cTrust: Trust Management in Cyclic Mobile Ad Hoc Networks

Huanyu Zhao, Xin Yang and Xiaolin Li, Member, IEEE

Abstract—Mobility model and network topology play important roles in mobile ad hoc networks (MANET). Most existing trust and reputation management systems in MANET do not address mobility issues adequately. In this paper, we study the trust management problem in MANET with cyclic movement patterns. In a cyclic mobile ad hoc network (cMANET) where nodes move periodically, we formulate trust management problems and propose the cTrust scheme to handle trust establishment and aggregation issues. Unlike trust management in conventional schemes, trust management in cMANET involves not only neighbor trust relationships but also location and time factors. We model trust relations as a trust graph in cMANET to enhance accuracy and efficiency of trust establishment among nodes. Leveraging the distributed Bellman-Ford algorithm and stochastic Markov chain process for fast and lightweight aggregation of trust scores, the cTrust scheme is a decentralized and self-configurable trust aggregation scheme. To evaluate the performance, we implement the proposed cTrust scheme. We use the student contact patterns in the National University of Singapore (NUS) campus and Seattle metro bus traces as case studies for our cMANET communication model. The simulation results demonstrate the features of trust relationship dissemination in real environments and the efficiency, accuracy, scalability and robustness of the cTrust scheme. With increasing scales of ad hoc networks and complexities of trust topologies, cTrust scales well with marginal overheads.

I. INTRODUCTION

Research in mobile ad hoc networks (MANET) has made tremendous progress in fundamental protocols, routing, packet forwarding and data gathering algorithms, and systems. Different from the conventional networks, in MANET, nodes carry out routing and packet forwarding functions so that they act as both terminals and routers. Each node is free to move independently and will therefore change its connections to other nodes frequently, which results in a very high rate of network topology changes. The communication is usually multi-hop, and each node may forward traffic even unrelated to its own use. The transmission power, computational ability and available bandwidth of each node in MANET are limited.

To handle trustworthiness issues in open environments, trust and reputation systems have been proposed. In a trust and reputation system, historical behaviors are recorded for each entity, and these statistics are used to predict how the entity is likely to behave in the near future. The trust and reputation mechanisms encourage nodes to behave in a trustworthy manner, detect the misbehavior as well as improve the network performance.

A critical component of the trust relationship research in MANET is the topologies and mobility models. The topologies and mobility models determine how the nodes move in the network. They significantly impact the performance of the trust and reputation management protocols. Therefore, it is important to design proper trust and reputation schemes according to various mobility models that accurately represent the intended application scenarios. Only in this way, the trust and reputation can be more accurately predicted.

A variety of mobility models have been proposed for ad hoc networks. Some typical models include random walk mobility model (RWMM) [6], random direction mobility model (RDMM) [21], random waypoint mobility model (RWPMM) [1], and realistic mobility model (RMM) [7]. RWMM and RDMM are the simplest mobility models depending on random directions and speeds. RWPMM introduces pause time between changes in destination, direction, and speed. RMM tries to simulate a more realistic environment by placing obstacles in the area.

Nevertheless, most of the existing trust and reputation systems in MANET merely considered fixed position models, which ignored the most important feature of mobility. Some schemes adopted totally random mobility models, which are not common in practice. For example, buses in a city, students on a campus, people in a shopping mall, cell phones in a cellular network, and sensor nodes in a battle field generally do not move in random directions or at random speeds. Most objects are moving towards specific destination or following particular traces in real MANET. Furthermore, random mobility models involve substantial management overheads to maintain trust relationships in a completely unpredictable environment. Previous trust and reputation management systems with unrealistic mobility settings may not correctly reflect the true trust relationships in MANET.

Contrary to the fixed and random mobility scenarios, a large class of objects in MANET follow predictable movement patterns in reality. Among these movement patterns, we focus on MANET where nodes moving cyclically, which is modeled...
as cyclic mobility in mobile ad hoc network (cMANET) [15], [26]. The cyclic mobility model is common in reality. E.g., a bus system could be considered as cMANET. In a bus system, each bus follows a scheduled route during one business day. Another instance is the movement pattern of students on university campus. Following the class schedule, a college student presents at a scheduled classroom periodically every week, which is also considered as a cyclic presence/appearance model.

Trust establishment in cMANET is still an open and challenging problem. Due to the features of cMANET, conventional centralized trust establishment approaches are not suited well for cMANET scenarios. Unlike the peer to peer (P2P) trust, trust in cMANET is established on neighbor trust relationships. Trust in cMANET is also location and time dependent. In this paper, we focus on the trust management problem in cMANET. We study trust propagation in a fully self-organized environment where nodes do not rely on any centralized server or trusted authority. We propose a trust aggregation scheme called cTrust to aggregate distributed trust information in decentralized and highly dynamic cMANET environments. cTrust aggregation scheme leverages a stochastic distributed Bellman-Ford algorithm to achieve fast and lightweight trust rating aggregation. Our contributions in this work are multifold. (1) We model the movement patterns and trust relationships in cMANET as a trust graph and model the most trustable path finding process as the Markov decision process (MDP). (2) We propose trust transfer function, value iteration function and distributed trust aggregation algorithm to solve the most trustable path finding problem. This algorithm uses a stochastic Markov chain based process, which greatly reduces the message overhead. It requires only local communication between neighbor nodes and captures a concise snapshot of the whole network from each node’s perspective. (3) We design the evaluation metrics for cTrust. We develop a trust topology generator to establish trust relationship topologies. Using random and scale-free trust topologies, we simulate cTrust system and conduct extensive experimental evaluations based on the student movement traces on the campus of National University of Singapore (NUS) and the bus movement traces of the Seattle metro bus system. We evaluate the performance of cTrust scheme in terms of efficiency, accuracy, and scalability.

