# Trusted Spanning Trees for Delay Tolerant Mobile Ad Hoc Networks

Apivadee Piyatumrong and Pascal Bouvry
University of Luxembourg,
FSTC - CSC,
Luxembourg
Email: apivadee.piyatumrong@uni.lu,
pascal.bouvry@uni.lu

Frédéric Guinand
Le Havre University,
LITIS,
Le Havre, France
Email: Frederic.Guinand@univ-lehavre.fr

Kittichai Lavangnananda
School of Information Technology,
King Mongkut's University
of Technology Thonburi (KMUTT),
Bangkok, Thailand
Email: kitt@sit.kmutt.ac.th

***Abstract*—Delay Tolerant Networks (DTNs) are an extension of Mobile Ad-Hoc Networks (MANETs). Global knowledge in DTNs cannot be obtained or guaranteed due to their dynamicity, decentralized nature and non-permanent structure. Managing such networks optimally is very difficult, if not impossible. Trust management in such networks receives much attention recently due to their potential application. One solution for managing information within DTNs lies in constructing and maintaining spanning forests. DA-GRS is a local computation based model for the description of decentralized algorithms designed for dynamically distributed environments like Delay-Tolerant MANETs (DTMs). DA-GRS proposes a framework for constructing and maintaining a spanning forest in such an environment. This work introduces the notion of trust into DA-GRS resulting in T-DA-GRS algorithm. The goal of the proposed algorithms is to construct and maintain robust trusted spanning tree where less trustable nodes are leaves. Three cost functions are suggested as means to assess the robustness of trusted spanning trees. T-DA-GRS is also further improved by incorporating greedy algorithm to become T-GDA-GRS. These algorithms were tested with four different networks generated by a DTM simulator known as Madhoc. Efficiency of these algorithms is compared with optimal values.**

## I. Introduction

Delay Tolerant Mobile Ad Hoc Networks (DTMs) are fluctuating networks populated by a set of moving materials equipped with wireless communicating devices. These materials are called stations, nodes or devices. They can spontaneously interconnect each other without any pre-existing infrastructure [1]. What makes the management of such network difficult is their nature. DTMs are mobile, ad hoc configuring, and frequently partitioned. At a given moment, two stations belonging to distinct partitions can neither communicate directly nor undirectly (using multi-hop communications).

The most popular wireless networking technologies available nowadays for building MANETs are Bluetooth and IEEE802.11 (WiFi). This implies that devices communicate within a limited range, and stations may move while communicating. A consequence of mobility is that the topology of such networks may change quickly and unpredictably. This dynamical characteristic constitutes one of the main obstacles for performing efficient communications. Furthermore, acquiring global information in this kind of network is difficult and impractical if not impossible. Therefore management information within this network needs to be done locally, but yet effective globally. Then, algorithms designed for DTMs have to be decentralized and robust to cope with both, the dynamic and the partitioned nature of the environment.

Dynamicity Aware - Graph Relabeling System (DA-GRS) [2] is a local computation based model for the description of decentralized algorithms intended to operate in dynamically distributed environments like DTMs. In [3] the author presents an algorithm for constructing and maintaining a spanning forest in DTMs. Such a structure may be of valuable help for exchanging information efficiently between different stations for instance. The presented algorithm relies on the use of tokens (a token is an information agent). Each tree in a spanning forest owns one unique token that moves randomly within its own tree. When two tokens meet, they merge stored information from both token and become only one token. As the previous two tokens become one token, their corresponding spanning trees are merged to form a new and larger spanning tree.

The work introduces a trust management algorithm in DTMs by cooperating the notion of trust into the existing algorithm used in DA-GRS (T-DA-GRS). The work assumes that the trust level of every node in the communication graph has been assessed. The goal of the algorithms in this work is to construct robust trusted spanning trees where less trustable nodes are leaves. The efficiency is furthered improved by cooperating the 'greedy algorithm' concept to form T-GDA-GRS.

## II. Related work

### A. Delay Tolerant Mobile Ad Hoc Networks (DTMs)

DTMs constitute an emerging subclass of mobile ad hoc networks that feature frequent and long-duration partitions [4]. In a DTM network, each station can reach a subset of the other stations using wireless communication abilities. Such communication ability is typically defined by a communication range and constrained by natural obstacles (e.g. walls, buildings, etc.).

