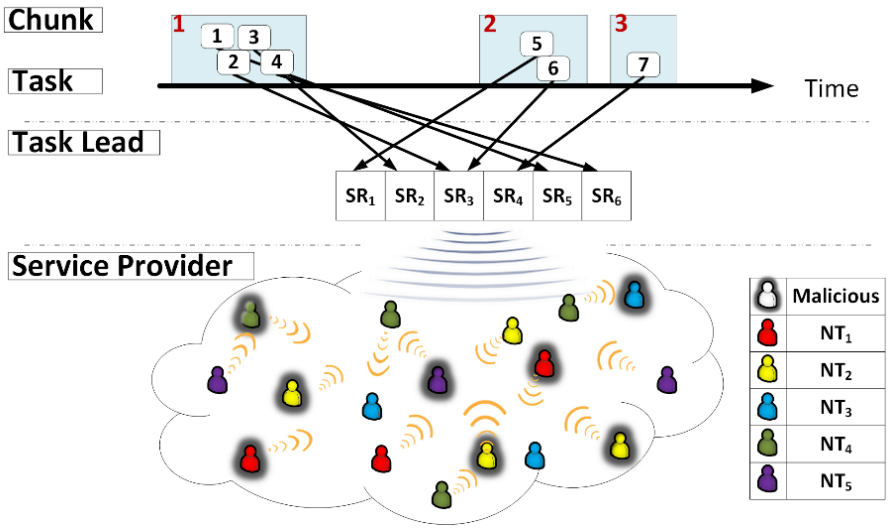


Figure 6.1: System Architecture of Nodes and Tasks.

Following the Byzantine Failure model [78], we assume that a task fails when at least 1/3 nodes providing bad service. A malicious node can opportunistically collude with other malicious nodes to fail a task, when it senses that there are enough bad nodes around (at least 1/3) at the expense of trust loss. Meanwhile, a malicious node can boost its service quality information so as to increase its chance of being selected as the SP. Our trust protocol deals with self-promotion attacks by severely punishing nodes that fail to provide the advertised service quality during task execution. In practice, self-promotion attacks can be easily detected, and as a result a malicious node would expose itself as vulnerable, resulting in a low trust. This attack is less likely to be performed by a smart attacker.

6.1.1 Task Model

Tasks arrive asynchronously and may start and complete at different times. Each task is characterized by the following unique properties:

- *Importance* (I_m) refers to the impact of task failure on mission completion with higher values indicating more importance.
- *Required node type* (NT_m) indicates the required functionality of nodes for executing task m. A node with a higher node type has a higher capability and, because of human involvement, also has social trust dimensions.
- *Required number of nodes* (N_m) refers to the number of nodes needed for execution of task m.
- *Task execution Flow* (F_m) indicates the task structure (sequential, parallel or both) by which nodes coordinate with each other. For simplicity, we assume N_m nodes execute the task sequentially.
- *Task execution timeframe* (t_m^{start}, t_m^{end}).

A task fails when (a) the TL cannot recruit enough SPs to execute the task; (b) the task is not completed by the task end time (deadline), or (c) when it suffers from a Byzantine failure due to conflicting service attacks which can result from malicious nodes purposely delaying task execution time to cause a task failure. When a task fails due to (b) or (c), we assume that the TL can differentiate the guilty parties from lawful members and will apply a penalty to guilty parties by either blacklisting the guilty parties for the non-trust-based scheme, or applying a trust loss to the guilty parties in terms of $\Delta\beta$ for the trust-based scheme as discussed in Section 6.3.

6.1.2 Trust Protocol

Our baseline trust protocol uses Beta Reputation [69] modeling a trust value in the range of [0, 1] as a random variable where α and β represent the amounts of positive service evidence and negative service evidence respectively, such that the estimated mean trust value of a node is $\alpha/(\alpha+\beta)$. When a task which a node participated in is executed successfully (unsuccessfully), this node's α is incremented by $\Delta\alpha$ (β is incremented by $\Delta\beta$ correspondingly). When we want to severely punish malicious behavior, we set $\Delta\beta \gg \Delta\alpha$. In this paper, we propose a "penalty severity" parameter denoted by $\Delta\beta{:}\Delta\alpha$ to analyze its effect of trust penalty severity on our trust protocol performance. For all nodes, the initial α and β values are 1, representing ignorance with the initial trust value of 0.5.

To reduce message overhead especially for large MANETs, trust propagation is not by flooding. Instead, trust propagation is performed (via a recommendation message) only when a trustor node encounters other nodes. A trustor node evaluating a trustee node will select n_{rec} recommenders whom it trusts most to provide trust recommendations toward the trustee node. A recommender should only pass its direct interaction experience with the trustee node in terms of (α, β) as a recommendation to

avoid dependence and looping. After a task is completed, the TL can serve as a recommender toward the members in its team because it had gathered interaction experiences. For trust aggregation, each trustor aggregates trust evidence of its own (α, β) with a recommender's (α, β) toward the trustee node. Note that a recommender's (α, β) trust evidence is discounted based on the concept of belief discounting, such that the lesser the trustor node trusts the recommender, the more the recommendation is discounted. Because a bad node can perform bad-mouthing and ballot-stuffing attacks, it can provide a bad recommendation (with $\beta \gg \alpha$) toward a good node and a good recommendation (with $\alpha \gg \beta$) toward a bad node, respectively. It can be shown that Beta Reputation is resilient to such attacks if the trustor node has a low trust value toward the bad recommender.

Below we do a cost analysis of the communication/memory overhead involved. Based on our protocol design, a recommender propagates its (α, β) recommendations toward other $|\mathcal{N}| - 2$ trustee nodes upon encountering a trustor node. The communication overhead per node in terms of bits/sec (from node i's perspective) is $\sum_{j=1}^{|\mathcal{N}|-1} \lambda_{ij} (|\mathcal{N}| - 2)(b_I + b_o)$ where b_I is the information bits holding (α, β) of a trustee node, b_o is the encryption bits for secure communication, and λ_{ij} is the encountering rate of node i (the recommender) with node j (the trustor node) which can be derived by analyzing the encounter pattern, e.g., a power-law distribution, in a mobility model such as SWIM and LSWTC [24]. Here we note that because trust propagation is encountered-based, trust convergence does not depend on the size of the network but depends on the encountering rate of node i (the recommender) with node j (the trustor node). Essentially upon encountering node j, node i exchanges trust information with node j in terms of the (α, β) value pairs of other nodes in the MANET community. In practice the message overhead is lower because one can combine all required information into one message during transmission.

The energy consumption overhead per node in terms of J/sec (from node i's perspective) is the sum of energy consumption rate for reception and energy consumption rate for transmission, i.e., $\sum_{j=1}^{|\mathcal{N}|-1} \lambda_{ij} (|\mathcal{N}| - 2)(b_I + b_o) E_R + \sum_{j=1}^{|\mathcal{N}|-1} \lambda_{ij} (|\mathcal{N}| - 2)(b_I + b_o) \left(1 + \frac{1}{p}\right) E_T$ where E_R is the reception energy consumption rate (J/bit), E_r is the transmission energy consumption rate (J/bit), and $p = e^{-\sum_{j=1}^{|\mathcal{N}|-1} \lambda_{ij}(T_{RTS}+T_{CTS})}$ is the probability that other nodes within radio range are not transmitting during $T_{RTS} + T_{CTS}$ and thus $1/p$ is the number of trials before node i clears

the channel for transmission based on the RTS/CTS transmission protocol [41]. Here λ_{ij} is the encountering rate between node i and node j, accounting for the rate at which node j will transmit a recommendation packet to node i when they encounter during $T_{RTS} + T_{CTS}$, thus causing a collision and triggering a packet transmission.

The memory overhead of our protocol is minimum. Each node only needs to allocate space to store its (α, β) value pairs toward other $|\mathcal{N}| - 1$ nodes in the MANET community.

6.2 MOO Problem Definition with Performance Metrics

Without loss of generality, we consider three objectives, namely, mission reliability (R), utilization variance (U), and delay to task completion (D). We note that energy may also be an important objective for MANET applications which must run a long time without energy replenishment. In such environments, energy consumption minimization is another conflicting goal with reliability maximization and delay minimization. The trust-based algorithm design principle for maximizing task allocation MOO performance for the three objectives considered in this paper can still be applied if additional objectives (such as energy minimization) need to be considered. One can view minimizing utilization variance as maximizing load balance, and minimizing delay to task completion as maximizing QoS. These three objectives are further defined below.

- **Mission Reliability** ($\boldsymbol{R}$): This is the task reliability weighted by the task importance I_m, computed by:

$$R = \sum_{m \in L} R_m \frac{I_m}{\sum_{m \in L} I_m} \tag{6.1}$$

  L is the set of tasks in the mission; R_m, a binary number in $\{0, 1\}$, is the task reliability of task m. Following the Byzantine failure model, we assume that a task fails when there are at least 1/3 bad nodes executing the task. Higher R is desirable. To achieve this objective, a TL should select highly trustworthy nodes.

- **Utilization Variance ($\boldsymbol{U}$):** This measures the average utilization of nodes and is defined by:

$$U = \frac{\sum_{i \in N}(|U_i - \widetilde{U}|)}{|N|} \text{ where } U_i = \sum_{m \in L} U_{i,m} \tag{6.2}$$

  N is the set of legitimate member nodes. $U_{i,m}$ is $DT_m/DT_{mission}$ if node i executes task m, and is zero otherwise, where DT_m is the task duration and $DT_{mission}$ is the mission duration. U_i is the overall utility of node i. $\widetilde{U}$ is the average utility of all

nodes. $|U_i - \tilde{U}|$ is the utilization variance of node i to the average. Lower U is desirable as it minimizes the utility variance and achieves the load balance objective. To achieve this goal, a TL should select nodes with low utilization.

- **Delay to Task Completion (D):** This is the average delay to task completion over all tasks, defined by:

$$D = \frac{\sum_{m \in L} D_m}{|L|} \text{ where } D_m = t_m^{complete} - t_m^{start} \quad (6.3)$$

$t_m^{complete}$ is the actual completion time of task m. It is desirable to complete task m as early as possible before the drop-dead end t_m^{end}. If task m is not completed by t_m^{end}, then task m fails and D_m is set to $DT_{mission}$. Lower D is desirable. To achieve this goal, a TL should select nodes with low execution time.

To formulate the MOO problem as a maximization problem, we first scale R, U and D into $\overline{R}$, $\overline{U}$ and $\overline{D}$ such that they each are in the range of [0, 1] and the higher the value to 1, the better the objective is achieved. Specifically, we scale R, U and D by [158]:

$$\bar{R} = \frac{R - R_{min}}{R_{max} - R_{min}}; \bar{U} = \frac{U_{max} - U}{U_{max} - U_{min}}; \bar{D} = \frac{D_{max} - D}{D_{max} - D_{min}} \quad (6.4)$$

Here R_{max} and R_{min} , U_{max} and U_{min} , and D_{max} and D_{min} are the maximum and minimum values of R, U, and D, respectively at the mission level. One can view $\overline{U}$ after scaling as "load balance" in the range of [0, 1], and $\overline{D}$ as "QoS" in the range of [0, 1]. Here we aim to solve the MOO problem by maximizing $\overline{R}$, $\overline{U}$ and $\overline{D}$, given node and task characteristics as input. We adopt the *weighted sum* form converting the MOO problem to a single-objective optimization problem. Specifically, we formulate the MOO problem as:

$$\text{Maximize } P_{MOO} = \omega_R \bar{R} + \omega_U \overline{U} + \omega_D \overline{D} \quad (6.5)$$

Here ω_R, ω_U and ω_D are the weights associated with $\bar{R}$, $\overline{U}$ and $\overline{D}$ with $\omega_R + \omega_U + \omega_D = 1$.

6.3 Trust-based Dynamic Task Assignment

We have two layers of task assignment: by a CN to TLs and by each TL to nodes. We assume that the TLs selected by the CN are fully trusted. A TL is assigned to execute one task at a time. A node can participate in only one task at a time, although it may participate in multiple tasks during its lifetime. TLs advertise tasks and free nodes respond as described next. Below we describe our heuristic-based dynamic task assignment algorithm design based on auctioning with the objective to achieve MOO with a polynomial runtime complexity. Figure 6.2 lists the pseudo code of the actions

taken by the TL acting as the SR and the actions taken by well-behaved and malicious SPs during the execution of the trust-based task allocation algorithm.