The rest of the paper is organized as follows. Section II presents the related works. Section III describes the conceptual architecture, and proposes the trust model. Section IV presents the stochastic distributed cTrust aggregation algorithm. We present the simulation results to investigate the performance of cTrust in Section V. Section VI concludes the paper.

II. RELATED WORK

The most popular reputation system is the feedback scheme used by eBay. The EigenTrust scheme proposed by Kamvar et al. presented a method to obtain a global trust value for each peer by calculating the eigen value from a trust ratings matrix [9]. Xiong et al developed a reputation-based trust framework PeerTrust [34]. Zhou et al proposed the PowerTrust system for DHT-based P2P networks [40]. Hwang’s group has also built the GossipTrust system [41] and the FuzzyTrust System [25]. Liang et al proposed a personalized trust model PET for resource sharing [13]. NICE is a trust inference scheme in distributed networks [10]. Zhao et al proposed the concept of trust vector and a trust management scheme VectorTrust for aggregation of distributed trust scores [37], [38], and a group trust rating aggregation scheme H-Trust inspired by the H-Index technique [36]. Credence is a decentralized object reputation and ranking management system for large-scale peer-to-peer file sharing networks [31]. The one-hop reputation protocol was designed for propagating reputation in P2P network for making service decisions [19]. A collaboration-based autonomous reputation system was proposed for Email services [33]. A new emerging research area in trust community is reputation systems in social networks [18], [24]. Comprehensive survey and overview on trust and reputation systems can be found in [16], [22], [28].

Trust research for MANET is also an active area. Buchegger et al proposed a reputation scheme to detect misbehaviors in MANET [2]. Their scheme was based on a modified Bayesian estimation method. Buchegger et al also proposed a self-policing reputation mechanism [3]. The scheme utilizes the local observation of nodes and leverages second-hand trust information to detect misbehaving nodes. The CORE system adopted a reputation mechanism to achieve cooperation in MANET [17]. The goal of CORE system was to prevent selfish behaviors. Yan Sun et al considered trust as a measure of uncertainty and presented a formal model to represent, model, and evaluate trust in MANET [29]. Ganeriwal, Saurabh et al. extended the trust scheme application scenario to sensor networks and built a trust framework for sensor networks [5]. Another ad hoc trust scheme was [30] where the trust confidence factor was proposed. In vehicular mobile ad hoc networks, a privacy-preserving system was described to guarantee the trustworthiness of vehicle-generated announcements [4]. A privacy-preserving system that guaranteed message trustworthiness in vehicle-to-vehicle communications was also proposed [32]. Li et al recently proposed a hierarchical reputation management system for large-scale MANET [12]. In their work, reputation and price systems were combined to increase performance. A distributed hash table approach was implemented to store the global reputation records. Other work with a relevant scope included [23], [14], [35], [28], [20].

Compared with existing literatures in the trust and reputation system research community, our approach and experiment settings are significantly different. To study the trust relationships in MANET, our scheme focuses on high-mobility settings (with time and location factors). This paper presents a realistic application scenario with rich real-world data traces.

III. TRUST MODEL IN cMANETS

We study a homogeneous MANET system without any centralized infrastructure or trust. Nodes can join or leave the MANET without any centralized administration. No nodes are pre-trusted. We represent file download/sharing and packet forwarding interactions as transactions. The transaction can
also be extended to real buy/sell behavior via virtual market in ad hoc networks [11]. Due to the transmission power limitation in MANET, remote transaction authentication is performed by multi-hop/chains of nodes. After transactions, nodes are required by the reputation schemes to assign their ratings for the transactions. A rating scheme exists to rate the service in such an environment. We assume most nodes move independently and generally follow some specific cyclic movement patterns.

The initial cTrust architecture is built on the top of routable MANET environments and trust rating schemes without centralized servers. The trust rating layer provides trust ratings for direct transactions. The trust rating value could be obtained by applying trust rating functions considering various factors in the historical transactions, such as importance, quality of service, time, and location. How to rate a service and how to generate the accurate and stable direct trust ratings are beyond the scope of this paper. In this paper, we adopt a personalized trust rating and flexible (full or selective) aggregation scheme which is defined in [36]. We assume that normalized trust ratings have been already generated in the trust rating layer, and the direct trust ratings are relatively stable. Practically, we use our trust relation generator to generate the direct trust ratings. This paper studies the trust dissemination and aggregation problems, and analyze several key features of the proposed cTrust scheme in cyclic mobile ad hoc environment.

A. Communication Model

In cMANET, each node has short radio range, high mobility, and uncertain connectivity. Two nodes can communicate only when they reach each other’s transmission range. When two objects meet, they have a contact probability \( P(P \in [0, 1]) \) that they contact or start some transactions.

Contrary to the completely random mobility, the cyclic movement pattern is quite common in real world MANET [15]. For instance, a bus system could be considered as a cyclic movement MANET. In a bus system, each bus visits stops in a scheduled route periodically. Following the class schedule, a college student regularly presents in a scheduled classroom every week, which is also considered as a cyclic movement.

cMANET movement traces are not required to follow certain shapes. For instance, on a college campus, most students generally walk along paths interconnecting campus buildings. Some individuals, however, may not strictly follow these paths (e.g., cut across lawns). And some students may be absent from classes. These situations are still considered as cyclic mobilities. Specifically, the “cyclic” property is characterized as follows: if two nodes meet at time \( T0 \) (\( T0 \) represents the first time two nodes meet), they have a high probability to meet after every particular time period \( TP \). In this paper, we represent movement traces in some regular shapes to ease the presentation and understanding, but the proposed scheme can be used for cyclic movements in irregular shapes.