At a given moment $t$, the communication graph, $G(t)$, of such network is a pair $(V_t(G), E_t(G))$, where $V_t(G)$ is a finite set of elements, called vertices, $E_t(G)$ is a binary relation on

$V_t(G)$ - a subset of pairs of elements of $V_t(G)$. The elements of $E_t(G)$ are called edges and constitute the edge set of $G(t)$. An edge between node $x_i$ and $x_j$ indicates that, at time $t$, it is possible for $x_i$ and $x_j$ to exchange information. $G(t)$ may be partitioned into a set of $m$ subgraphs: $G(t) = \bigcup_{1..m} P_i(t)$

### B. Dynamicity Aware - Graph Relabeling System (DA-GRS)

The DA-GRS model was invented as a help for design and analysis of decentralized applications and algorithms targeting dynamically distributed environments like DTMs. Normally, such applications and algorithms are often very difficult to set up, describe and validate [3]. Using DA-GRS is a convenient way to design algorithms for DTMs, since its outstanding properties are localized in a dynamic working manner. In the context of the study, DA-GRS approach proposes a way of designing a decentralized algorithm for constructing and maintaining a spanning forest in DTMs, relying on a careful rule-based token management. Hence forth this concept will be referred to as 'DA-GRS' for brevity. The work in [3] described rules to handle four different scenarios, (a) tokens traversal in general case, (b) when a token meets another token, (c) partition occurs at a node which belongs to the spanning tree that possess the token, (d) partition occurs at a node which belongs to the spanning tree which does not possess the token. As DA-GRS constructs random spanning trees, quality (in term of trust) of each spanning tree ought to be assessed.

Let $\Gamma_i$ be the set of all possible spanning trees for $P_i$. DA-GRS randomly selects $\gamma^{dagrs} \in \Gamma_i$. An ideal situation is to be able to select $\gamma^{optimal} \in \Gamma_i$ , or at least to select preferable $\gamma^*$ such that $\gamma^{dagrs} \leq \gamma^* \leq \gamma^{optimal}$

### C. Trust Management

In human society, trust has become the basis of almost all activities, such as communications, work, etc. People gradually form the standard of mutual trust, and they also refer to opinions of the third-party in assessing the trust. Trust can be regarded as a criterion for making a judgment under complex social conditions and can be used to guide further actions [5]. In summary, trust can be viewed as the expectation or the belief that a party will act kindly and cooperatively with the trusting party [6]. It is no surprise that some research related to security or mutual cooperation on multi-agent system paid particular attention to trust factor in various facets, [7], [8], [9].

In early stage of trust and security on MANETs, several trust and security establishments relied on cryptographic methods, authentication codes and hashing chains for their solutions. Although these schemes are effective, they are centralized system which produced significant communication overheads from both preprocessing and during processing periods, as well as energy consuming. These approaches are also not applicable to DTMs. In the last few years, cooperation enforcement methods (avoidance the effect of selfish nodes on the networks' robustness [10]), and reputation schemes [6], [11], [12] have been proposed for trust establishment in MANET. Recent literature suggests that the cooperation enforcement techniques are more appropriate if the primary goals are availability, robustness of the network, and the overall throughput. A comprehensive survey on cooperation enforcement can be found in [13], while detailed discussion on peer-to-peer key and trust management approach can be found in [14].

Quality of service is a key issue in DTMs where members, mobile stations may present different level of services. In [10] different strategies for mobile nodes are examined. In this work, trust is used in establishing on-the-fly security (in term of robustness) in a purely self-organized manner of DTMs (no pre-established relationship among nodes or off-line key distribution). An assumption is made that a trust model has been established and provision of trust information to different nodes is assumed.