6.3.1 Advertisement of Task Specification

The task specification disseminated during the auction process includes a set of requirements for task execution specified by:

$$[ID_m, I_m, NT_m, F_m, (t_m^{start}, t_m^{end})] \tag{6.6}$$

where ID_m is the identifier of task m.

6.3.2 Bidding a Task

When a node receives the task specification message by a TL, it makes a bidding decision on whether to bid the task or not. A node meeting the node type requirement NT_m is considered capable of handling the required work elements imposed by task m and will respond to the request with its node ID if it is free. To help the TL make an informed decision, a node, say node j, sends its service information to the TL as follows:

$$\left[ID_j, U_j, D_j, NT_j\right] \tag{6.7}$$

Here ID_j is the identifier of node j, U_j is the utility of node j so far at the time of bidding, and D_j is the time required by node j to execute task m.

6.3.3 Member Selection

TLs implicitly seek to optimize the MOO function. However, to achieve run-time efficiency, they adopt heuristics with local knowledge of node status to work independently of each other. The TL of task m ranks all bidding nodes (node j's) based on $\omega_R \bar{R}_j + \omega_U \bar{U}_j + \omega_D \bar{D}_j$ where $\bar{R}_j$, $\bar{U}_j$ and $\bar{D}_j$ are defined as:

$$\bar{R}_J = T_{TL,j}; \bar{U}_J = \frac{U_{max} - U_j}{U_{max} - U_{min}}; \bar{D}_J = \frac{D_{max} - D_j}{D_{max} - D_{min}} \tag{6.8}$$

Here a TL considers U_j (utilization of node j) instead of U (utilization variance of all nodes who have participated in at least one task) because it does not have global knowledge about the latter and picking nodes to minimize utilization variance essentially can be achieved by picking nodes with low U_j (equivalently with high $\bar{U}_j$ after scaling). Also a TL uses $\bar{R}_j = T_{TL,j}$ or its trust toward node j to predict task reliability if node j (a bidder) is selected for task execution. Top N_m nodes with the highest ranking scores are selected to execute task m. Here we note that the trust-based algorithm has a polynomial runtime complexity $O(N_B log(N_m))$ where N_B is the number

of bidders. This is so because it only needs to examine all bidders once and selects the top ranked N_m bidders.

```
# TL Execution in each task chunk:
if a task is assigned then
    N_m ← number of nodes required for the task
    taskExecuted ← false
    while not taskExecuted do
        advertise task specification to all SPs
        S ← all SPs that bid the task
        Ŝ ← SPs that are selected for task execution based on Eq. (8)
        send offers to Ŝ
        Ŝ' ← SPs that commit to the task
        N_m ← N_m − |Ŝ'|
        if N_m == 0 then
            taskExecution ← true
            wait for task execution
            update and broadcast trust of participant SPs
        end
    end
end

#  SP Execution in each task chunk:
initialize committed ← false, P ← ∅, R ← ∅
if task requests received and not committed then
    P ← received task requests
    NT_x ←SP's node type
    foreach p in P do
        NT_p ← required node type by p
        if NT_x ≥ NT_p then
            send bidding information to the TL of p
        end
    end
    R ← all tasks that make offers
    if |R| ≥ 1 then
        r ← the task with the highest importance in R
        send task commitment to the TL of r
    end
    wait for the TL decision
    if SP is selected then
        committed ← true
        if SP is malicious and Byzantine failure condition meets then
            fail r
        else
            execute r
        end
    end
end
```

Figure 6.2: Actions taken by the SR and SPs during Auctioning for Task Assignment.

6.3.4 Task Commitment by Nodes

A node may receive more than one offer from multiple TLs where tasks arrive concurrently. A TL sends out a winner notification with the full list of winners where the winners are potential members that are selected by the TL to execute its task. A node determines which task to join based on the expected payoff. This depends on if the node is a good node or a bad node. For a good node, it selects the task of the highest importance to join so as obtain the highest trust gain. For a malicious node, it does the same in order to gain high trust except when the Byzantine Failure condition is satisfied, i.e., at least 1/3 of the task members are malicious nodes. In the latter case, the bad node selects the highest important task that is bound to fail to join to cause the greatest damage to the mission at the expense of trust loss.

6.4 Numerical Results and Analysis

Table 6.2: Parameters Used in Simulation.

Parameter	Value	Type
$(\|\mathcal{N}\|, N_m)$	(20, 3), (100, 15), (1000, 150)	Input
$\|\mathcal{T}\|$	180	Input
I_m	1-5	Input
D_j	U(1,5) min	Input
$t_m^{end} - t_m^{start}$	U(N_m, $5N_m$)min	Input
P_b	10%-70%	Input
Slope of SWIM	1.45	Input
Max pause time of SWIM	4 hours	Input
ω_R: ω_U: ω_D	(1/3:1/3:1/3),(1/2:1/6:1/3),(1/2:1/3:1/6)	Design
n_{rec}	3	Design
$\Delta\alpha$	I_m	Design
$\Delta\beta$:$\Delta\alpha$	$0.1 - 10$	Design

In this section, we perform a comparative performance analysis of trust-based solutions against ideal solutions based on perfect knowledge, and non-trust-based solutions in terms of MOO performance with Matlab simulation. The mobility pattern of each node is modeled by the SWIM mobility pattern based on the C++ simulator in [76]. We also perform sensitivity analysis of the results with respect to key design parameters and alternative trust protocol designs.

Table 6.2 summarizes key parameter values used for this case study. Our example system considers $|\mathcal{N}| = 20$, 100, and 1000 nodes for small-sized, medium-sized, and large-sized problems, respectively. For the small-sized problem, each task will need

only $N_m = 3$ nodes, for the medium-sized problem, $N_m = 15$ nodes, while for the large-sized problem, $N_m = 150$ nodes. For all problems, there are $|\mathcal{T}| = 180$ tasks arriving dynamically. A node's capability is specified by its node type, ranging from NT_1 to NT_4 equally divided among $|\mathcal{N}|$ nodes. A node's service quality in terms of service time required (regardless of whether it is malicious) is specified by D_j which follows uniform distribution U(1, 5) min. Tasks with overlapping start and end times are grouped into a concurrent "chunk." Task importance is in the range from 1 to 5. We simulate task m's execution duration DT_m by U(N_m, $5N_m$) such that t_m^{end} is $t_m^{start} + DT_m$, defining the task end time (deadline) by which a task must be completed, or it will fail. An effect of this is that nodes with a long execution time delay will not be selected for task execution by the TL of the task to prevent failure. A task's execution time is the sum of those of individual nodes selected for task execution since we consider sequential execution in this paper. The percentage of malicious nodes P_b ranges from 10% to 70% whose effect will be analyzed in this section. The weights associated with multiple objectives, i.e., $\overline{R}$, $\overline{U}$, and $\overline{D}$ in the MOO problem are specified by a weight ratio ω_R: ω_U: $\omega_D$which we vary to analyze its sensitivity on MOO performance. The system designer must provide the weight ratio ω_R: ω_U: ω_D as input to reflect the relative importance of $\overline{R}$ vs. $\overline{U}$ and $\overline{D}$ as dictated by the application requirement.

We consider $|\mathcal{N}| = 20$, 100 and 1000 nodes moving according to SWIM modeling human social behaviors in an $m \times m = 16 \times 16$ (4km×4km) operational region, with each region coving $r = 250m$ radio radius based on 802.11n. The operational area and the radio range remain the same for all problems. In SWIM, a node has a home location and a number of popular places. A node makes a move to one of the population places based on a prescribed pattern. The probability of a location being selected is higher if it is closer to the node's home location or if it has a higher *popularity* (visited by more nodes). Once a node has chosen its next destination, it moves towards the destination following a straight line and with a constant speed that equals the movement distance. When reaching the destination, the node pauses at the destination location for a period of time following a bounded power law distribution with the slope set to 1.45 (as in [76]) and the maximum pause time set to 4 hours. In addition, when a node is selected to execute a task, it moves to the location of the TL and follows the TL over the task execution duration. After completing a task, a node resumes its SWIM mobility pattern until it is selected again for executing another task. The movements and locations of $|\mathcal{N}|$ mobile nodes are simulated this way and then integrated with Matlab for evaluating trust-based, ideal, and non-trust-based solutions.

A malicious node performs attacks as specified in the attack model in Section 3.2. For the trust protocol, the number of recommenders n_{rec} is set to 3. The increment to

positive evidence $\Delta\alpha$ is set to I_m so that the increment is proportional to task importance, while the increment to negative evidence $\Delta\beta$ is set to I_m multiplied by the penalty severity parameter (i.e., $\Delta\beta{:}\Delta\alpha$) in the range of 0.1 to 10, with a larger number representing a more severe penalty to negative evidence. We will analyze the effect of severely punishing malicious behavior (when $\Delta\beta{:}\Delta\alpha$ is much larger than 1) on MOO performance.

We consider two baseline algorithms against which our trust-based algorithm is compared in the performance analysis, namely, ideal selection with perfect knowledge of node status vs. non-trust-based selection, as follows:

- ***Ideal selection***: The TL of task m ranks all bidding nodes in the same way as the trust-based algorithm described earlier except that it has perfect knowledge of node status, i.e., $\bar{R}_j = 1$ if node j is a good node, and $\bar{R}_j = 0$ if node j is malicious. The ideal solution is impossible to achieve; it is just used to predict the performance upper bound to the trust-based solution.
- ***Non-trust-based selection***: The TL of task m also ranks all bidding nodes in the same way as the trust-based algorithm except that $\bar{R}_j = 0$ if the bidding node is blacklisted; $\bar{R}_j = 1$ if the bidding node is not blacklisted and had participated in a successful task execution for which the TL was the task lead; and $\bar{R}_j = 0.5$ (no knowledge) otherwise. We assume intelligent behavior so that each TL can learn from experiences. If a TL experiences a task failure, it blacklists nodes participated in the task execution and excludes them from a future node-to-task assignment for which it is the TL. Top N_m ranked nodes are selected for executing task m.

6.4.1 Comparative Performance Analysis

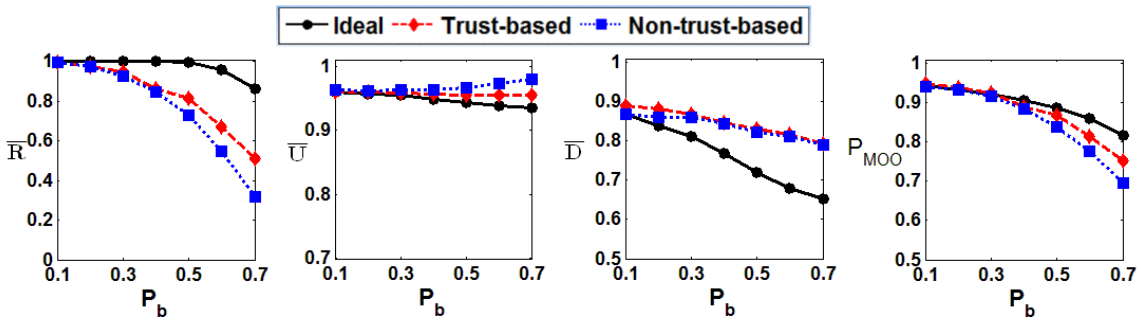


Figure 6.3: Mission reliability, Load Balance, QoS, and P_{MOO} vs. Bad Node Percentage (P_b) for a Small MOO Problem.

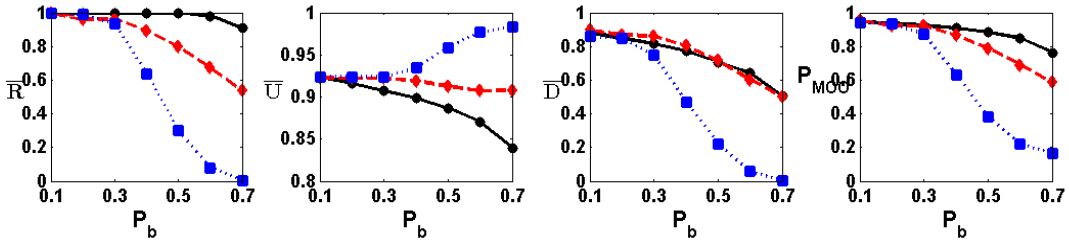


Figure 6.4: Mission reliability, Load Balance, QoS, and P_{MOO} vs. Bad Node Percentage (P_b) for a Medium MOO Problem.