B. Trust Graph

Figure 1 illustrates a three-node cMANET and their cyclic movement snapshots. The unit time is set as 10. Each node \( i \) moving cyclically has a motion period or cycle time of \( C_i \). We can tell from the trace that \( C_A = 30 \), \( C_B = 30 \) and \( C_C = 20 \). The system motion cycle time \( C_S \) is the least common multiple (LCM) of all the nodes’ motion cycle time in the network, \( C_S = lcm(C_A, C_B, C_C) = 60 \).

To represent the cMANET system features and trust relationships, we combine the snapshot graphs and trust relationships into a directed trust graph as shown in Figure 2. In a trust graph, based on the states and locations of nodes at time points (following a cyclic pattern), all the moving nodes are represented as a collection of states \( \{X_1, X_2, X_3, ..., X_n\} \) which are all vertices in trust graph. One node may have multiple states according to locations and time. We represent the states for each node as \( i/T_i\{Loc\} \) where \( i \) is the node ID and \( T_i\{Loc\} \) is appearance time at particular locations. The appearance time is give by,

\[
T_i\{Loc\} = T_0\{Loc\} + C_i \times n (n = 0, 1, 2, \ldots)
\]

where \( T_0\{Loc\} \) is the first time node \( i \) appears at this location, and \( C_i \) is node \( i \)'s motion cycle time. For example, node \( A \) appears at three locations. So in trust graph, node \( A \) is represented by three states: \( A/T_A\{Loc_0\}, A/T_A\{Loc_1\} \), \( A/T_A\{Loc_2\} \). Following Equation (1), we have,

\[
T_A\{Loc_0\} = 0 + C_A \times n = 30n (n = 0, 1, 2, \ldots)
\]

\[
T_A\{Loc_1\} = 10 + C_A \times n = 10 + 30n (n = 0, 1, 2, \ldots)
\]

\[
T_A\{Loc_2\} = 20 + C_A \times n = 20 + 30n (n = 0, 1, 2, \ldots)
\]

The three states for node \( A \) include \( A/30n, A/(10 + 30n) \), and \( A/(20 + 30n) \). As state transfer edges in a trust graph, the directed dashed lines between states represent nodes’ movements.

Leveraging our previous proposed notion of trust vector [37], [38], a personalized trust is represented as a vector of trust rating and direction, where trust rating is defined as a real number \( R (R \in [0, 1]) \) and direction is defined as a directed edge in the trust graph. This directed link with trust rating is called trust vector (TV). The initial trust relationships (trust vector) are represented by the solid directed edges in the
the trust among various states of the same node indicates node direct trust neighbors is represented by the set $H$. The appearance time for each link is given by function showing when two nodes can communicate via this stable. We study how trust information propagates in a high-ratings have been generated. The trust ratings are relative service quality. In this paper, we assume the normalized trust to consider all the history transactions’ importance, date, and value can be derived by applying rating generation functions all history transactions between two nodes. This trust rating observations generated from a combination/collection result of node. The trust table consists of the remote node ID as entry, the trust rating and appearance time, one solid edge in trust graph is trust rating, there is no negative edge. In a personalized edge is trust rating, $R_{i,j}$. The trust graph also solves the MTP finding problems in cMANET. Towards the same remote node, various trust paths may yield conflicting trust opinions. The meaning of the trust paths and ratings are application scenario dependent. In this paper, for a local node with limited local information, we advice to rely on MTP to start a multi-hop transaction or packet forwarding.

### C. Trust Path Finding Problems

In the cTrust system, each node maintains a local trust table. The trust table consists of the remote node ID as entry, the trust rating for each reachable remote node, and the next hop to reach the remote node. Each entry shows only the next hop instead of the whole trust path. Initially, one node’s trust table contains only the trust information of its one-hop direct interacted neighbors with pre-computed trust ratings.

In cMANET, due to the communication range and power constraint, nodes cannot communicate with remote nodes directly. Suppose node $i$ wishes to start a transaction with remote node $k$. Node $i$ wishes to infer an indirect trust rating for node $k$ to check node $k$’s reputation. In cTrust, the trust transfer is defined as:

**Definition 3.2 (Trust Transfer):** If node $i$ has a trust rating $R_{i,j}$ towards node $j$, node $j$ has trust rating $R_{j,k}$ towards node $k$, then node $i$ has indirect trust $R_{i,k}$ which is a function of $R_{i,j}$ and $R_{j,k}$ (defined as $R_{i,k} = R_{i,j} \otimes R_{j,k}$), towards node $k$.

Note that in Definition 3.2, $R_{i,j}$ and $R_{j,k}$ can be either direct or indirect trust. The transfer function ($R_{i,j} = R_{i,j} \otimes R_{j,k}$) definition is relying on the application scenarios. Beside the trust rating, node $i$ also wishes to find a trustable path, and it relies the multi-hop communication to finish one transaction. There might be multiple trust paths from node $i$ to node $k$ with various trust ratings. Given a set of paths between $i$ and $k$, $i$ tends to choose the most trustable path (MTP).

**Definition 3.3 (Most Trustable Path):** The most trustable path from node $i$ to node $k$ is the trust path yielding highest trust rating $R_{i,k}$.

MTP is computed as the maximal value applying trust transfer function on all edges along a path. And this value will be considered as $i$’s trust rating towards node $k$. cTrust system solves the MTP finding problems in cMANET. Towards the same remote node, various trust paths may yield conflicting trust opinions. The meaning of the trust paths and ratings are application scenario dependent. In this paper, for a local node with limited local information, we advice to rely on MTP to start a multi-hop transaction or packet forwarding.

**D. Markov Decision Process Model**

Markov decision process (MDP) is a discrete time stochastic control process consisting of a set of states. In each state there are several actions (to determine the next hop in trust path finding process) to choose. In a trust path finding process, the current hop/state is represented as $x$, and next hop/state is represented as $x’$. The state transition function $P_{x,x’}$ determines the transition probabilities to the next state. A reward is also
earned for each state transition. We model the MTP finding process as a MDP. We propose value iteration to solve the MTP finding problem.