## III. Trusted Spanning Trees

Trusted spanning tree in this work is a spanning tree which is cooperatively built in a cluster/partition of trusted nodes in order to manage information embedded in a MANET. Trust level of a node $n$, denoted by $trust(n)$, where $trust(n) \in Z^+$, defines the level of quality of services it can provide. Whether a node $n$ can be trusted is determined by a given threshold. Let $\Theta_t = \{n' \in V_t(G) | trust(n') \leq threshold\}$ be the set of all non-trustable nodes at moment $t$. A node in a cooperative network can have low level of trust for various reasons such as low battery, poor communication signal, moving out of communication range, etc. An ideal situation is to determine an optimal trusted spanning tree among many possibilities in a given cooperative network. To date, there is no efficient algorithm which can generate an optimal spanning forest in DTMs due to their dynamic and decentralized characteristics and lack of global knowledge in the cooperative network. Nevertheless, more robust trusted spanning trees are preferable to arbitrary ones. In order to determine robust trusted spanning trees, this work introduces quality measurement for trusted spanning trees by means of three cost functions.

### A. Cost functions

A trusted spanning tree in this work is evaluated by three cost functions, these are $weight()$, $weight\_penalty()$ and $isolating\_low\_trusted\_node()$. In order to summarize the quality of created trust spanning tree, the value of functions from different studied algorithms will be compared where a higher value indicates a better quality. These functions will be applied to an optimal trusted spanning tree and will be used as best-case. Figure 1(a) and (b) are examples to illustrate the idea behind these cost functions which the threshold of being non-trustable node is defined as equal to one.

*1) $weight()$ function:* In general case, nodes with higher trust level are more likely to be able to complete their tasks than lower ones. $weight()$ function introduced here, can be used to assess trust spanning trees with respect to this objective. The $weight()$ function of a trusted spanning tree

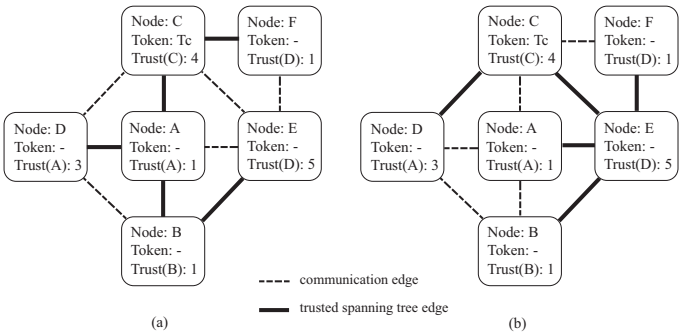



Fig. 1. An example scenario for illustrating cost functions used in this study

can be determined by the following equation:

$$weight(\gamma) = \sum_{x \in V(\gamma)} trust(x) * degree(x) \qquad (1)$$

The function $degree(x)$ represents the degree of node $x$. Figure 1 is used to illustrate how the $weight$ function can assess this quality. In Figure 1(a), the node with lowest trust level gets the highest degree, while the node with highest level gets the lowest degree (i.e the node A has a trust level of 1 and degree of 3, while the node E a trust level of 5 and degree of 1), hence the $weight()$ function for this trusted spanning tree is 22. Figure 1(b) depicts the opposite (i.e. the node with highest trust level possess highest degree (node E), while the node with lowest level possess lowest degree (node E)). The $weight()$ function for this trusted spanning tree is 34.

With more specific case, having nodes with low trust levels localized on leaves is advantageous since they would not be responsible for forwarding information to others. Furthermore, loosing them at these positions has little effect to the overall structure. On the contrary, as low trust level nodes have tendency to break away from the network, allowing them to have high degrees presents a difficult task of re-connecting the trusted spanning trees as a result of their breaking away. Therefore, in order to minimize the re-connecting task, nodes with lowest trust levels should be assigned the lowest degree position in the trees. The next two functions $isolating\_low\_trusted\_node()$ and $weight\_penalty()$ functions are introduced as means to assess trusted spanning trees with respect to the objective.

*2)* $isolating\_low\_trusted\_node()$ *function:* This function indicates the efficiency of a trusted spanning tree by noting how well it can isolate non-trustable nodes. The function measures the percentile of $n'$ nodes at terminal position. The higher value of $isolating\_low\_trusted\_node()$ function signifies better quality trusted spanning tree. Let $\Theta^*(\gamma) = \{n' \in \Theta(\gamma) | n'$ is at terminal position of $\gamma\}$. The $isolating\_low\_trusted\_node()$ function can be determined by the following equation :

$$isolating\_low\_trusted\_node(\gamma) = \left(\frac{\Theta^*(\gamma)}{\Theta(\gamma)}\right) 100 \qquad (2)$$

Hence, the *isolating_low_trusted_node* value for Figure 1(a) is 33.33% while this value is 100% for Figure 1(b).