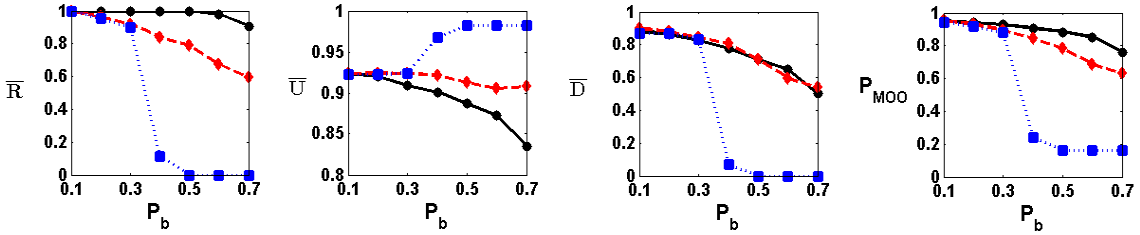


Figure 6.5: Mission reliability, Load Balance, QoS, and P_{MOO} vs. Bad Node Percentage (P_b) for a Large MOO Problem.

Figure 6.3, Figure 6.4 and Figure 6.5 presents the solution quality in terms of the scaled mission reliability ($\bar{R}$), load balance ($\bar{U}$), QoS ($\bar{D}$), and P_{MOO} obtained by the trust-based solution against the ideal solution and the non-trust-based solution for the small-sized problem ($|\mathcal{N}| = 20$, $N_m = 3$), medium-sized ($|\mathcal{N}| = 20$, $N_m = 3$), medium-sized ($|\mathcal{N}| = 100$, $N_m = 15$), and large-sized ($|\mathcal{N}| = 1000$, $N_m = 150$) problems, respectively, as a function of the percentage of malicious nodes in the range of 10% to 70% with $\Delta\beta : \Delta\alpha = 1:1$ and the weight ratio $\omega_R : \omega_U : \omega_D = \frac{1}{2} : \frac{1}{6} : \frac{1}{3}$ for a case in which reliability is more important than QoS and load balance. Note that $\bar{R}$, $\bar{U}$, $\bar{D}$ and P_{MOO} are all scaled in the range of [0, 1] with a higher number indicating a higher performance. Each result point indicates the average value of the metric based on 100 simulation runs, each of which has the same task arrival sequence with the D_j distribution randomized. We observe that Figure 6.3, Figure 6.4 and Figure 6.5 are similar in trend with the trust-based solution outperforms the non-trust- based solution and approaching the ideal scheme in the overall performance. As the problem size increases, the performance gain of our trust-based scheme over the non-trust-based scheme is more pronounce because there are more qualified nodes to select from for task execution. Furthermore, the dominance is more manifested as the percentage of malicious nodes (P_b) increases. There is an interesting tradeoff between the multiple objectives in terms of $\bar{R}$, $\bar{U}$, and $\bar{D}$. The ideal solution attempts to maximize P_{MOO} in (6.5) by maximizing $\bar{R}$ because of its perfect knowledge of node reliability at the expense of $\bar{U}$ and $\bar{D}$. On the other hand, without having sufficient evidence to establish trust (at least initially), the trust-based

solution attempts to maximize $\overline{D}$ without overly compromising $\overline{R}$ and $\overline{U}$. Finally, with only private blacklisting information kept by the TLs, the non-trust-based solution attempts to maximize $\overline{U}$ at the expense of $\overline{R}$ and $\overline{D}$. The ability for the TLs to differentiate good nodes from malicious nodes thus dictates how P_{MOO} in (6.5) is maximized. We conclude that our heuristic trust-based solution with polynomial complexity indeed can achieve solution efficiency without compromising solution optimality.

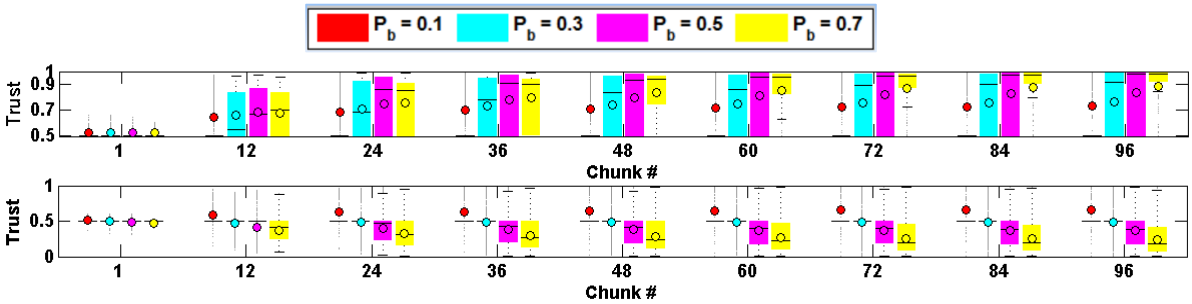


Figure 6.6: Trust Values of Good Nodes (Top) and Bad Nodes (Bottom) over time in Boxplot Format.

Figure 6.6 depicts the trust distribution of good nodes (top) and bad nodes (bottom) in boxplot format as a function of time (chunk #) in our trust protocol. This boxplot is the same for all problem sizes. A boxplot graphically depicts trust values through their quartiles without making any assumption about the distribution. In a boxplot, the bottom and top of a boxplot are the first and third quartiles, and the band inside the box is the second quartile (the median) with the ends of the whiskers showing the minimum and maximum of all of the trust values. The trust value of a good node should be above 0.5 and approaching 1 (ground truth), while the trust value of a bad node should be blow 0.5 and approaching 0 (ground truth). We can see that for $P_b = 10\%$, trust values are less dispersed, so the first and third quartiles are clustered into a thick dot. Further, the trust values of bad nodes are mostly above 0.5 because there are too few bad nodes in the system (2 out of 20) and the chance for them to be in the same task to perform opportunistic service attacks is low. In this case, bad nodes remain hidden and behave, with their trust values maintained above 0.5 to earn the trust reward. As P_b increases, the chance of performing opportunistic service attacks increases. As a result, the trust values of bad nodes are quickly updated to fall below 0.5 because of trust penalty. We also observe that trust convergence is achieved after 50 chunks (about 100 tasks). This is particularly the case when P_b is sufficiently high at which the medium trust value of bad nodes is sufficiently low and the medium trust value of good nodes is sufficiently high, so the system is able to differentiate good nodes from bad nodes for task execution. For example, the medium good node trust value is 0.75 and the medium bad

node trust value is 0.35 when P_b is 50%. This explains why when P_b is 50% in Figure 6.3, Figure 6.4, and Figure 6.5, the trust-based solution outperforms the non-trust-based solution and approaches the ideal solution.

6.4.2 Sensitivity Analysis of Δβ:Δα and ωR:ωU:ωD

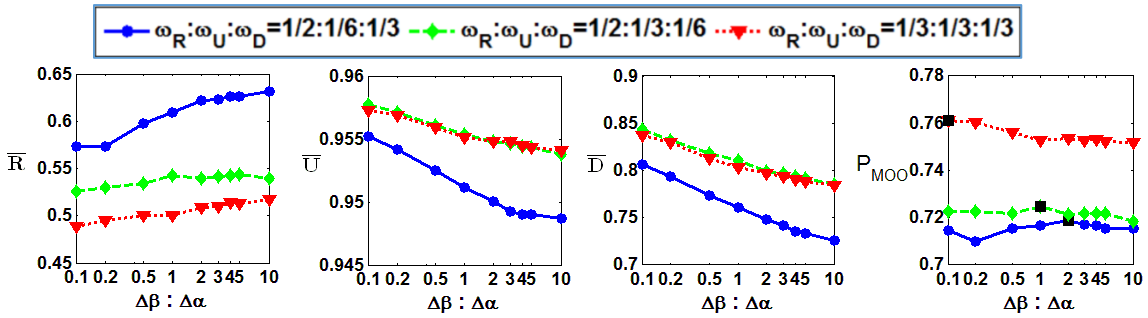


Figure 6.7: Sensitivity Analysis of MOO with respect to Δβ:Δα and ωR:ωU:ωD.

The observation that people hope to severely punish malicious behavior by having a large $\Delta\beta:\Delta\alpha$ ratio is true if mission reliability is the most important or is the sole objective especially for mission-critical applications. However, when there are multiple conflicting objectives such as mission reliability ($\bar{R}$), utilization variance ($\bar{U}$), and delay to task completion ($\bar{D}$) considered in this paper, this observation is not necessarily true. One contribution of this paper is that we identify the best $\Delta\beta:\Delta\alpha$ to use in order to maximize P_{MOO}, depending on the weights associated with multiple objectives, i.e., $\bar{R}$, $\bar{U}$ and $\bar{D}$ in the MOO problem.

Figure 6.7 tests the sensitivity of $\bar{R}$, $\bar{U}$, $\bar{D}$ and P_{MOO} obtained from our trust-based solution with respect to the ratio of the positive increment to the negative increment $\Delta\beta:\Delta\alpha$, and the weight ratio of the weights associated with multiple objectives $\omega_R:\omega_U:\omega_D$ for the case in which $P_b = 70\%$ (picked to show area of interest) for the small sized problem. We see that in general $\bar{R}$ increases while $\bar{U}$ and $\bar{D}$ decrease as $\Delta\beta:\Delta\alpha$ increases because a larger ratio severely punishes bad nodes for performing attacks, making the bad nodes more distinguishable from good nodes. Selecting mostly good nodes for task execution, however, increases $\bar{R}$ but sacrifices $\bar{U}$ because node selection tends to select mostly good nodes, and also sacrifices $\bar{D}$ because bad nodes with good service quality are not selected.

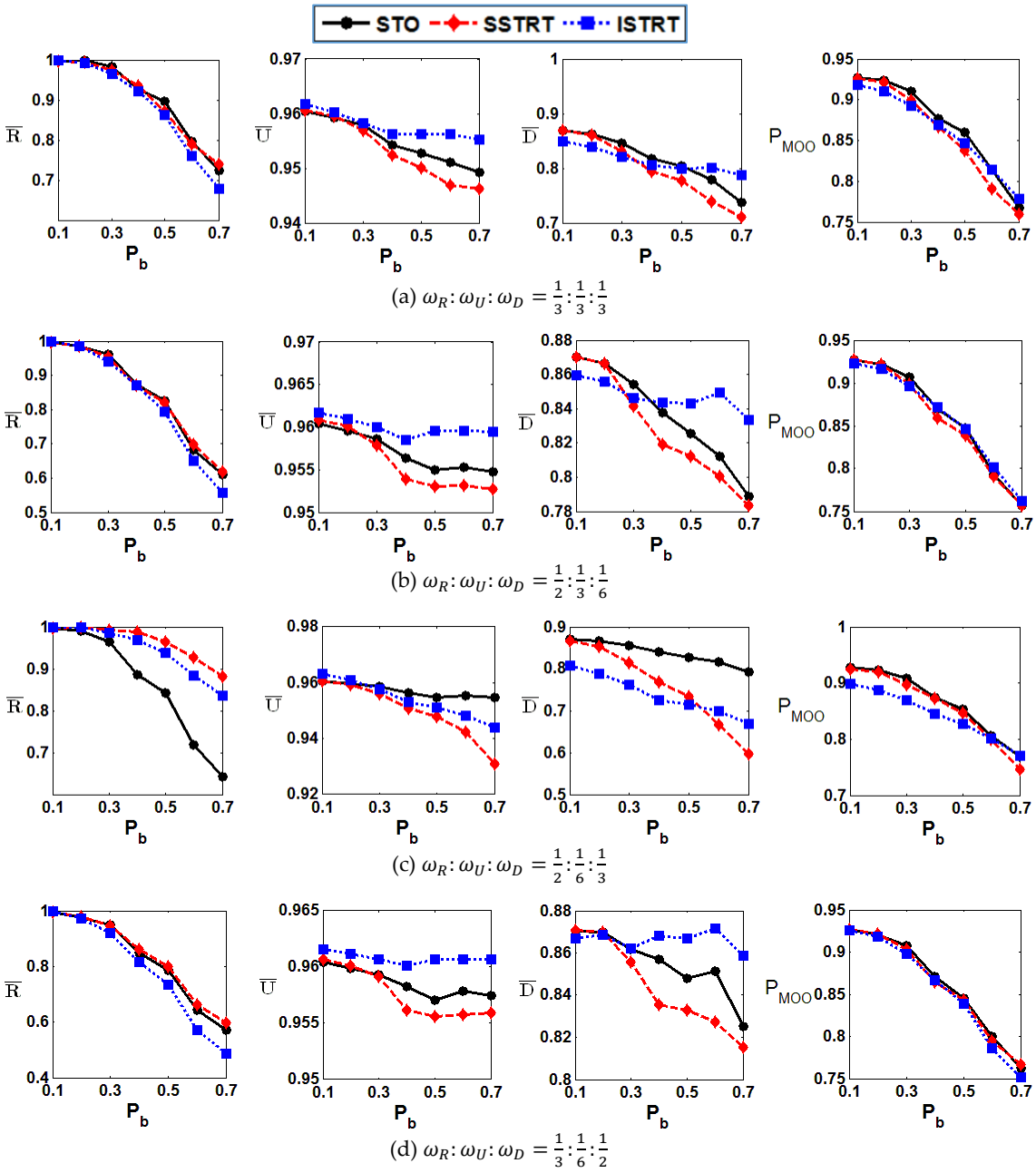
We observe that the best $\Delta\beta:\Delta\alpha$ to maximize P_{MOO} is affected by the weights associated with multiple objectives, i.e., $\bar{R}, \bar{U}$ and $\bar{D}$ in the MOO problem. This is evident in the rightmost graph of Figure 6.7 where we observe that the best $\Delta\beta:\Delta\alpha$ ratios are 0.1, 1, and 2, for $\omega_R:\omega_U:\omega_D = \frac{1}{3}:\frac{1}{3}:\frac{1}{3}, \frac{1}{2}:\frac{1}{3}:\frac{1}{6}$, and $\frac{1}{2}:\frac{1}{6}:\frac{1}{3}$ respectively, for maximizing P_{MOO} in