Initially, for a sequence of random node states in trust path \((x_1,x_2,x_3,...,x_t)\) and a collection of all states in trust graph \(\{X_1,X_2,X_3,...,X_n\}\), the trust path has the following relation:

\[
Pr(x_{t+1} = X_{t+1} | x_1 = X_1, x_2 = X_2,..., x_t = X_t) = Pr(x_{t+1} = X_{t+1} | x_t = X_t)
\]  

(3)

where \(Pr()\) is the probability that state transition in current state at time \(t\) will lead to new state at time \(t+1\). Equation (3) indicates the state transitions of a trust path possess the Markov property: the future states depend only on the present state, and are independent of past states.

The state transition probability from state \(x\) to state \(x'\) is computed from normalizing all state \(x\)'s out trust links (trust ratings).

\[
P_{x,x'} = \frac{R_{x,x'}}{\sum_{y \in \mathcal{C}(x)} R_{x,y}} \quad (4)
\]

In each node state, the next state probabilities sum to 1. The trust path finding process is a stochastic process that all state transitions are probabilistic.

The goal is to maximize the cumulative trust rating for the whole path, typically the expected product from the source node to the destination node.

\[
\gamma R_{x_1,x_2} \otimes \gamma^2 R_{x_2,x_3} \otimes \gamma^3 R_{x_3,x_4} \otimes \ldots \otimes \gamma^t R_{x_t,x_{t+1}}
\]  

(5)

where \(\gamma\) is the discount rate and satisfies \(0 \leq \gamma \leq 1\). It is typically close to 1.

Therefore, we claim the MTP finding process is a MDP \((S,A,P,\pi)\).

The solution to this MDP can be expressed as a trust path \(\pi\) (MTP), the standard family of algorithms to calculate the policy \(\pi\) is the value iteration process.

### E. Value Iteration

Section III-C presents the trust transfer function \(R_{i,k} = R_{i,j} \otimes R_{j,k}\). The upper bound for \(R_{i,j} \otimes R_{j,k}\) is \(\min(R_{i,j},R_{j,k})\) because the combination of trust cannot exceed any original trust. \(R_{i,j} \otimes R_{j,k}\) should be larger than \(R_{i,j} \times R_{j,k}\), which avoids a fast trust rating dropping in trust transfer. The discount rate \(\gamma (\gamma \in [0,1])\) determines the importance of remote trust information. The trust transfer function \(R_{i,j} \otimes R_{j,k}\) needs to satisfy the following condition:

\[
R_{i,j} \times \gamma R_{j,k} \leq R_{i,j} \otimes R_{j,k} \leq \min(R_{i,j},\gamma R_{j,k})
\]  

(6)

In cTrust scheme, we set the trust transfer function as:

\[
R_{i,j} \otimes R_{j,k} = \min(R_{i,j},\gamma R_{j,k}) \ast \sqrt[\sqrt{\gamma}]\times \max(R_{i,j},\gamma R_{j,k})
\]  

(7)

where \(\alpha\) is the adjusting factor. By setting up \(\alpha\) (\(\alpha = 1,2,3...\)), \(R_{i,j} \otimes R_{j,k}\) can be sliding between the upper and lower bound. It is straightforward to prove that the trust transfer function (7) meets the condition (6). We skip the proof here.

In each round of the iteration, the trust table of each node is updated by choosing an action (next hop state in trust graph). The value iteration is executed concurrently for all nodes. It compares the new information with the old trust information and makes a correction to the trust tables based on the new information. The trust tables associated with the nodes are updated iteratively to keep local trust table updated. Utilizing the trust transfer function, the value iteration function is defined as:

\[
R_{i,k} = \max \left( R_{i,k} \alpha \left[ \min(R_{i,j},\gamma R_{j,k}) \ast \sqrt[\sqrt{\gamma}]\times \max(R_{i,j},\gamma R_{j,k}) \right] \right)
\]  

(8)

where \(R_{i,k}\) is the is the trust rating towards node \(k\) given the local trust table of node \(i\), \(R_{i,j}\) is the direct trust from node \(i\) to node \(j\). And \(R_{j,k}\) is the received trust information towards node \(k\) from node \(j\), which could be direct or inferred trust information. \(\alpha (\alpha \in [0,1])\) is the learning rate. The learning rate determines to what extent the newly acquired trust information will replace the old trust rating. A learning rate \(\alpha = 0\) indicates that the node does not learn anything, and a learning rate factor \(\alpha = 1\) indicates that the node fully trusts and learns the new information.

A special case in value iteration function (8) is that when node \(j\) is currently the next hop in node \(i\)'s local trust table towards node \(k\). Receiving a new trust rating \(R_{j,k}\) from node \(j\) indicates the next hop node \(j\) revised its current trust rating towards node \(k\). So the current trust information \(R_{i,k}\) in node \(i\) is no longer valid. We consider this case as a current trust path/rating update instead of a new trust path calculation/comparison. The trust rating revision \(R_{j,k}\) on node \(j\) could be increasing or decreasing. In either case, function (9) is used instead to refresh node \(i\)'s current trust table information.

\[
R_{i,k} = \alpha \left[ \min(R_{i,j},\gamma R_{j,k}) \ast \sqrt[\sqrt{\gamma}]\times \max(R_{i,j},\gamma R_{j,k}) \right]
\]  

(9)

### IV. Trust Aggregation Algorithm in cTrust

#### A. cTrust Distributed Trust Aggregation Algorithm

In the initial stage of an evolving cMANET, pre-set direct trust ratings are stored in local trust tables. However, the direct trust information is limited and does not cover all potential interactions. For most remote nodes, without adequate direct trust information, indirect trusts are needed. The distributed trust aggregation algorithm (Algorithm 1) gathers trust ratings to any node in a network. In this algorithm, each trust path is aggregated to MTP with the highest trust rating towards destination. Indirect trust information will be added to trust tables and be updated as the aggregation process evolves.