*3)* $weight\_penalty()$ *function:* This function assigns an additional penalty to non-trustable nodes $\Theta(\gamma)$ which are *not* leaves and is used in conjunction with the $weight()$ function. This additional penalty can be determined by the following equation:

$$weight\_penalty(\gamma) =$$

$$(\sum_{x \in V(\gamma)} trust(x) * degree(x)) - (\sum_{n' \in \Theta(\gamma)} degree(n') - 1) \qquad (3)$$

As all non-trustable nodes with degrees higher than 1 are assigned additional penalty. Therefore, the trusted spanning tree in Figure 1, (a) incurs additional penalty of 19, while the one in Figure 1, (b) does not as there is no node with lowest trust level possesses the degree of more than 1 (i.e. all nodes with trust level of 1 are at terminal).

### B. Optimal tree

Since problem in this work is a multi-criteria one, the optimal tree can be viewed in different facets among the three cost functions proposed. For example, the optimal tree respecting to the $weight()$ function, at any particular time $t$, can be defined as: $\gamma^{optimal} \in \Gamma_i, |\forall \gamma \in \Gamma, weight(\gamma) \leq weight(\gamma^{optimal})$

## IV. Algorithms for Building Trust Spanning Trees

Trust management within a partition of DTM is very difficult because of its dynamicity, decentralized nature and non-permanent connection that can break up into two or more partitions at any moment. Although cooperative working manner among stations (i.e. nodes) within a DTM can be assumed, any trust management algorithm has to work at local level as global knowledge of the network cannot be acquired.

Spanning tree is a structure which facilitates trust management where communication among the set of nodes in the tree is possible via its edges (i.e. communication edges). In [2], the authors introduce an algorithm to manage spanning forest within a DTM environment. It constructs possible spanning trees in a DTM by means of using tokens. Token can be seen as an information agent, initially possessed by each node, that becomes obsolete as it connects to another spanning tree. Hence, each spanning tree owns a unique token. In this work, trust level is assumed to be from 1 to 5 (1 being the lowest and vice versa).

### A. T-DA-GRS

In this work, trust management in DTM is maintained by incorporating the notion of trust into the algorithm in DA-GRS. Hereafter, this algorithm will be referred to as T-DA-GRS for ease of reference. In T-DA-GRS, trusted spanning trees are initially constructed in the same way as in DA-GRS. Each token moves within their own trusted spanning tree. Merging of two trusted spanning trees occurs when two tokens meet. After the merge is complete, a new and larger trusted spanning tree is formed, and the two tokens also merge in one unique token. Figure 2, illustrates this operation. There are four trusted spanning trees and their tokens are at nodes A, B, C and D respectively. These tokens are within each

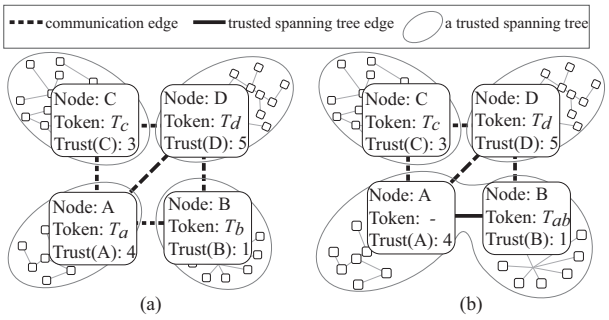



Fig. 2. An example merging using T-DA-GRS

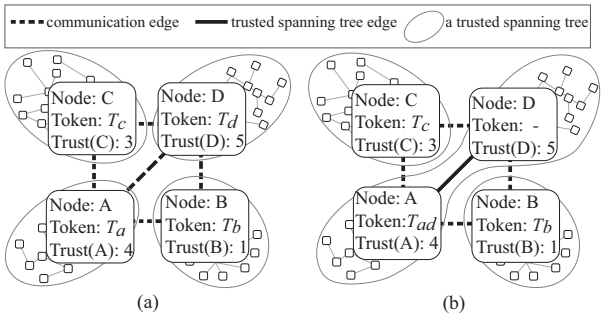



Fig. 3. An example merging using T-GDA-GRS

others access range. T-DA-GRS merges trusted spanning trees randomly. In this example, the trusted spanning tree with token at node A happens to merge with another trusted spanning tree with token at node B. Note that no consideration is given to the trust level of each node.