(6.5). The reason is that a higher $\Delta\beta:\Delta\alpha$ increases $\bar{R}$ but sacrifices $\bar{U}$ and $\bar{D}$ as they are conflicting goals. Hence, under the equal weight scenario (the red line) when all objectives contribute equally, the best $\Delta\beta:\Delta\alpha$ value is small as so to best balance the gain of $\bar{R}$ vs. the loss of $\bar{U}$ and $\bar{\mathrm{D}}$ for MOO. Here we note that although the sensitivity analysis is demonstrated for the case in which $P_b = 70\%$, the general behavior observed is true across. The only difference is the degree of sensitivity.

6.4.3 Sensitivity Analysis of Trust Protocol Design

In this section, we consider the effect of trust protocol design on trust-based MOO performance. Recall that the trust protocol design considered in Section 6.1.2 is based on a service trust value in the range of [0, 1] represented by α and β denoting the amount of positive evidence and negative evidence, respectively. We consider three trust protocol designs as follows:

1. Service Trust Only (STO): this is the protocol described in Section 6.1.2.
2. Separating Service Trust from Recommendation Trust (SSTRT): this protocol behaves the same as STO when updating the service trust. In addition, it maintains a separate trust value for rating a recommender. Specifically, there is a second pair of α and β denoting the amount of positive evidence and negative evidence respectively for rating a recommender. Upon receiving a recommendation from node k regarding node j, node i can compare its own service rating toward j with the recommendation received from k about j. If the difference deviates more than a percentage threshold (25% is considered in the paper), then node i views it as negative evidence against node k because of a possible bad-mouthing or ballot staffing attack by node k. Otherwise, node i views it as positive evidence for node k. When node i receives a recommendation from node k, node i uses the recommendation trust (instead of the service trust) it has toward k for trust merging. Here α and β for recommendation trust are set to 1 initially. The same $\Delta\beta:\Delta\alpha$ ratio applies.
3. Integrating Service Trust with Recommendation Trust (ISTRT): this protocol also considers positive/negative evidence of a recommender as SSTRT does. However, as STO, it only maintains a pair of α and β denoting the amount of positive evidence and negative evidence. Both recommendation quality evidence and service quality evidence collected are combined for updating α and β. Again α and β are set to 1 initially. The same $\Delta\beta:\Delta\alpha$ ratio applies.



(a) $\omega_R:\omega_U:\omega_D = \frac{1}{3}:\frac{1}{3}:\frac{1}{3}$

(b) $\omega_R:\omega_U:\omega_D = \frac{1}{2}:\frac{1}{3}:\frac{1}{6}$

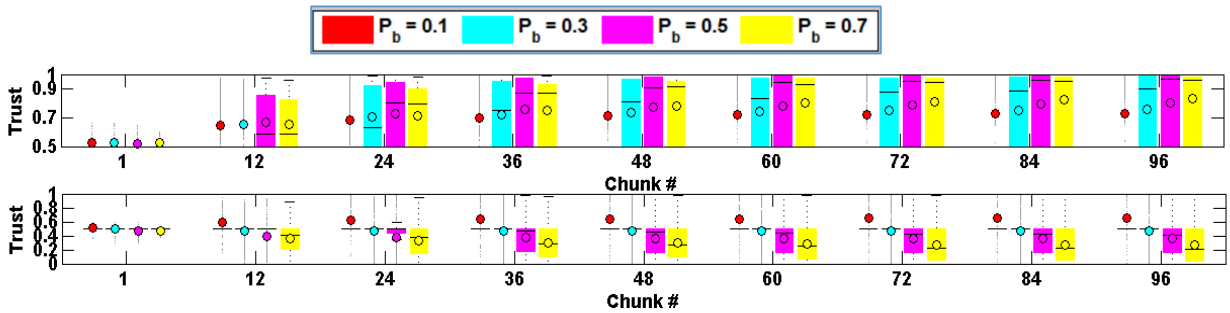
(c) $\omega_R:\omega_U:\omega_D = \frac{1}{2}:\frac{1}{6}:\frac{1}{3}$

(d) $\omega_R:\omega_U:\omega_D = \frac{1}{3}:\frac{1}{6}:\frac{1}{2}$

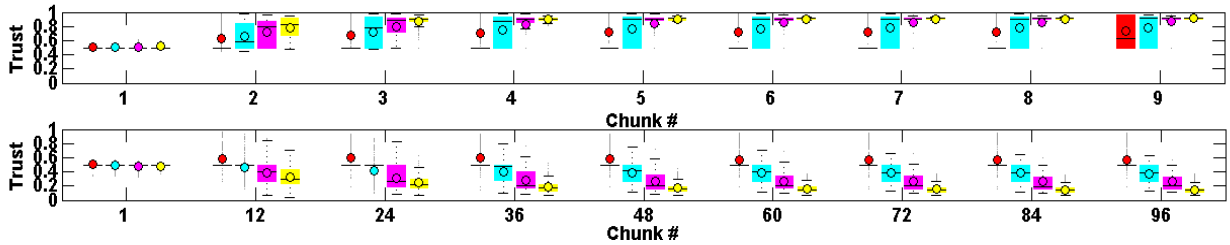
Figure 6.8: Sensitivity Analysis of MOO with respect to Trust Protocol Design: STO vs. SSTRT and ISTRT.

The effect of trust protocol design on MOO performance is summarized in Figure 6.8 and Figure 6.9. Figure 6.8 (a), (b), (c) and (d) compare $\bar{R}$, $\bar{U}$, $\bar{D}$ and P_{MOO} obtained by STO, SSTRT, and ISTRT as a function of the percentage of malicious nodes in the range of $10\% - 70\%$ for $\omega_R:\omega_U:\omega_D = \frac{1}{3}:\frac{1}{3}:\frac{1}{3}, \frac{1}{2}:\frac{1}{3}:\frac{1}{6}, \frac{1}{2}:\frac{1}{6}:\frac{1}{3}$ and $\frac{1}{3}:\frac{1}{6}:\frac{1}{2}$, respectively. While there is no clear winner among these three trust protocol designs, SSTRT appears to perform

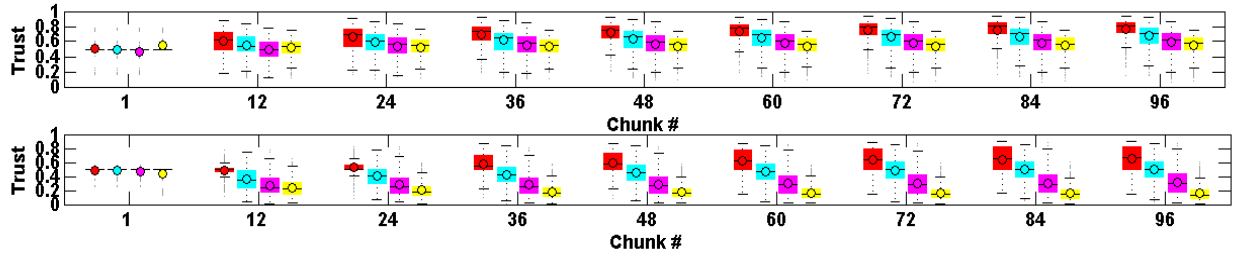
the best in terms of $\overline{R}$ over all weight ratio scenarios. Because of the tradeoff between the multiple goals between $\bar{R}$ vs. $\bar{U}$ and $\bar{D}$, maximizing $\bar{R}$ is often offset by sacrificing $\bar{U}$ and $\bar{D}$. In particular, we observe that under the weight ratio $\omega_R:\omega_U:\omega_D = \frac{1}{2}:\frac{1}{3}:\frac{1}{6}$, SSTRT outperforms STO and ISTRT in P_{MOO} since in this scenario SSTRT can best balance the gain of $\bar{R}$ against the combined loss of $\bar{U}$ and $\bar{D}$ compared with STO and ISTRT.



(a) Trust Values of Good Nodes (top) and Bad Nodes (bottom) under STO.



(b) Trust Values of Good Nodes (top) and Bad Nodes (bottom) under SSTRT.



(c) Trust Values of Good Nodes (top) and Bad Nodes (bottom) under ISTRT.

Figure 6.9: Trust Value Distribution of Good Nodes and Bad Nodes under STO vs. SSTRT and ISTRT.

Figure 6.9 (a), (b), and (c) show the trust values of good nodes (top) and bad nodes (bottom) over time (chunk #) in boxplot format under STO, SSTRT and ISTRT, respectively, with the weight ratio $\omega_R:\omega_U:\omega_D = \frac{1}{2}:\frac{1}{3}:\frac{1}{6}$. We see clearly that SSTRT can best discern good nodes from bad nodes compared with STO and ISTRT. We attribute this to the ability of SSTRT to separate service trust from recommendation trust, which improves trust accuracy.

We conclude that the choice of the best trust protocol design is dictated by the relative importance of $\bar{R}$ vs. $\bar{U}$ and $\bar{D}$, i.e., it is highly sensitive to the weight ratio $\omega_R:\omega_U:\omega_D$ which in turn is dictated by the application requirement. The analysis performed here allows the system designer to choose the best trust protocol design for maximizing task assignment MOO performance.

6.4.4 Performance Comparison of CATrust vs. STO

In this section, we perform a comparative performance analysis of CATrust vs. STO as the underlying trust protocol for the trust-based algorithm for solving the service composition and binding MOO problem.

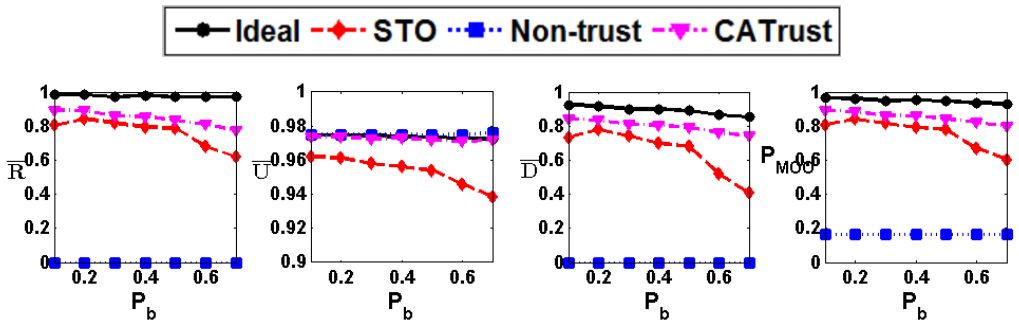


Figure 6.10: CATrust vs. STO in Mission reliability, Load Balance, QoS, and P_{MOO} as a Function of Bad Node Percentage (P_b).

Figure 6.10 compares CATrust with STO in terms of the scaled mission reliability ($\bar{R}$), load balance ($\bar{U}$), delay ($\bar{D}$), and P_{MOO}. The experimental setting is the same as that for Figure 6.3. All nodes have comparable service delay (D) randomly generated. To reflect the influence of the context environment, D is adjusted by the amount of energy required for service (measured by the distance between the TL and the SP) and the incentive (measured by the task importance). A task is satisfactory if it does not fail and the TL is satisfied with the service quality measured by MOO. The TL measures user satisfaction in terms of US_m as in (5.5)-(5.10). If $US_m \geq UST_m$ (the minimum service quality threshold set as 90% in the experiment) the task is considered satisfactory. Otherwise it is unsatisfactory in which case only malicious nodes performing opportunistic attacks will be punished with trust loss. Specifically for CATrust, if $US_m \geq UST_m$, then $s_{TL,j}^t = 1$; otherwise, $s_{TL,j}^t = 0$. For STO, if $US_m \geq UST_m$, then $\alpha = \alpha + \Delta\alpha$; otherwise, $\beta = \beta + \Delta\beta$.