Updates are performed periodically where nodes retrieve the trust table from one of their direct trust neighbors and replace existing trust ratings with updated ones in local trust tables, and then include relevant neighbors as the next hops.
We present below an illustrated example to ease the understanding of cTrust aggregation operations.

![Illustrated Example](image.png)

**Illustrated Example.** Consider a network with six nodes shown in Figure 3. We illustrate the evolution of the trust table in Figure 4, where $T$ denotes the current time (unit time is set as 10). To ease the understanding, in this example, the parameters in value iteration function (8) and (9) is set up as $\alpha = 0.95$ (learning rate), $\gamma = 0.99$ (discount factor), $n_{i_a} = 8$ (adjusting factor). The contact probability for all nodes is set as $P = 0.9$.

The original trust tables are constructed according to the direct experience and are shown in Figure 4(a) to 4(f).

The aggregation begins at time $T = 0$. $A$ requests $D$’s trust table, and $D$ requests $A$’s trust table. Node $C$ connects to both $F$ and $E$. So $C$ normalizes its transition probabilities by Equation (4). The transition probability to $F$ is $0.9/(0.9 + 0.9)=0.5$ and the transition probability to $E$ is $0.9/(0.9 + 0.9)=0.5$. Suppose that $C$ requests table from $F$ by this transition probability distribution, $C$ and $F$ successfully finish the table exchange (the contact probability $P$ is set as $P = 0.9$, and in this example we assume $C$ and $F$ successfully finish the contact). As nodes receive this information, they recalculate the trust ratings. E.g., $A$ receives a trust table from $D$ that tells $A$ there is a trust path via $D$ to $E$, with a trust rating of $R_{D,E}=0.6$. Then $A$ knows it has a trust path to $E$ that is $R_{A,E}=0.53$ by value iteration function (8). Similarly, $D$ and $E$ have gained new trust information. The updated trust tables after this round of iteration are shown in Figure 4(g) to 4(l).

At $T = 10$, no node is within each other’s communication range. At $T = 20$, communications prompt nodes to recalculate their trust tables by value iteration function. This process proceeds, and finally the trust tables of nodes reach convergence status. The final trust tables are shown by Figure 4(m) to 4(r). In this example, the convergence status is defined as the difference between any node’s two consecutive trust tables is 0. However, in a dynamic environment and in a real mobile ad hoc network, it is hard to achieve strictly convergence status. With this consideration, we are introducing $\varepsilon$-convergence, which will be presented in following sections. Note that we introduce convergence status here to ease the presentation of examples and experiments. In real decentralized networks, no global convergence status can be detected by individual node. Individual node keeps sending and receiving tables to keep local trust table updated.

**B. Trust Search and Trust Inference**

With the aggregation process going on, each node is establishing a local trust table, which represents a current local view of the network. When there is a need to obtain a trust value on a remote node, trust search is initiated. E.g., assume node $A$ wants to infer a trust value for node $F$. Node $A$ searches for a target entry $F$ in its own local trust table as shown in Figure 4(m) and finds the trust rating for $F$ is $R_{A,F} = 0.64$. The MTP can also be inferred hop by hop as shown by the red colored trace in Figure 3.

**C. Membership Maintenance**

**Algorithm 2** cTrust New Node Initialization Algorithm

1. Assign node ID to the new node by MANET layer
2. The new node’s initial local trust table is established by direct transactions
3. The new node sends a trust request to its trust neighbors
4. Neighbors that receive the trust request send back their entire trust table
5. Received entries and local direct trust ratings are checked, and the initial local trust table are updated

In cTrust, the initialization algorithm for a new joined node is described in Algorithm 2. Each node $i$ periodically sends its entire trust table to its previous clients every $T_p$ time. If new trust table information is received, the node will relax its local trust table by trust value iteration function (8) or (9). If the received trust table does not contain new information, the current local trust table should be synchronized to validate.
each entry. In cTrust, life time of a trust table entry is set to $T_L$ ($T_L > T_p$). Each entry is periodically checked. If no update is received towards a specific node for $T_L$ period, node left or node dead is assumed. In the case of node leaving, the relevant entries in a trust table will be removed and update messages will be propagated through neighbors.

V. EXPERIMENTAL EVALUATION

A. NUS Student Contact Trace Experiment

We construct an unstructured network based on the National University of Singapore (NUS) student trace. The data of contact patterns is from the class schedules for the spring semester of 2006 in NUS among 22341 students with 4875 sessions [26], [27]. For each enrolled student, we have her/his class schedule. It gives us extremely accurate information about the contact patterns among students over large time scales. The contact patterns among students inside classrooms are presented in [26] and are recapitulated as follows.

Following class schedules, students move around on campus and meet each other when they are in the same session. The trace data set considers students' movements during business hours, and ignores contacts that students hang around campus for various activities outside of classes. The contact patterns among students inside classrooms are presented in [26] and are recapitulated as follows.

The NUS contact patterns can be modeled as cMANET. In our experiment, 1000 students are randomly chosen among 22341 to simulate 1000 moving nodes in cMANET. Following her/his class schedule, each student presents cyclically in classrooms. The contact probability $P$ is set as 0.9 which indicates that when two nodes meet, they have a probability of 90% to communicate. We considered all 4875 sessions in the data set. The time unit is 1 hour. The time for the whole system cycle ($C_S$) is 77 hours (77 time units). Note that in NUS student patterns, all traces are function-independent and shape-independent.