### *B. T-GDA-GRS*

Since T-DA-GRS constructs and merges trusted spanning trees randomly, robustness of each trusted spanning tree is left to chance. Therefore, T-DA-GRS is more likely to result in trusted spanning trees with low cost function values. In order to improve the possibility of generating more robustness of spanning trees, T-GDA-GRS is further improved by incorporating greedy algorithm concept. The principle of this improvement arises from the fact that merging tokens located on highest trust level nodes is likely to result in a more robust trusted spanning tree (i.e. a trusted spanning tree with higher cost function values). Hereafter, this algorithm will be referred to as T-GDA-GRS for brevity. The extension in the merging operation in T-GDA-GRS is described below.

**Algorithm 1 Look for other trees (tokens) around token $\tau_i$**

1: $\tau^{best}$ is the most trusted token in one hop neighbourhood
2: **if** $\tau^{best} \neq \{\}$ **then**
3: $Merge\_With(\tau_i, \tau^{best})$ //merge the two tokens
4: **else**
5: $Move\_Token(\tau_i)$ //continue to move the token randomly
6: **end if**

Figure 3 illustrates this improvement. In this instance, merging of two trusted spanning trees occurs where tokens are at nodes with highest trust level resulting in a larger and more robust trusted spanning tree than in Figure 2.

## V. Simulation of Trusted Spanning Trees in DTMs

Classical DTM applications include Military Ad-Hoc Networks, Vehicle Ad-Hoc Network (VANETs), Exotic Media Networks, and etc. Suitable networks for simulation of any MANET ought to comprise lay-out of nodes (citizens), environmental properties and radio propagation (communication link) which reflect real-world situations. The networks used in this work were generated by Madhoc [15] (an ad-hoc networks simulator that provides mobility models allowing realistic motion of citizens in variety of environments). Simulations in this work are divided into two main categories, static and dynamic. In each category, two different characteristic networks are selected. To ensure validity of the simulation, three different networks of each characteristic are generated. Altogether, twelve networks were selected, Table 1 summarizes the networks and their characteristics in each category used in this work.

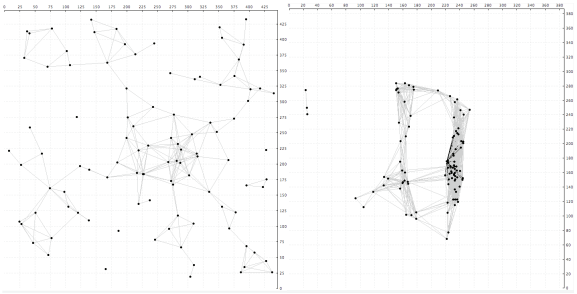
| **Network Category** | | | |
|---|---|---|---|
| ***Static*** | | ***Dynamic*** | |
| ***Random*** | ***City Street*** | ***Shopping Mall*** | ***Highway*** |
| random 1 | street 1 | mall 1 | highway 1 |
| random 2 | street 2 | mall 2 | highway 2 |
| random 3 | street 3 | mall 3 | highway 3 |

Fig. 4. Summary of graphs used according to its characteristic

Properties of each type of networks is discussed in the following sub-sections.

### *A. Static Networks*

While dynamic networks are more appropriate in simulation of this type, the use of static networks was not be overlooked as they provided good starting point to investigate flaws and short comings of proposed algorithms. Random and city street scenarios were selected for static networks. Figures 5(a) and 5(b) depict examples of random and city street networks respectively. Tables I and II summarize properties of each networks in the static category.



(a) Random. (b) City Street.