We observe that CATrust outperforms STO in the scaled mission reliability ($\bar{R}$), load balance ($\bar{U}$), delay ($\bar{D}$), and the overall MOO value P_{MOO}. Here the MOO value is the service quality metric to decide if a task is satisfactory. When a task is unsatisfactory, bad nodes are punished with trust loss. These bad nodes must be either

of low service quality causing low user satisfaction, or performing opportunistic attacks causing the task to fail in which case they actually provide unacceptably low service quality causing the task deadline to be missed. These bad nodes are remembered by CATrust and STO differently. STO remembers these low service quality nodes by their identity. CATrust remembers these low service quality nodes by the context conditions (energy, incentive). On the other hand, when a task is satisfactory, the task must have been executed by high service quality nodes (good or bad). Again these nodes (good or bad) are remembered by CATrust and STO differently. STO remembers these high service quality nodes by their identity. So this adds confusion to STO since a bad node sometimes has high service quality and sometimes has low service quality. CATrust remembers these good service quality nodes by the context conditions (energy, incentive). So CATrust has distinctive advantages over STO in associating the specific context environment under which a node can provide good or bad service quality. Consequently, CATrust outperforms STO because of its unique capability to associate good/bad service quality with context.

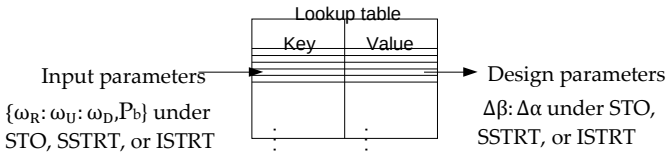


Figure 6.11: Lookup Table Mechanism.

6.5 Applicability

The simulation results obtained reveal the best trust protocol settings in terms of the best $\Delta\beta:\Delta\alpha$ ratio to achieve MOO, given the relative importance of $\bar{R}$ vs. $\bar{U}$ and $\bar{D}$ (which determines $\omega_R:\omega_U:\omega_D$) and the hostility condition (which determines P_b) as input. More specifically, the simulation results obtained can be built into a lookup table, covering a conceivable range of $\omega_R:\omega_U:\omega_D$ and P_b values as input.

The lookup table as shown in Figure 6.11 would store key-value pairs where the "keys" are combinations of input parameter values, and the "values" are the best $\Delta\beta:\Delta\alpha$ ratio for maximizing MOO performance under the input parameter values. The *input parameters* on the left are input to the lookup table at runtime. The *design parameters* on the right are output as a result of a table lookup operation. Upon sensing the environment changes in terms of input parameter values, the system can perform a simple table lookup operation augmented with extrapolation or interpolation techniques to determine and apply the best $\Delta\beta:\Delta\alpha$ ratio in response to dynamically changing conditions. Depending on data granularity, a set of input parameter values

may not directly map to a set of output parameter values. Extrapolation or interpolation techniques may be used to produce the matching output. The lookup time is O(1) and can be efficiently applied at runtime to determine the best trust protocol design as well as the best $\Delta\beta:\Delta\alpha$ ratio for maximizing task allocation MOO performance.

6.6 Summary

In this chapter, we proposed a trust-based dynamic task assignment algorithm for autonomous service-oriented MANET where we are concerned with MOO for multiple objectives with conflicting goals. The results demonstrated that our trust-based solution has low complexity and yet can achieve performance comparable to that of the ideal solution with perfect knowledge of node reliability, and can significantly outperform the non-trust-based solution. We also provided insight of how MOO is achieved by the ideal, trust-based and non-trust-based solutions, and identified the trust protocol parameter settings under which MOO performance is maximized for the trust-based solution which can best balance multiple objectives with conflicting goals. The results obtained are useful for dynamic trust protocol management to maximize application performance in terms of MOO.

Chapter 7

Conclusions and Future Research

In this chapter, we summarize research accomplishments and outline future research areas.

7.1 Publication Summary

We have designed a new trust model for service-oriented MANETs and validated the concept of trust-based service management with two service-oriented MANET applications with MOO requirements. More specifically, we have developed and analyzed a logit regression-based context-aware trust model, CATrust, to estimate dynamic trust based on a node's distinct behavior pattern in response to operational and environmental changes (Chapter 4). We also designed a recommendation filtering mechanism to effectively fend off recommendation attacks. We demonstrated that CATrust outperforms Beta Reputation [69] with belief discounting and Adaptive Trust [31] in terms of false positive probability and false negative probability. We have developed a multi-trust-based algorithm to solve a node-to-service composition and binding MOO problem in service-oriented MANETs (Chapter 5). We demonstrated that our trust-based solutions for solving the MOO problem in MANET environments are efficient and effective without compromising solution optimality when compared with non-trust-based solutions. In particular, CATrust based solutions outperform other trust-based solutions (BRS, TRM, and SRM) based on Bayesian inference. We attribute CATrust's superiority to its ability to more accurately learn an SP's context-dependent service behavior and the delivered service quality during service binding, while BRS, TRM, and SRM do not take context into consideration and merely predict an SP's average delivered service quality across the context space. We have also designed a trust-based algorithm to solve a node-to-task assignment MOO problem in service-oriented MANETs (Chapter 6). We demonstrated that our trust-based solutions approximate the theoretically achievable MOO performance, and outperform

counterpart non-trust-based solutions based on blacklisting. We also demonstrated that CATrust based solutions outperform other trust-based solutions due to its unique context-aware design.

The dissertation work has resulted in two journal publications, four conference publications, and three journal/conference submissions given below. At the end of each paper publication, we give the reference number of the publication and annotate the dissertation chapter in which part of the paper is included.

Journal Publications:

1. Y. Wang, I.R. Chen, J.H. Cho, A. Swami, Y.C. Lu, C.T. Lu, and J.J.P. Tsai, "CATrust: Context-Aware Trust Management for Service-Oriented Ad Hoc Networks," *IEEE Transactions on Service Computing*, *IEEE Transactions on Service Computing*, provisionally accepted to appear, 2016. [138] (This paper is part of Chapter 4.)
2. Y. Wang, I.R. Chen, J.H. Cho, A. Swami, K.S. Chan, "Trust-based Service Composition and Binding with Multiple Objective Optimization in Service-Oriented Mobile Ad Hoc Networks." *IEEE Transactions on Service Computing*, in press, 2016. [137] (The paper is part of Chapter 5.)

Conference Publications:

1. Y. Wang, Y.C. Lu, I.R. Chen, J.H. Cho, A. Swami, C.T. Lu, "LogitTrust: A Logit Regression-based Trust Model for Mobile Ad Hoc Networks," *6th ASE International Conference on Privacy, Security, Risk and Trust*, Dec. 2014. [144] (The paper is part of Chapter 4.)
2. Y. Wang, I.R. Chen, J.H. Cho, K.S. Chan, and A. Swami, "Trust-based Service Composition and Binding for Tactical Networks with Multiple Objectives," *32th IEEE Military Communications Conference*, pp. 1862-1867, Nov. 2013. [136] (The paper is part of Chapter 5.)
3. J.H. Cho, I.R. Chen, Y. Wang, K.S. Chan, and A. Swami, "Multi-Objective Optimization for Trustworthy Tactical Networks: A Survey and Insight," *18th International Command and Control Research and Technology Symposium*, June 2013. [46] (The paper is part of Chapter 2.)
4. J.H. Cho, I.R. Chen, Y. Wang, and K.S. Chan, "Trust-based Multi-Objective Optimization for Node-to-Task Assignment in Coalition Networks," *19th IEEE International Conference on Parallel and Distributed Systems*, pp. 372-279, Dec. 2013. [45] (The paper is part of Chapter 6.)

Papers Submitted:

1. Y. Wang, I.R. Chen, J.H. Cho, and J.J.P. Tsai, "Trust-based Task Assignment with Multi-Objective Optimization in Service-Oriented Ad Hoc Networks," *IEEE Transactions on Network and Service Management,* submitted in November 2015, revised April 2016. (The paper is part of Chapter 6.)
2. J.H. Cho, Y. Wang, I.R. Chen, K.S. Chan, and A. Swami, "Multi-Objective Optimization in Coalition Formation: A Survey," *ACM Computing Surveys,* submitted in August 2014. (The paper is part of Chapter 2.)
3. Y. Wang, I.R. Chen, J.H. Cho, and J.J.P. Tsai, "A Comparative Analysis of Trust-based Service Composition Algorithms in Mobile Ad Hoc Networks," *IEEE Globecom 2016,* submitted in April 2016. (The paper is part of Chapter 2.)

7.2 Future Research Directions

There are some future possible research directions that can be extended from the dissertation research.

7.2.1 Context Variable Selection

The effectiveness of CATrust hinges on context variable selection that can effectively capture the effect of context on a node's service behavior and quality. A future research direction is to apply feature selection approaches in the field of data mining to select context variables, apply correlation and dimension reduction techniques to reduce the context variable set to a minimum but essential set, and analyze the effect on CATrust performance compared with existing context-aware trust protocols [83] [86] [108] [110] [113] [130] [142] [152] [159].

7.2.2 Robustness of CATrust against Environment Noise and Node Social Behavior

A future research direction is to conduct more sensitivity analyses of the prediction accuracy of CATrust against noisy, erroneous, incomplete, and uncertain MANET environments. This dissertation research considered malicious recommenders reporting false service quality observations. This research also only considered that CATrust can tolerate conflicting behavior attack because the relationship between an SR and an SP determines the SP's service behavior toward the SR and CATrust can precisely capture such service behavior based on SR-SP pairing. A future research direction is to further verify the robustness of CATrust against attacks derived from node social behavior such as recommendation attacks and conflicting behavior attacks. A possibility is to

separate nodes into social groups and prove that CATrust treating social relationships as context variables is efficient and effective against social behavior based attacks.

Our CATrust design so far is able to address "uncertainty" by considering the association of a node's service quality with the context environment in which the service is provided. It also treats trust as the "willingness to take risk" as it considers service quality under the influence of context as the sole criterion for rating an SP even if the SP selected for service is malicious for its own benefits. One thing we do not consider in our CATrust design is "subjectivity", i.e., user satisfaction is not subjective and service quality is not subjective. A future research direction is to treat "subjectivity" as context based on SR-SP pairing and extend CATrust's applicability to MANET applications for which "subjectivity" on service quality or user satisfaction is important.

7.2.3 Validity of CATrust for Service-Oriented MANET Applications

Another future direction is to validate CATrust with real-world applications or data traces in service-oriented MANETs. CATrust is a trust model based on the design concept that trust computation is context-ware. The dissertation research has applied CATrust to service composition and task allocation MANET applications with MOO goals based on synthesized data to demonstrate its applicability, treating the MOO function as an indicator of service quality. However the input service record data was synthesized based on certain assumptions regarding the characteristics of context variables (discussed in Section 4.1.6). A future direction is to discover data traces from real-world MANET applications that can validate these assumptions. Our preliminary work conducted in Section 4.4 can be further extended. The challenge lies in discovering the relationships between context variables and service behaviors for nodes in a service-oriented MANET application because the relationships between context variables and service behaviors are application dependent. Also a possible future research direction is to examine the effect of matching linear/nonlinear CATrust design with the linear/non-linear relationship between context and service quality on CATrust performance.

In the dissertation research, we only applied CATrust to two applications. A future direction is to discover applications in service-oriented MANETs that can benefit from the design concept of context-aware trust management developed in the dissertation research. This includes, but not limited to trust-based admission control strategies as in [35] [36] [38] [43] [132] [153] for an SP to determine which service requests to accept for service to maximize its payoff, trust-based location service management [34] [35] [60] [80] [81] [82] for a lightweight node to determine which cloud service computing/location server [139] to select for task offloading to maximize its performance, trust-based detection [7] [8] [9] for a node to assess trustworthiness of

neighbor nodes for MANET routing decision making, and trust-enhanced intrusion detection for detecting complex context-dependent attack behaviors as considered in [96] [97] [98] [99] [100] [102] for a service-oriented MANET system with fuzzy failure criteria [18] [32] [33]. One may utilize more sophisticated modeling tool such as Stochastic Petri Nets [39] [40] for performing the analysis.