Trust Topologies. The initial trust relationships of nodes are generated by specifying the network size, trust density and trust topology. Then we generate trust relationship files that are used as the initial trust relationships in our simulation. The random trust topology and scale-free trust topology are used to establish trust relationships in this simulation. The trust outdegree of a node is the number of directed trust neighbors per node, which indicates the initial trust relation density. In random trust topology, the trust outdegree follows normal distribution with mean value $\mu_d$ and variance $\sigma_d^2$. All nodes have similar level of initial trust links. Under the scale-free trust topology, highly active nodes possess large number of trust links, while most nodes only have a small number of trust links. The number of trust links follows the power law distribution with a scaling exponent $k$.

Parameter Setting. The network size is configured from 100 nodes to 1000 nodes using our 1000 random selected students. The network complexity is represented in terms of the average outdegree of nodes $d$. A network complexity with $d = 20$ indicates on average, the initial outdegrees is 20. Node's real behavior is represented by a pre-set rating score $r$ that follows the normal distribution ($\mu_r$, $\sigma_r^2$). In our experiment, “real behavior score” is a pre-set value ($r \in [0, 1]$) for each node to indicate each node’s real behavior. It indicates the percentage of honest behavior and cheating behavior in transactions for one node ($r = 0$ is dishonest, indicating a node will cheat in all transactions; $r = 1$ is honest, indicating a node will never cheat in transactions). Direct trust ratings will be pre-generated. We assume the direct trust rating is relative accurate. This is reasonable because any trust inference scheme relies on an accurate trust rating scheme. In our simulation, the direct trust rating $R$ is generated with a normal
TABLE I
PARAMETER SETTINGS FOR NUS STUDENT TRACE EXPERIMENT AND SEATTLE BUS TRACE EXPERIMENTS

<table>
<thead>
<tr>
<th>Notation</th>
<th>Environment Parameter</th>
<th>NUS Student Trace</th>
<th>Seattle Bus Trace</th>
</tr>
</thead>
<tbody>
<tr>
<td>α</td>
<td>Learning rate</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>γ</td>
<td>Discount factor</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>ρ_a</td>
<td>Adjusting factor</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>N</td>
<td>Network size</td>
<td>1000 (Students)</td>
<td>1163 (Buses)</td>
</tr>
<tr>
<td>N_S</td>
<td>Total sessions</td>
<td>4875</td>
<td>N/A</td>
</tr>
<tr>
<td>C_S</td>
<td>cMANET system cycle</td>
<td>77 hours (time units)</td>
<td>24 hours (4320 time units)</td>
</tr>
<tr>
<td>P</td>
<td>Contact probability</td>
<td>0.9</td>
<td>0.9</td>
</tr>
<tr>
<td>d</td>
<td>Trust topology complexity (average outdegree)</td>
<td>20, 25, 30</td>
<td>40, 60, 80</td>
</tr>
<tr>
<td>µ_d</td>
<td>Node outdegree normal distribution mean value</td>
<td>20, 25, 30</td>
<td>40, 60, 80</td>
</tr>
<tr>
<td>σ_d^2</td>
<td>Node outdegree normal distribution variance</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>k</td>
<td>Node outdegree pow-law distribution scaling exponent</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>r</td>
<td>Node’s real behavior score</td>
<td>0.25, 0.75</td>
<td>0.25, 0.75</td>
</tr>
<tr>
<td>µ_r</td>
<td>Initial direct trust rating</td>
<td>r</td>
<td>r</td>
</tr>
<tr>
<td>σ_R^2</td>
<td>Initial direct trust rating normal distribution variance</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>θ_acc</td>
<td>Aggregation accuracy threshold</td>
<td>10%</td>
<td>10%</td>
</tr>
<tr>
<td>ε</td>
<td>ε-convergence threshold</td>
<td>0.02</td>
<td>0.02</td>
</tr>
</tbody>
</table>

distribution \((\mu_R, \sigma^2_R)\) with mean value equal to the node’s “real behavior score” \(r\), which indicates the direct trust ratings are generally consistent with nodes’ real behavior. More parameter settings and default values are given in Table I.

Procedure. The simulation starts with initial parameters, generates the NUS student trace network movement pattern, a trust topology, and initializes local trust tables following the given distribution. Then the aggregation processes are simulated step by step for all nodes concurrently until convergence. In a dynamic model or in a real mobile ad hoc network, it is hard to achieve strictly convergence status. So the convergence in our simulation is \(\epsilon\)-convergence, and \(\epsilon\)-convergence is defined as that the variance/difference between any node’s two consecutive trust tables’ any entry is smaller than the pre-set threshold \(\epsilon\).

Convergence Time. The convergence time is measured in terms of the number of time units needed to achieve \(\epsilon\)-convergence status. The relationship between the network size, network complexity and convergence time are shown in Figure 5. The figure shows that cTrust only needs a small number of aggregation cycles before convergence. We also observe that convergence time increases as network complexity increases. As network size \(N\) increases, convergence time increases relatively slowly \((O(n))\). This shows that cTrust features satisfactory scalability. We also observe that the trust topologies do not affect the convergence time as much as network size.

Communication Message Overhead. Figure 6 shows the average communication message overhead to achieve convergence per individual node. cTrust greatly reduce the communication message overhead by using the MDP model. This is because, in each iteration, each node only receives trust table from one of its most trusted neighbors (as shown by Equation (4)) per iteration. The message overhead grows slowly as the network size grows, which shows that cTrust is a lightweight scheme. In a network with high complexity, cTrust system incurs more message overheads. In a typical cTrust network, the average message overhead is only affected by network size \(N\) and complexity \(d\) and not affected by trust topology. The overhead curves for both topologies in the figures appear similar.