Fig. 5. Examples of Random and City Street Networks

### *B. Dynamic Networks*

The duration consists of 40 simulation steps, a simulation step was taken at 0.25 seconds interval. A duration in a

TABLE I
PROPERTIES OF RANDOM NETWORKS (STATIC)

| | *random 1* | *random 2* | *random 3* |
|---|---|---|---|
| number of stations | 100 | 100 | 100 |
| average number of degrees | 4.50 | 4.86 | 4.22 |
| maximum number of degrees | 10 | 15 | 10 |
| minimum number of degrees | 0 | 0 | 0 |
| total connections | 331 | 356 | 331 |

TABLE II
PROPERTIES OF CITY STREET NETWORKS (STATIC)

| | *street 1* | *street 2* | *street 3* |
|---|---|---|---|
| number of stations | 100 | 100 | 100 |
| average number of degrees | 28.12 | 32.16 | 43.05 |
| maximum number of degrees | 50 | 57 | 65 |
| minimum number of degrees | 1 | 1 | 1 |
| total connections | 1,408 | 1,585 | 2,158 |

TABLE III
PROPERTIES OF SHOPPING MALL NETWORKS (DYNAMIC)

| | *mall 1* | *mall 2* | *mall 3* |
|---|---|---|---|
| number of stations | 99 | 99 | 99 |
| average number of degrees | 7.13 | 7.68 | 7.69 |
| average of max. number of degrees | 13.20 | 16.10 | 14.83 |
| average of min. number of degrees | 2 | 1.07 | 1 |
| average number of total connections | 344.49 | 371.17 | 371.27 |

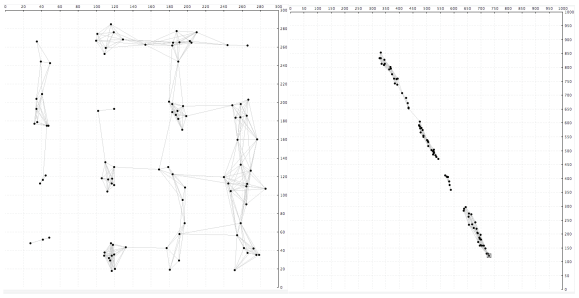
TABLE IV
PROPERTIES OF HIGHWAY NETWORKS (DYNAMIC)

| | *highway1* | *highway2* | *highway3* |
|---|---|---|---|
| number of stations | 80 | 80 | 80 |
| avg. number of degrees | 11.07 | 11.51 | 39.61 |
| avg. of max. number of degrees | 19.10 | 19.41 | 55.34 |
| avg. of min. number of degrees | 2.61 | 2.39 | 19.44 |
| avg. number of total connections | 433 | 448.60 | 1,547.05 |

simulation was set to last 10 seconds. The duration was selected carefully to reflect what may happen in practice. In reality, changes in a highway network are likely to occur more often than in a shopping mall network. Figures 6(a) and 6(b) depict initial configurations at $t0$ of the shopping mall and highway networks respectively. Tables III and IV summarize properties of each networks in the dynamic category.

## VI. SIMULATION RESULTS

As stated earlier in Section III, determination of an optimal spanning tree in MANETs is extremely difficult due to their dynamic and lack of global knowledge and no algorithm exists to date. Since the networks used in this work were generated by Madhoc, global knowledge and changes in their dynamicity could be predetermined. Therefore, optimal spanning trees can also be obtained. This advantage makes it possible for the efficiency of the two algorithms, the three cost functions proposed and robustness of spanning trees generated in this work to be evaluated. Hereafter, all values which can be predetermined from optimal spanning trees will be referred to as optimal value for ease of reference. In order to ensure validity of the study, 375 runs were carried out for each of the four networks. Referring to the number of stations in each communication graph used in this simulation which is state in the previous section, the overall average percentage of trusted



(a) Shopping Mall. (b) Highway.

Fig. 6. Example of Shopping Mall and Highway Networks at $t0$

nodes in simulations is 24. Their findings are discussed below:

### A. Results from Static Networks

Tables V and VI summarize the averages for the three cost function values for the optimal trusted spanning trees and those generated by T-DA-GRS and T-GDA-GRS for random and city streets networks respectively.