7.2.4 Generalization of CATrust Design

Another future research direction is to generalize CATrust design to handle more general cases. One case is to deal with the service quality grading score being in a range, e.g., 1-5, instead of just a binary number (0/1). One possible solution is that instead of outputting a single trust value predicting if the service score is 1, CATrust would output a set of trust values, one for each service score in the range. Logit regression would still be used at the expense of more computational time. Another case is to extend CATrust to deal with more sophisticated attack behaviors such as opportunistic, collusion, and insidious attacks [92] [93] such that malicious nodes may not perform persistent attacks all the time but rather can attack opportunistically. A possible solution is to represent the condition under which an opportunistic service attack or a time-varying attack will perform as a context-service quality relationship for CATrust to learn so as to cope with time-varying attacks gracefully. A third case is to consider other MOO formulations other than weighted sum, such as ε-constraints, goal programming, and min-max method, when applying CATrust to service-oriented MANET applications with MOO goals.

Bibliography

[1] I. Aad, J.-P. Hubaux, and E.W. Knightly, "Impact of Denail of Service Attacks on Ad Hoc Networks," *IEEE/ACM Trans. Networking*, vol. 16, no. 4, 2008.

[2] K. Aberer and Z. Despotovic, "Managing Trust in a Peer-2-Peer Information System," in *Proc. 10th Int'l. Conf. Information and Knowledge Management*, 2001, pp. 310-317.

[3] Air Land Sea Application Center, *Tactical Convoy Ops: Multi-Service Tactics, Techniques, and Procedures for Tactical Convoy Operations*, March 2005.

[4] E. Aivaloglou and S. Gritzalis, "Trust–Based Data Disclosure in Sensor Networks," in *IEEE Int'l. Conf. Communications*, 2009.

[5] P. Albers et al., "Security in Ad Hoc Networks: A General Intrusion Detection Architecture Enhancing Trust Based Approaches," *Wireless Information Systems*, pp. 1-12, 2002.

[6] D.S. Alberts and R.E. Hayes, "Power to the Edge: Command and Control in the Information Age," in *Information Age Transformation Series, Command and Control Research Program (CCRP).*, 2003.

[7] H. Al-Hamadi and I.R. Chen, "Adaptive Network Management of Countering Smart Attack and Selective Capture in Wireless Sensor Networks," *IEEE Trans Network and Service Management*, vol. 12, no. 3, pp. 451-466, 2015.

[8] H. Al-Hamadi and I.R. Chen, "Integrated Intrusion Detection and Tolerence in Homogeneous Clustered Sensor Networks," *ACM Trans Sensor Networks*, vol. 11, no. 3, 2015.

[9] H. Al-Hamadi and I.R. Chen, "Redundancy Management of Multipath Routing for

Intrusion Tolerance in Heterogeneous Wireless Sensor Networks," *IEEE Trans. Network and Service Management*, vol. 19, no. 2, pp. 189-203, 2013.

[10] M. Alrifai and T. Risse, "Combining Global Optimization with Local Selection for Efficient QoS-Aware Service Composition," in *Proc. 18th Int'l Conf. World Wide Web*, 2009, pp. 881-990.

[11] A. Anagnostopoulos, L. Becchetti, C. Castillo, A. Gionis, and S. Leonardi, "Online Team Formation in Social Networks," in *21st Int. Conf. World Wide Web*, 2012, pp. 839-848.

[12] M.A. Ayachi, C. Bidan, T. Abbes, and A. Bouhoula, "Misbehavior Detection Using Implicit Trust Relations in the AODV Routing Protocol," in *Proc. 2009 Int'l. Conf. Computational Science and Engineering*, 2009.

[13] V. Balakrishnan, V. Varadharajan, and U. Tupakula, "Subjective Logic Based Trust Model for Mobile Ad hoc Networks," in *The 4th International Conference on Security and Privacy in Communication Networks*, 2008, pp. 1-11.

[14] J. Balicki, "An Adaptive Quantum-Based Multiobjective Evolutionary Algorithm," in *World Scientific and Engineering Academy and Society (WSEAS) 13th Int. Conf. Computers*, 2009, pp. 417-422.

[15] S.K. Bansal, A. Bansal, and M.B. Blake, "Trust-based dynamic web service composition using social network analysis," in *Proc. 2010 IEEE Int'l Workshop Business Applications of Social Network Analysis*, 2010.

[16] F. Bao and I.R. Chen, "Dynamic Trust Management for Internet of Things Applications," in *Proc. 2012 Int'l. Workshop Self-Aware Internet of Things*, 2012.

[17] F. Bao, I.R. Chen, M. Chang, and J.H. Cho, "Trust-based Intrusion Detection in Wireless Sensor Networks," in *IEEE Int. Conf. Computer Communications*, 2011, pp. 1-6.

[18] F.B. Bastani, I.R. Chen, and T. Tsao, "Reliability of Systems with Fuzzy-Failure Criterion," *Annual Reliability and Maintainability Symposium*, 1994.

[19] M. Blaze, J. Feigenbaum, J. Ioannidis, and A.D. Keromytis, "The Role of Trust

Management in Distributed System Security," in *Security Internet Programming*, J. Vitek and C.D. Jensen, Eds.: Springer Berlin Heidelberg, 1999, pp. 185-210.

[20] M. Blaze, J. Feigenbaum, and J. Lacy, "Decentralized Trust Management," in *Proc. 1996 IEEE Conf. Security and Privacy*, 1996, pp. 164-173.

[21] E. Borgia, "The Internet of Things Vision: Key Features, Applications and Open Issues," *Computer Communications*, vol. 54, pp. 1-31, December 2014.

[22] A. Boukerch, L. Xu, and K. E-Khatib, "Trust-Based Security for Wireless Ad Hoc and Sensor Networks," *Computer Communications*, no. 30, pp. 2413-2427, 2007.

[23] S. Boyd and L. Vandenberghe, *Convex Optimization*.: Cambridge university press, 2004.

[24] M.R. Brust, C. Ribeiro, D. Turgut, and S. Rothkugel, "LSWTC: A Local Small-World Topology Control Algorithm for Backbone-Assisted Mobile Ad hoc Networks," in *IEEE Conf. Local Computer Networks*, 2010, pp. 144-151.

[25] S. Buchegger and J.Y. Le-Boudec, "A Robust Reputation System for Peer-to-Peer and Mobile Ad-hoc Networks," in *Proc. 2nd Workshop on Economics Peer-to-Peer Systems*, 2004.

[26] S. Buchegger and J.Y. Le-Boudec, "Performance Analysis of The CONFIDANT Protocol: Cooperation of Nodes - Fairness in Dynamic Ad-hoc Networks," in *Proc. 3rd ACM Int'l. Symp. Mobile Ad Hoc Networking and Computing*, 2002, pp. 226-236.

[27] V. Cahill, "Using Trust for Secure Collaboration in Uncertain Environments," in *Proc. 2003 IEEE Pervasive Computing*, 2003, pp. 52-61.

[28] D. Chakraborty, Y. Yesha, and A. Joshi, "A Distributed Service Composition Protocol for Pervasive Environments," in *IEEE Wireless Communications and Networking Conf.*, 2004, pp. 2575-2580.

[29] I.R. Chen, F. Bao, M. Chang, and J.H. Cho, "Dynamic Trust Management for Delay Tolerant Networks and Its Application to Secure Routing," *IEEE Trans. Parallel Delay Tolerant Networks*, vol. 25, no. 5, pp. 1200-1210, 2014.

[30] I.R. Chen, F. Bao, M. Chang, and J.H. Cho, "Trust Management for Encounter-Based Routing in Delay Tolerant Networks," in *2010 IEEE Global Commun. Conf.*, 2010.

[31] I.R. Chen, F Bao, and J. Guo, "Trust-based Service Management for Social Internet of Things Systems," *IEEE Trans Dependable and Secure Computing*, 2016.

[32] I.R. Chen and F.B. Bastani, "Effect of Artificial-Intelligence Planning Procedures on System Reliability," *IEEE Trans Reliability*, vol. 40, no. 3, pp. 364-369, 1991.

[33] I.R. Chen, F.B. Bastani, and T.W. Tsao, "On the Reliability of AI Planning Software in Real-Time Applications," *IEEE Trans Knowledge and Data Engineering*, vol. 7, no. 1, pp. 4-13, 1995.

[34] I.R. Chen, T.M. Chen, and C. Lee, "Agent-Based Forwarding Strategies for Reducing Location Management Cost in Mobile Networks," *Mobile Networks and Applications*, vol. 6, no. 2, pp. 105-115, 2001.

[35] I.R. Chen, T.M. Chen, and C. Lee, "Performance Evaluation of Forwarding Strategies for Location Management in Mobile Networks," *The Computer Journal*, vol. 41, no. 4, pp. 243-253, 1998.

[36] S.T. Cheng, C.M. Chen, and I.R. Chen, "Dynamic Quota-Based Admission Control with Sub-Rating in Multimedia Servers," *Multimedia Systems*, vol. 8, no. 2, 2000.

[37] I.R. Chen, J. Guo, F. Bao, and J.H. Cho, "Trust Management in Mobile Ad Hoc Networks for Bias Minimization and Application Performance Maximization," *Ad Hoc Networks*, vol. 19, pp. 59-74, 2014.

[38] I.R. Chen and T.H. Hsi, "Performance Analysis of Admission Control Algorithms Based on Reward Optimization for Real-Time Multimedia Servers," *Performance Evaluation*, vol. 33, no. 2, pp. 89-112, 1998.

[39] I.R. Chen and D.C. Wang, "Analysis of Replicated Data with Repair Dependency," *The Computer Journal*, vol. 39, no. 9, pp. 767-779, 1996.

[40] I.R. Chen and D.C. Wang, "Analyzing Dynamic Voting Using Petri Nets," *IEEE 15th Symposium Reliable Distributed Systems*, pp. 44-53, 1996.

[41] I.R. Chen and Y. Wang, "Reliability Analysis of Wireless Sensor Networks with Distributed Code Attestation," *IEEE Communications Letters*, vol. 16, no. 10, pp. 1640-1643, 2012.

[42] J. Chen, X. Yan, H. Chen, and D. Sun, "Resource Constrained Multirobot Task Allocation with A Leader-Follower Coalition Method," in *IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2010, pp. 5093 - 5098.

[43] I.R. Chen, O. Yilmaz, and I. Yen, "Admission Control Algorithms for Revenue Optimization with QoS Guarantees in Mobile Wireless Networks," *Wireless Personal Communications*, vol. 38, no. 3, pp. 357-376, 2006.

[44] J.H. Cho, I.R. Chen, and P.G. Feng, "Effect of Intrusion Detection on Reliability of Mission-Oriented Mobile Group Systems in Mobile Ad Hoc Networks," *IEEE Trans. Reliability*, vol. 59, no. 1, pp. 231-241, 2010.

[45] J.H. Cho, I.R. Chen, Y. Wang, and K.S. Chan, "Trust-Based Multi-Objective Optimization for Node-to-Task Assignment in Coalition Networks," in *Proc. 19th IEEE Int'l. Conf. Parallel and Distributed Systems*, 2013.

[46] J.H. Cho, I.R. Chen, Y. Wang, K.S. Chan, and A. Swami, "Multi-Objective Optimization for Trustworthy Tactical Networks: A Survey and Insight," in *18th Int'l Command and Control Research and Technology Symposium*, 2013.

[47] J.H. Cho, A. Swami, and I.R. Chen, "A Survey on Trust Management for Mobile Ad Hoc Networks," *IEEE Comm. Surveys and Tutorials*, vol. 13, no. 4, pp. 562-583, 2011.

[48] J.H. Cho, A. Swami, and I.R. Chen, "Modeling and Analysis of Trust Management for Cognitive Mission-Driven Group Communication Systems in Mobile Ad Hoc Networks," in *Int'l. Conf. Computational Science and Engineering*, 2009, pp. 641-650.

[49] J.H. Cho, A. Swami, and I.R. Chen, "Modeling and Analysis of Trust Management with Trust Chain Optimization in Mobile Ad Hoc Networks," *Journal of Network and Computer Applications*, vol. 35, no. 3, pp. 1001-1012, May 2012.

[50] G. Dai and Y. Wang, "Trust-Aware Component Service Selection Algorithm in Service Compositio," in *Proc. 4th Int'l. Conf. Frontier of Computer Science and*

Technology, 2009.