Average Trust Path Length. Trust path length is the
number of hops from a source to a destination node. Figure 7 indicates the average length of a trust path starts from a source node to a destination node in convergence status. Generally, the trust path length increases with the network size and complexity increasing, which indicates nodes gain more remote trust information. In the scale-free trust topology, the trust path length is greatly reduced. This is because in the scale-free trust topology most nodes have only a few connections while some power nodes control many links, making trust information hard to spread. In a complex network where trust information can be spread farther, there are longer trust paths and involve more trust transfers.

![Average Trust Table Size](image)

**Fig. 8.** Average Trust Table Size (NUS Student Contact Trace)

**Average Trust Table Size.** The average trust table size (length) in convergence status is shown in Figure 8. This metric indicates how much trust information each node is able to obtain in cTrust. Trust table size grows with network size and complexity increasing, indicating there is more trust information in larger and complex networks. In a less complex network where nodes have a few connections to other nodes (the curve of \(d = 20\)), the average trust table size is low because trust information has fewer options to propagate. The table size in the scale-free trust topology is much smaller than that in the random trust topology, which indicates in the scale-free topology the trust relationships are more concentrated around super nodes.

![Aggregation Accuracy](image)

**Fig. 9.** Aggregation Accuracy (NUS Student Contact Trace)

**Accuracy.** cTrust aggregation accuracy is measured by comparing all nodes’ trust ratings with nodes’ real behavior scores. In our experiment, “real behavior score” is a pre-set value \((r \in [0, 1])\) for each node to indicate each node’s real behavior. For a particular node, we collect all the trust ratings towards it in the convergence status. When the relative difference between trust rating and real behavior score is less than a pre-set accuracy threshold \(\theta_{acc}\), we mark it as an accurate trust rating. Remember that initial direct trust rating normal distribution variance \(\sigma^2_R\) is set as 0.1, which introduces inaccurate direct trust ratings. Therefore, we are using \(\theta_{acc} = 10\%\) in this experiment. The overall percentage of accurate trust ratings across all nodes in the network represents system/network level aggregation accuracy. As shown in Figure 9, on average, cTrust aggregation accuracy is maintained above 90%. The result is very encouraging because cTrust is a personalized trust system using inferred (not direct) trust and the information for each node to access is limited in cMANETs. As the network complexity increases, the accuracy decreases. This is because in complex networks, there are more long trust paths that involve more trust transfers, resulting in lower accuracy in inferred trust ratings due to multi-hop relationships. The accuracy in scale-free trust topology is slightly higher than that in random trust topology. One reason is in scale-free trust topology, the average trust path is shorter, which leads to high accuracy in trust transfer.

**B. Seattle Bus Trace Experiment**

![Seattle Bus Trace](image)

**Fig. 10.** Our metro bus traces are generated from Seattle, Washington area King County metro bus system. The whole area is divided into regions. Metro bus traffic density varies in different regions. By courtesy of http://metro.kingcounty.gov/maps/.

Bus movements are typically cyclic. We explore the trust dissemination patterns in the Seattle metro bus system. We use the public metro buses’ actual movement traces in the Seattle, Washington area King County metro bus system. The movement data is acquired from several week-long traces of the movement of the fleet of city metro buses in Seattle, WA, on their daily normal routes providing passenger bus service throughout the city. The original data trace and contact patterns were presented in [8] and are recapitulated as follows.

Seattle bus trace presents a topographically challenging mobile ad hoc environment with a lake of 35 square miles in the middle area of the city. The metro bus system consists of 1163 vehicles. It covers an area of 5100 square kilometer, which is shown in Figure 10(a). The entire area is divided into several regions (shown in Figure 10(b)).

Each bus is tracked by an Automated Vehicle Location (AVL) system with a combination of odometry and signpost transmitters. Using the Busview software, Internet users are
able to monitor the location of buses in real-time. The traces' raw data are generated from location update messages sent by buses. Each bus's x and y coordinates were calculated using the latitude and longitude values obtained in the raw traces. The route IDs signify the routes followed by buses. There might be multiple shifts serving each of these routes. In our experiment, we allow multiple buses to follow a given bus route at any time.

We investigate how trust relationships can be efficiently spread and aggregated in such a dynamic environment. The scenario and results can be used to infer user behaviors and system usage patterns in cyclic mobile environments.

Our experiments are performed on 1163 buses. The unit time is set as 20 seconds. The system motion cycle (CS) is chosen as one day (24 hours), which consists of 4320 time units. A virtual potential contact is created if two buses are within 200 feet. In other words, we assume that when two buses are within 200 feet, they are within wireless communication range to each other. The bus contacts could take place either at a bus station or on a street. Random trust topology and scale-free trust topology are used to establish trust relationships. Parameter settings are given in Table I.

Fig. 11. Bus Encounters and Contacts in One Day (24 hours) Period (Seattle Bus Trace)

We consider all the 1163 buses with trust topology complexity \( d = 60 \) and investigate the effect of trust dissemination. Figure 11(a) shows the distribution of all bus encounters in one day (24 hours). Note that, bus encounters do not indicate bus contacts. Two buses conduct potential contact only when they have trust relations. Figure 11(b) reports actual bus contacts for one day (24 hours). The actual bus contacts are much fewer than all the bus encounters for the reason that buses contacts occur only when they are trust neighbors. The stochastic Markov chain based aggregation algorithms also greatly cut down potential trust contacts to reduce communication overheads.

Fig. 12. Trust path length and trust table size distributions in convergence status are summarized in Table II. The distributions are shown in Figure 12(a) and 12(b). We observe the difference in trust dissemination performance between random and scale-free topologies. It is easier for trust information to spread and aggregate in a random topology, which is also verified by the NUS trace experiment. In addition, we see that trust tables in more than 60% of the population hold less than 100 entries at convergence. With scale-free topology, the result is a little worse. In scale-free topology, the power nodes were initially assigned a large number of trust relationships. At convergence, the random topology gained more nodes with 1163 (maximum) trust table entries than scale-free topology. In a followup experiment, we show that at convergence, 20% of random picked pairs of buses have trust paths. We also show that with scale-free topology, this number reduces to 16%.