TABLE V
AVERAGES OF COST FUNCTION VALUES (RANDOM NETWORKS)

| | *weight* | *weight_penalty* | *isolating_low_trusted_node* |
|---|---|---|---|
| Optimal Value | 632.73 | 623.63 | 97.07 |
| T-GDA-GRS | 577.60 | 554.07 | 65.01 |
| T-DA-GRS | 518.73 | 489.13 | 48.32 |

TABLE VI
AVERAGES OF COST FUNCTION VALUES (CITY STREET NETWORKS)

| | *weight* | *weight_penalty* | *isolating_low_trusted_node* |
|---|---|---|---|
| Optimal Value | 747.40 | 743.17 | 99.20 |
| T-GDA-GRS | 717.30 | 693.60 | 87.05 |
| T-DA-GRS | 546.07 | 508.07 | 49.88 |

For both static networks, T-DA-GRS achieved the average of 77.15% for $weight$ value, 72.96% for $weight\_penalty$ value and 50.03% for *isolating_low_trusted_node* value of their corresponding optimal values. These values are 93.82%, 91.28% and 77.47% for T-GDA-GRS respectively. For both dynamic networks, T-DA-GRS achieved the average of 56.24% for $weight$ value, 53.16% for $weight\_penalty$ value and 46.78% for *isolating_low_trusted_node* value of their corresponding optimal values. These values are 67.23%, 65.12% and 73.53% for T-GDA-GRS respectively.

### B. Results from Dynamic Networks

Tables VII and VIII summarize the averages for the three cost function values for the optimal trusted spanning trees and those generated by T-DA-GRS and T-GDA-GRS for shopping mall and highway networks respectively.

TABLE VII
AVERAGES OF COST FUNCTION VALUES (SHOPPING MALL NETWORKS)

| | *weight* | *weight_penalty* | *isolating_low_trusted_node* |
|---|---|---|---|
| Optimal Value | 776 | 769.03 | 97.6 |
| T-GDA-GRS | 643.80 | 619.97 | 75.84 |
| T-DA-GRS | 537.03 | 504.07 | 48.12 |

TABLE VIII
AVERAGES OF COST FUNCTION VALUES (HIGHWAY NETWORKS)

| | *weight* | *weight_penalty* | *isolating_low_trusted_node* |
|---|---|---|---|
| Optimal Value | 972 | 968.10 | 99.8 |
| T-GDA-GRS | 531.45 | 511.21 | 69.31 |
| T-DA-GRS | 446.13 | 419.37 | 44.23 |

### C. Discussion

While both T-DA-GRS and T-GDA-GRS could not yield optimal performances, trusted spanning trees generated by both algorithms were comparable to optimal ones and ought to be applicable in practice. In all three aspects of cost functions, T-GDA-GRS yielded superior performances and hence it can be concluded that T-GDA-GRS is an improvement to T-DA-GRS. The interesting value of *isolating_low_trusted_node* in both types of networks emphasize the improvement of T-GDA-GRS over T-DA-GRS. While the T-GDA-GRS algorithm need only another 25% to reach the optimal value, the other algorithm has up to 50% to improve. Another interesting information lies in the optimal value crossing with *isolating_low_trusted_node* function from both table showing us that the optimal value cannot isolate all non-trustable node to terminal position, the value is not 100%. The loss number happens because of *'articulate node'*, the node in which removal can create a disconnected graph. In order to connecting through a network partition, the algorithms cannot isolate these articulate nodes to leaf of the tree. Hence, this articulate node can be seen as a point of failure in a partition which cannot be avoid in the assumption of this study. In static networks, more efficient trusted spanning trees in city street networks could be generated than in random networks. This is due to the fact that city street networks, in general, are more dense than random networks. In dynamic networks, density of the networks was not an important factor in determination of effective trusted spanning trees as dynamicity has more direct influence. A crucial factor which prevents maintaining efficient trusted spanning trees is that existing ones cannot be altered or modified unless break away of node(s) or merge(s) with other trees occur. This suggests that trusted spanning trees in a MANET may need to posses an ability to adapt and learn from their experience and local knowledge under the assumption that global knowledge cannot be assumed.