[51] C.R. Davis, "A Localized Trust Management Scheme for Ad Hoc Networks," in *Proc. 3rd Int'l. Conf. Networking*, 2004, pp. 671-675.

[52] E. Edalat, C. Than, and W. Xiao, "An Auction-Based Strategy for Distributed Task Allocation in Wireless Sensor Networks," *Computer Communications*, vol. 35, no. 8, pp. 916-928, 2012.

[53] M.T. El-Melegy, M.H. Essai, and A.A. Ali, "Robust Training of Artificial Feedforward Neural Networks," in *Fondations of Computational Intelligence Volume 1*, A.-E. Hassanien et al., Eds.: Springer Berlin Heidelberg, 2009, ch. 9, pp. 217-242.

[54] S. Fruhwirth-Schnatter and R. Fruhwirth, "Bayesian Inference in the Multinomial Logit Model," *Austrian Journal of Statistics*, vol. 41, no. 1, pp. 27-43, 2012.

[55] S. Ganeriwal, L.K. Balzano, and M.B. Srivastava, "Reputation-Based Framework for High Integrity Sensor Networks," *ACM Trans. Sensor Networks*, vol. 4, no. 3, 2008.

[56] H. Gao et al., "A Survey of Incentive Mechanisms for Participatory Sensing," *IEEE Commun. Surveys & Tutorials*, vol. 17, no. 2, pp. 918-943, 2015.

[57] M. R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*.: W.H. Freeman & Co., 1979.

[58] C. Glaßer, C. Reitwießner, H. Schmitz, and M. Witek, "Approximability and Hardness in Multi-Objective Optimization," in *Programs, Proofs, Processes*.: Springer, 2010, pp. 180-189.

[59] K. Govindan and P. Mohapatra, "Trust Computations and Trust Dynamics in Mobile Adhoc Networks: A Survey," *IEEE Comm. Survey and Tutorials*, vol. 14, no. 2, pp. 279-298, 2012.

[60] B. Gu and I.R. Chen, "Performance Analysis of Location-Aware Mobile Service Proxies for Reducing Network Cost in Personal Communication Systems," *Mobile Networks and Applications*, vol. 10, no. 4, pp. 453-463, 2005.

[61] L. Guo, G. Shao, and S. Zhao, "Multi-Objective Task Assignment in Cloud Computing by Particle Swarm Optimization," in *8th Int. Conf. Wireless Communications, Networking and Mobile Computing*, 2012, pp. 1 - 4.

[62] C.W. Hang, A.K. Kalia, and M.P. Singh, "Behind the Curtain: Service Selection via Trust in Composite Services," in *Proc. 10th Int'l. Conf. Web Services*, 2012, pp. 9-16.

[63] C.W. Hang and M.P. Singh, "Trustworthy Service Selection and Composition," *ACM Trans. Autonomous and Adaptive Systems*, vol. 6, no. 1, 2011.

[64] F.T. Johnsen, J. Flathagen, and T. Hafsoe, "Pervasive Service Discovery Across Heterogeneous Tactical Networks," in *IEEE 2009 Military Comm. Conf.*, 2009, pp. 1-8.

[65] F.T. Johnsen, M. Rustad, T. Hafsoe, A. Eggen, and T. Gagnes, "Semantic Service Discovery for Interoperability in Tactial Military Networks," *The Int'l C2 J.*, vol. 4, no. 1, 2010.

[66] D.B. Johnson, D.A. Maltz, and J. Broch, "DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks," in *Ad Hoc Networking*, C.E. Perkins, Ed.: Addison-Wesley, 2001, ch. 5, pp. 139-172.

[67] A. Jøsang, "A Logic for Uncertain Probabilities," *International Journal of Uncertainty, Fuzziness and Knowledge-based Systems*, vol. 9, no. 3, pp. 279-311, 2001.

[68] A. Jøsang and J. Haller, "Dirichlet Reputation Systems," in *Proc. 2nd Int'l. Conf. Availability, Reliability and Security*, 2007, pp. 112-119.

[69] A. Jøsang and R. Ismail, "The Beta Reputation System," in *Proc. 15th Bled Electronic Commerce Conf.*, 2002, pp. 1-14.

[70] A. Jøsang, R. Ismail, and C. Boyd, "A Survey of Trust and Reputation Systems for Online Service Provision," *Decision Support Systems*, vol. 43, no. 2, pp. 618-644, 2007.

[71] S. Kalasapur, M. Kumar, and B.A. Shirazi, "Dynamic Service Composition in Pervasive Computing," *IEEE Trans. Parallel and Distributed Systems*, pp. 1997-2009, 2012.

[72] S.D. Kamvar, M.T. Schlosser, and H. Garcia-Molina, "The Eigentrust Algorithm for Reputation Management in P2P Networks," in *2003 World Wide Web Conf.*, 2003.

[73] E. Karmouch and A. Nayak, "A Distributed Constraint Satisfaction Problem Approach to Virtual Device Composition," *IEEE Trans. Parallel and Distributed Systems*, vol. 23, no. 11, pp. 1997-2009, 2012.

[74] L.M. Kaufman, "Data Security in the World of Cloud Computing," *IEEE Security & Privacy*, vol. 7, no. 4, pp. 61-64, 2009.

[75] B. Khosravifar, J. Bentahar, and A. Moazin, "Analyzing the Relations between Some Parameters of Web Services Reputation," in *IEEE 2010 Int'l Conf. Web Services*, 2010, pp. 329-336.

[76] S. Kosta, A. Mei, and J. Stefa, "Large-Scale Synthetic Social Mobile Networks with SWIM," *IEEE Trans. Mobile Computing*, vol. 13, no. 1, pp. 116-129, 2014.

[77] S. Kosta, A. Mei, and J. Stefa, "Small World in Motion (SWIM): Modeling Communities in Ad-Hoc Mobile Networking," in *7th IEEE Communication Society Conf. Sensor, Mesh and Ad Hoc Communications and Networks*, 2010.

[78] L. Lamport, R. Shostak, and M. Pease, "The Byzantine Generals Problem," *ACM Trans. Program. Lang. Syst.*, vol. 4, no. 3, pp. 382-401, 1982.

[79] S. Lee, R. Sherwood, and B. Bhattacharjee, "Cooperative Peer Groups in NICE," in *Proc. 22nd Annual Joint Conf. IEEE Computer and Communications*, 2003, pp. 1272-1282.

[80] Y. Li and I.R. Chen, "Adaptive Per-User Per-Object Cache Consistency Management for Mobile Data Access in Wireless Mesh Networks," *J. Parallel and Distributed Computing*, vol. 71, pp. 1034-1046, 2011.

[81] Y. Li and I.R. Chen, "Design and Performance Analysis of Mobility Management Schemes based on Pointer Forwarding for Wireless Mesh Networks," *IEEE Trans Mobile Computing*, vol. 10, no. 3, pp. 349-361, 2011.

[82] Y. Li and I.R. Chen, "Mobility Management in Wireless Mesh Networks Utilizing Location Routing and Pointer Forwarding," *IEEE Trans Network and Service*

Management, vol. 9, no. 3, pp. 226-239, 2012.

[83] W. Li, A. Joshi, and T. Finin, "CAST: Context-Aware Security and Trust Framework for Mobile Ad-hoc Networks Using Policies," *Distributed and Parallel Databases*, vol. 31, no. 2, pp. 353-376, June 2013.

[84] Z. Li, X. Li, V. Narasimhan, A Nayak, and I Stojmenovic, "Autoregression Models for Trust Management in Wireless Ad Hoc Networks," in *IEEE Global Telecommunications Conference*, 2011, pp. 1-5.

[85] C. Liu, "Robit Regression: A Simple Robust Alternative to Logistic and," in *Applied Bayesian Modeling and Causal Inference*, A. Gelman and X. L. Meng, Eds. London: Wiley, 2004, ch. 21.

[86] X Liu and A. Datta, "Modeling Context Aware Dynamic Trust Using Hidden Markov Model," in *26th AAAI Conference on Artificial Intelligence*, 2012, pp. 1938-1944.

[87] C.H. Liu, J. Fan, P. Hui, J. Wu, and K.K. Leung, "Towards QoI and Energy-Efficiency in Participatory Crowdsourcing," *IEEE Trans. Vehicular Technology*, vol. 64, no. 10, pp. 2684-4700, 2015.

[88] F. Li and J. Wu, "Mobility Reduces Uncertainty in MANETS," in *Proc. 26th IEEE Int'l. Conf. Computer Communications*, 2007, pp. 1946-1954.

[89] M. Mathew and N. Weng, "Quality of Information and Energy Efficiency Optimization for Sensor Networks via Adaptive Sensing and Transmitting," *IEEE Sensors Journal*, vol. 14, no. 2, pp. 341-348, 2014.

[90] D.H McKnight and N.L. Chervany, "The Meanings of Trust," University of Minnesota, Tech. Rep 1996.

[91] D.H. McKnight and N.L. Chervany, "What Trust Means in E-Commerce Customer Relationships: An Interdisciplinary Conceptual Typology," *Int'l. J. Electronic Commerce*, vol. 6, pp. 35-59, 2002.

[92] M. Mehdi, N. Bougnuila, and J. Bentahar, "Correlated Multi-Dimensional QoS Metrics for Trust Evaluation Within Web Services," in *AAMAS*, 2014, pp. 1605-

1606.

[93] M. Mehdi, N. Bouguila, and J. Bentabar, "Probabilistic Approach for QoS-Aware Recommender System for Trustworthy Web Service Selection," *Applied Intelligence*, vol. 41, no. 2, pp. 503-524, 2014.

[94] M. Mehdi, N. Bouguila, and J. Bentahar, "QoS-Based Reputation Feedback Fusion under Unknown Correlation," in *Adaptive and Intelligent Systems*.: Springer, 2014, pp. 172-181.

[95] P. Michiardi and R. Molva, "CORE: A Collaborative Reputation Mechanism to Enforce Node Cooperation in Mobile Ad Hoc Networks," *Advanced Comm. & Multimedia Security*, pp. 107-121, 2002.

[96] R. Mitchell and I.R. Chen, "A Survey of Intrusion Detection in Wireless Network Applications," *Computer Communications*, vol. 42, pp. 1-23, 2014.

[97] R. Mitchell and I.R. Chen, "A Survey of Intrusion Detection Techniques in Cyber Physical Systems," *ACM Computing Survey*, vol. 46, no. 4, 2014.

[98] R. Mitchell and I.R. Chen, "Adaptive Intrusion Detection for Unmanned Aircraft Systems based on Behavior Rule Specifications," *IEEE Trans Systems, Man and Cybernetics*, vol. 12, no. 1, pp. 593-604, 2014.

[99] R. Mitchell and I.R. Chen, "Behavior Rule Based Intrusion Detection Systems for Sasfety Critical Smart Grid Applications," *IEEE Trans. Smart Grid*, vol. 4, no. 3, pp. 1254-1263, 2013.

[100] R. Mitchell and I.R. Chen, "Behavior Rule Specification-based Detection for Safety Critical Medical Cyber Physical Systems," *IEEE Trans Dependable and Secure Computing*, vol. 12, no. 1, pp. 16-30, 2015.

[101] R. Mitchell and I.R. Chen, "Effect of Intrusion Detection and Response on Reliability of Cyber Physical Systems," *IEEE Trans. Reliability*, vol. 62, no. 1, pp. 199-210, 2003.

[102] R. Mitchell and I.R. Chen, "Modeling and Analysis of Attacks and Counter Defense Mechanisms for Cyber Physical Systems," *IEEE Trans Reliability*, 2016.

[103] V. Natarajan, S. Zhu, M. Srivatsa, and J. Opper, "Semantics-Aware Storage and Replication of Trust Metadata in Mobile Ad-Hoc Networks," in *Proc. IEEE 26th Int'l. Conf. Advanced Information Networking and Applications*, 2012, pp. 376-383.

[104] S. Pearson and A. Benameur, "Privacy, Security, and Trust Issues Arising from Cloud Computing," in *Proc. 2nd IEEE Int'l. Conf. Cloud Computing Technology and Science*, 2010, pp. 693-702.

[105] A.A. Pirzada, A. Datta, and C. McDonald, "Propagating Trust in Ad-hoc Networks for Reliable Routing," in *Proc 2014 Int'l. Workshop Wireless Ad-Hoc Networks*, 2004, pp. 58-62.

[106] A.A. Pirzada and C. McDonald, "Establishing Trust in Pure Ad-hoc Networks," in *Proc. 27th Australasian Conf. Computer Science*, 2004.