Fig. 13. Convergence Time (Seattle Bus Trace)

Fig. 14. Average Trust Path Length (Seattle Bus Trace)

Fig. 15. Average Trust Table Size (Seattle Bus Trace)

C. Summary of Experiment

We observe that the convergence time is much longer in the Seattle bus case than that in the NUS student case, indicating
slower trust information spreading. The cTrust aggregation performance using the NUS student contact trace outperforms that using the Seattle bus trace. This is mainly due to the regional feature of metro systems (shown in Figure 10(b)), i.e., the system is divided into several regions. Bus route density is high within each region but sparser across regions. Trust information spreads fast within some regions with heavy traffic like in the Seattle downtown area. While in some countryside regions, the trust information spreads much slower. Another reason is that there are much fewer inter-region bus routes compared with buses running within regions, which makes it hard to spread trust information from one region to another. These interesting patterns, however, is not so evident in the NUS student trace.

Figure 5 and 13 show that higher trust complexity will lead to longer convergence time. In both NUS student contact trace and Seattle bus trace, higher trust complexity achieves longer trust path (shown by Figure 7 and 14) and larger trust table size (shown in Figure 8 and 15) at convergence status, which indicates that trust information is easier to spread in high trust complexity environment.

In both experiments, the results show significant trust dissemination difference in random topology and scale-free trust topology. The trust dissemination performance was improved in a random trust topology in terms of longer average trust path and larger trust table size than that in a scale-free topology. However, the accuracy in the random topology is a litter lower. Figure 6 in the NUS student trace experiment shows that these two topologies involve similar message overheads. Figure 11(a) and 11(b) demonstrate that cTrust scheme greatly reduces the communication contacts to achieve lightweight trust aggregation.

Considering all the results, cTrust performs well with an increasing network size and complexity. Moreover, it achieves an aggregation accuracy of 90%.

VI. CONCLUSION

We have presented the cTrust scheme in cMANET (cyclic mobile ad hoc network). To the best of our knowledge, our scheme is the first fully distributed trust model in a cyclic mobile space. The cTrust scheme is aimed to provide a common framework to enable trust inferring and searching in a cMANET trust landscape. In this paper, we presented the trust transfer function, trust value iteration function, and the cTrust distribution trust aggregation algorithm. To validate our proposed algorithms and protocols, we conducted extensive experiments using NUS student traces and Seattle bus traces. The experimental results show the trust dissemination patterns and features in real cMANET environments. It demonstrates that cTrust is an efficient trust aggregation scheme. The convergence time of the cTrust scheme increases modestly with network size. Message overheads in cTrust are marginal. The trust information spreads fast and extensively in cMANET. The trust rating inference accuracy in the cTrust scheme is over 90%. We believe that cTrust establishes a solid foundation for designing trust-enabled applications and middleware in cMANET.

REFERENCES


<table>
<thead>
<tr>
<th>Topology</th>
<th>Random Topology</th>
<th>Scale-Free Topology</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convergence Time</td>
<td>44682</td>
<td>48119</td>
</tr>
<tr>
<td>Trust Path Length</td>
<td>308334</td>
<td>320501</td>
</tr>
<tr>
<td>Trust Table Size</td>
<td>23</td>
<td>27</td>
</tr>
<tr>
<td>Mean</td>
<td>5.17</td>
<td>4.72</td>
</tr>
<tr>
<td>StdDev</td>
<td>3.94</td>
<td>3.49</td>
</tr>
<tr>
<td>Total</td>
<td>1163</td>
<td>1163</td>
</tr>
<tr>
<td>Shortest</td>
<td>57</td>
<td>44</td>
</tr>
<tr>
<td>Mean</td>
<td>51</td>
<td>276</td>
</tr>
<tr>
<td>StdDev</td>
<td>351</td>
<td>99</td>
</tr>
<tr>
<td>Total</td>
<td>1163</td>
<td>1163</td>
</tr>
<tr>
<td>StdDev</td>
<td>463.03</td>
<td>408.11</td>
</tr>
</tbody>
</table>


Huanyu Zhao is currently a senior software engineer in AOL Advertising. Before joining AOL, he was a software engineer in Yahoo!, and a research intern in Mitsubishi Electric Research Labs. He received his PhD in Computer Science from Oklahoma State University in 2011, and dual bachelors in Information Security and Management Science from University of Science and Technology of China in 2006. His research interests include distributed computing, Hadoop ecosystem, trust management, security and privacy.

Xin Yang is a Ph.D. student in the Department of Computer & Information Science & Engineering at University of Florida and a research assistant at the Scalable Software Systems Laboratory. Prior to pursuing his Ph.D. study, he earned a MS in Computer Science at Oklahoma State University in 2010 and a BS in Computer Science at Nanjing University, China in 2004. His research interests include distributed computing, parallel computing, and cloud computing.

Xiaolin Li is Associate Professor in Department of Electrical and Computer Engineering at University of Florida. His research interests include Parallel and Distributed Systems, Cyber-Physical Systems, and Network Security. His research has been sponsored by National Science Foundation (NSF), Department of Homeland Security (DHS), and other funding agencies. He is an associate editor of several international journals and a program chair or co-chair for over 10 international conferences and workshops. He is on the executive committee of IEEE Technical Committee on Scalable Computing (TCSC) and served as a panelist for NSF. He received a PhD degree in Computer Engineering from Rutgers University, USA. He is directing the Scalable Software Systems Laboratory (http://www.s3lab.ece.ufl.edu). He received the National Science Foundation CAREER Award in 2010. He is a member of IEEE and ACM.