## VII. CONCLUSION AND FUTURE WORK

As trust management in MANETs receives much attention recently due to its immense application and is an active research area, management of DTMs is even more problematic, and in turn, presents a greater challenge. While algorithms to manage trusted spanning trees proposed in this work may not be comprehensive and several other aspects are yet to be considered, they present a good starting point in trust management in DTMs. This work affirms the advantage of greedy strategy and demonstrates its benefit in spanning trees traversal in such networks where less trustable nodes are at leaves. Ability to assess and select effective spanning trees among many possibilities is crucial, this work proposes a method to assess them by means of three different cost functions. Their use may be beneficial in similar applications.

Future work can be carried out in several facets. T-DA-GRS and T-GDA-GRS merit further investigation. They can be tested on other types of network models. Good candidates of these are city traffic and city centre models as they may reveal some useful properties. As discussed in the previous section, an ideal trusted spanning tree is the one which can adapt itself for better efficiency. This suggests that the solution lies in invention of an adaptive trust management algorithm that is capable of learning from experience and local knowledge even under the assumption that global knowledge cannot be assumed because of dynamicity and decentralized nature of DTMs.

## ACKNOWLEDGEMENT

The authors gratefully acknowledge the financial support from Ministére de la Recherche, réf BFR05/037, Grand-Duche de Luxembourg and the Ministry of Science and Technology, Thai Government, for Miss A. Piyatumrong throughout the period of this research.

## REFERENCES

[1] K. Fall, "A delay tolerant network architecture for challenged internets," *Proceedings of ACM SIGCOMM 2003, Computer Communications Review*, vol. Vol 33, August 2003.

[2] A. Casteigts and S. Chaumette, "Dynamicity aware graph relabeling systems (da-grs), a local computation based model to describe manet algorithms," *International Conference on Parallel and Distributed Computing Systems*, pp. 231–236, November 2005.

[3] A. Casteigts, "Model driven capabilities of the da-grs model," *ICAS '06: Proceedings of the International Conference on Autonomic and Autonomous Systems*, p. 24, 2006.

[4] L. Hogie, *Mobile Ad Hoc Networks: Modelling, Simulation and Broadcast-based Applications*. PhD thesis, Univerity of Le Havre, University of Luxembourg, April 2007.

[5] A. W. J. David Lewis, "Trust as a social reality," *Social Forces*, vol. 63, pp. 967–985, June 1985.

[6] T. D. Huynh, N. R. Jennings, and N. R. S. Shadbolt, "An integrated trust and reputation model for open multi-agent systems," *Autonomous Agents and Multi-Agent Systems*, vol. 13, pp. 119–154, September 2006.

[7] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins, "Propagation of trust and distrust," *International World Wide Web Conference (WWW2004)*, 2004.

[8] S. Buchegger and J.-Y. L. Boudec, "Performance analysis of the CONFIDANT protocol: Cooperation of nodes — fairness in dynamic ad-hoc networks," in *MobiHOC*, IEEE, June 2002.

[9] S. David and T. J. Pinch, "Six degrees of reputation: The use and abuse of online review and recommendation systems," *First Monday*, vol. 11, March 2006.

[10] M. Seredynski, P. Bouvry, and M. A. Klopotek, "Preventing selfish behavior in ad hoc networks," in *Congress on Evolutionary Computation (CEC 2007)*, pp. 3554 – 3560, IEEE Computer Society, September 2007.

[11] S. Sukumaran and R. E. Blessing, "Reputation based localized access control for mobile ad-hoc networks," in *ADHOC-NOW*, Lecture Notes in Computer Science, pp. 197–210, Springer, 2006.

[12] Q. He, D. Wu, and P. Khosla, "Sori: A secure and objective reputation-based incentive scheme for ad-hoc networks," *WCNC2004*, 2004.

[13] G. F. Marias, P. Georgiadis, D. Flitzanis, and K. Mandalas, "Cooperation enforcement schemes for manets: a survey: Research articles," *Wirel. Commun. Mob. Comput.*, vol. 6, no. 3, pp. 319–332, 2006.

[14] J. V. D. Merwe, D. Dawoud, and S. McDonald, "A survey on peer-to-peer key management for mobile ad hoc networks," *ACM Comput. Surv.*, vol. 39, no. 1, p. 1, 2007.

[15] L. Hogie, F. Guinand, and P. Bouvry, "The madhoc metropolitan adhoc network simulator." http://litis.univ-lehavre.fr/∼hogie/madhoc/.