[107] J. Rao and X. Su, "A Survey of Automated Web Service Composition Methods," *LNCS*, vol. 3387, pp. 43-54, 2005.

[108] S. Ries, "Extending Bayesian Trust Models Regarding Context-Dependence and User Friendly Representation," in *ACM Symposium Applied Computing*, 2009, pp. 1294-1301.

[109] V. Roy, "Convergence Rates for MCMC Algorithms for a Robust Bayesian Binary Regression Model," *Electronic Journal of Statistics*, vol. 6, pp. 2463-2485, 2012.

[110] Y.B. Saied, A. Olivereau, D. Zeghlache, and M. Laurent, "Trust Management System Design for the Internet of Things: A Context-Aware and Multi-Service Approach," *Computer & Security*, no. 39, pp. 351-365, 2013.

[111] C. Sandionigi, D. Ardagna, G. Cugola, and C. Ghezzi, "Optimizing Service Selection and Allocation in Situational Computing Applications," *IEEE Trans. Services Computing*, vol. 6, no. 3, pp. 414-428, 2013.

[112] S. Sariel-Talay, T.R. Balch, and N. Erdogan, "A Generic Framework for Distributed Multirobot Cooperation," *J. of Intelligent and Robotic Systems*, vol. 63, no. 2, pp. 323-358, 2011.

[113] R. Shankaran, V. Varadharajan, M.A. Orgun, and M. Hitchens, "Context-Aware

Trust Management of Peer-to-Peer Mobile Ad-Hoc Networks," in *33rd Annual IEEE Int'l Computer Software and Applications Conference*, 2009, pp. 188-193.

[114] L. Shen, L. Zhang, and D. Huang, "Trust-Driven Both-Matched Algorithm for Grid Task Multi-Objective Scheduling," in *2nd Int. Conf. Information Science and Engineering*, 2010, pp. 1661–1664.

[115] E. Shi and A. Perrig, "Designing Secure Sensor Networks," *IEEE Wireless Comm.*, vol. 11, no. 6, pp. 38-53, 2004.

[116] M.P. Singh, "Trustworthy Service Composition: Challenges and Research Questions," *LNCS*, vol. 2631, pp. 39-52, 2003.

[117] A. Srinivasan, J. Teitelbaum, and J. Wu, "DRBTS: Distributed Reputation-Based Beacon Trust System," in *Proc. 2nd IEEE Int'l. Symp. Dependable, Autonomic and Secure Computing*, 2006, pp. 277-283.

[118] B. Srivastava and J. Koehler, "Web Service Composition - Current Solutions and Open Problems," in *Proc. 13rd Int'l Conf. Automated Planning and Scheduling Workshop Planning for Web Service*, 2003, pp. 28-35.

[119] A. Strunk, "QoS-Aware Service Composition: A Survey," in *Proc. 8th European Conference on Web Services*, 2010, pp. 67-74.

[120] Z. Su, L. Liu, M. Li, X. Fan, and Y. Zhou, "ServiceTrust: Trust Management in Service Provision Networks.," in *IEEE International Conference on Service*, 2013, pp. 272-279.

[121] P. Sun, "Service Composition and Optimal Selection with Trust Constraints," in *Proc. 2010 IEEE Asia-Pacific Services Computing Conf.*, 2010, pp. 645-653.

[122] Y. Sun, Z. Han, W. Yu, and K. Liu, "A Trust Evaluation Framework in Distributed Networks: Vulnerability Analysis and Defense against Attacks," in *25th IEEE International Conference on Computer Communications*, 2006, pp. 1-13.

[123] Y.L. Sun, W. Yu, Z. Han, and K.R. Liu, "Information Theoretic Framework of Trust Modeling and Evaluation for Ad Hoc Networks," *IEEE J. Selected Areas in Communications*, vol. 24, no. 2, pp. 305-317, 2006.

[124] N. Suri, "Dynamic Service-Oriented Architectures for Tactical Edge Networks," in *The 4th Workshop Emerging Web Service Technology*, 2009, pp. 3-10.

[125] C. Szabo and T. Kroeger, "Evolving multi-objective strategies for task allocation of scientific workflows on public clouds," in *IEEE World Congress on Computational Intelligence*, 2012, pp. 1-8.

[126] F. Tao, "Resource Service Composition and Its Optimal Selection Based on Particle Swarm Optimization in Manufacturing Grid System," *IEEE Trans. Industrial Informatics*, vol. 4, no. 4, pp. 311-327, 2008.

[127] D.J. Thornley, R. Young, and J. Richardson, "From Mission Specification to Quality of Information Measures Closing the Loop in Military Sensor Networks," in *Proc. Annual Conf. Int'l Technology Alliance*, 2008, pp. 1-2.

[128] A. Tolmidis and L. Petron, "Multi-Objective Optimization for Dynamic Task Allocation in a Multi-Robot System," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 5-6, pp. 1458-1468, May/June 2013.

[129] A. Twigg, "A Subjective Approach to Routing in P2P and Ad Hoc Networks," in *Trust Management*, Paddy Nixon and Sotirios Terzis, Eds.: Springer Berlin Heidelberg, 2003, pp. 225-238.

[130] M.G. Uddin, M. Zulkernine, and S.I. Ahamed, "CAT: A Context-Aware Trust Model for Open and Dynamic Systems," in *ACM Symposium Applied Computing*, 2008, pp. 2024-2029.

[131] R. Venkataraman, M. Pushpalatha, and T. Rama Rao, "Regression-based trust model for mobile ad hoc networks," *Information Security, IET*, vol. 6, no. 3, pp. 131-140, September 2012.

[132] N. Verma and I.R. Chen, "Admission Control Algorithms Integrated with Pricing for Revenue Optimization with QoS Guarantees in Mobile Wireless Networks," in *IEEE 10th Int'l Conf. Parallel and Distributed Systems*, July 2004, pp. 495-502.

[133] R.R.S. Verma, D. O'Mahony, and H. Tewari, "NTM - Progressive Trust Negotiation in Ad Hoc Networks," in *Proc. 1st Joint IEI/IEEE Symp.*

Telecommunications Systems Research, 2001.

[134] F. Wagner, B. Kl, F. Ishikawa, and S. Honiden, "Towards Robust Service Compositions in The Context of Functionally Diverse Services," in *Proc. 21st Int'l Conf. World Wide Web*, 2012, pp. 969-978.

[135] O.A. Wahab, J. Bentahar, H. Otrok, and A. Mourad, "A Survey on Trust and Reputation Models for Web Services: Single, Composite, and Communities," *Decision Support Systems*, vol. 74, pp. 121-134, 2015.

[136] Y. Wang, I.R. Chen, J.H. Cho, K.S. Chan, and A. Swami, "A Trust-Based Service Composition and Binding for Tactical Networks with Multiple Objectives," in *Proc. 32th IEEE Military Comm. Conf.*, 2013.

[137] Y. Wang, I.R. Chen, J.H. Cho, A. Swami, and K.S. Chan, "Trust-based Service Composition and Binding with Multiple Objective Optimization in Service-Oriented Ad Hoc Networks," *IEEE Trans. Services Computing, in press*, p. in press, 2016.

[138] Y. Wang et al., "CATrust: Context-Aware Trust Management for Service-Oriented Ad Hoc Networks," *IEEE Trans. Service Computing, provisionally accepted to appear*, 2016.

[139] Y. Wang, I.R. Chen, and D.C. Wang, "A Survey of Mobile Cloud Computing Applications: Perspectives and Challenges," *Wireless Personal Communications*, vol. 80, no. 4, pp. 1607-1623, 2015.

[140] X. Wang et al., "Trust and Independence Aware Decision Fusion in Distributed Networks," in *Proc. 2013 IEEE Int'l. Conf. Pervasive Computing and Communication Workshops*, 2013, pp. 481-486.

[141] J.L. Wang and S.P. Huang, "Fuzzy Logic Based Reputation System for Mobile Ad Hoc Networks," in *Knowledge-Based Intelligent Information and Engineering Systems*, B. Apolloni, R.J. Howlett, and L. Jain, Eds.: Springer Berlin Heidelberg, 2007, vol. 4693, pp. 1315-1322.

[142] Y Wang, L Li, and G. Liu, "Social Context-Aware Trust Inference for Trust Enhancement in Social Network based Recommendations on Service Providers,"

World Wide Web, vol. 18, no. 1, pp. 159-184, January 2015.

[143] H. Wang, C. Li, C. Yan, Q. Li, and J. Li, "Ad hoc Grid Task Scheduling Algorithm Considering Trust-Demand," in *2nd Int. Conf. Future Computer and Communication*, 2010, pp. 108-113.

[144] Y. Wang et al., "LogitTrust: A Logit Regression-based Trust Model for Mobile Ad Hoc Networks," in *PASSAT*, 2014.

[145] D.A. Whetten, "What Constitutes A Theoretical Contribution?," *Academy of Management Review*, vol. 14, no. 4, pp. 490-495, 1989.

[146] "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," *IEEE Standard 802.11*, Jun. 1999.

[147] J. Wright et al., "A Model-Driven Approach to The Construction, Composition and Analysis of Services on Sensor Networks," in *Proc. Annual Conf. the Int'l. Technology Alliance*, 2010.

[148] H. Xia, Z. Jia, L. Ju, and Y. Zhu, "Trust Management Model for Mobile Ad Hoc Network Based on Analytic Hierarchy Process and Fuzzy Theory," *IET Wireless Sensor Systems*, vol. 1, no. 4, pp. 248-266, 2011.

[149] T. Xie and X. Qin, "An Energy-Delay Tunable Task Allocation Strategy for Collaborative Applications in Network Embedded Systems," *IEEE Trans. Comput.* , vol. 57, no. 3, pp. 329-343, March 2008.

[150] L. Xiong and L. Liu, "PeerTrust: Supporting Reputation-Based Trust for Peer-to-Peer Electronic Communities," *IEEE Trans. Knowledge and Data Engineering*, vol. 16, no. 7, pp. 843-857, 2004.

[151] Z. Yao, D. Kim, and Y. Doh, "PLUS: Parameterized and Localized Trust Management Scheme for Sensor Networks Security," in *Proc. 2006 IEEE Int'l. Conf. Mobile Adhoc and Sensor Systems*, 2006, pp. 437-446.

[152] B Ye, Y Wang, and L Liu, "CrowdTrust: A Context-Aware Trust Model for Workers Selection in Crowdsourcing Environment," in *IEEE 2015 Int'l Conf. Web Services*, 2015.

[153] O. Yilmaz and I.R. Chen, "Utilizing Call Admission Control for Pricing Optimization of Multiple Service Classes in Wireless Cellular Networks," *Computer Communications*, vol. 32, no. 2, pp. 317-323, 2009.

[154] Y. Yu, L. Guo, X. Wang, and C. Liu, "Routing security scheme based on reputation evaluation in hierarchical ad hoc networks," *Computer Network*, vol. 54, no. 9, pp. 1460-1469, 2010.

[155] T. Yu and K.-J. Lin, "A Broker-Based Framework for QoS-aware Web Service Composition," in *Proc. IEEE Int'l Conf. e-Technology, e-Commerce and e-Service*, 2005, pp. 22-29.

[156] T. Yu, Y. Zhang, and K.J. Lin, "Efficient Algorithms for Web Services Seslection with End-to-End QoS Constraints," *ACM Transactions on the Web*, vol. 1, no. 1, May 2007.

[157] V.A. Zeithaml, "Consumer Perceptions of Price, Quality, and Value: A Means-End Model and Synthesis of Evidence," *Journal of Marketing*, vol. 52, pp. 2-22, 1988.

[158] L. Zeng et al., "QoS-Aware Middleware for Web Services Composition," *IEEE Trans. Software Engineering*, vol. 30, no. 5, pp. 311-327, 2004.

[159] H. Zhang, Y. Wang, and X. Zhang, "A Trust Vector Approach to Transaction Context-Aware Trust Evaluation in E-commerce and E-service Environments," in *5th IEEE Int'l Conf. Service-Oriented Computing and Applications*, 2012, pp. 1-8.

[160] Z Zheng and M.R. Lyu, "Investigating QoS of Real-World Web Services," *IEEE Trans Services Computing*, vol. 7, no. 1, pp. 32-39, 2014.

[161] H. Zhu, S. Du, Z. Gao, M. Dong, and Z. Cao, "A Probabilistic Misbehavior Detection Scheme Towards Efficient Trust Establishment in Delay-Tolerant Networks," *IEEE Trans. Parallel and Distributed Syst.*, vol. 25, no. 1, pp. 22-32, 2014.