Placement of Control Network for Mobile Agents over Opportunistic Networks

Yao-Nan Lien
Dept. of Computer Science
National Chengchi University, Taiwan, ROC
lien@cs.nccu.edu.tw

Yi-Shiuan Lin
Dept. of Computer Science
National Chengchi University, Taiwan, ROC
g9508@cs.nccu.edu.tw

***Abstract*—**Transmitting data on an opportunistic network is much more difficult than that on a conventional network. We propose to use mobile agent technology to enhance its message transfer efficiency. A mobile agent platform requires a search mechanism to control its agents. We investigated the application of mobile agent on the opportunistic network characterized by "CenWits Search and Rescue System" applied in YuShan National Park as well as proposed a control network technology to support rapid agent search. Under different objectives and constraints, the control network deployment problem is formulated into several placement models. After proving them to be NP-hard, we designed several simple but efficient heuristic algorithms to solve the placement problems. A simple agent search algorithm based on the designed control network was designed to support rapid agent search, too.

***Keywords*—Opportunistic Network, Mobile Agent**

I. Introduction

Data transmission over opportunistic networks is not only much slower but also more unpredictable than conventional networks. Applying mobile agent technology on opportunistic network will be able to enhance its message transfer efficiency as well as to enable more applications, as a consequence. However, a functional mobile agent system demands an agent tracking mechanism to facilitate the control of mobile agents such as termination, suspension, and resuming of a mobile agent. The control network proposed in this paper will be able to fulfill this demand.

1.1 Opportunistic Network

Opportunistic Network, OppNet [1], is a special case of Delay-Tolerant Network (DTN), which may lack continuous network connectivity. In general, a DTN uses so called *store-carry-forward* mechanism to transfer messages. Typical examples [1,2,3] are hiker tracking, wild-field animal tracking, battle field networking, etc. Since the message transfer on OppNet is slow and unpredictable, some network protocols such as TCP may suffer from severe performance degradation. For instance, a typical TCP protocol requires an acknowledgement message to be feedback to the sender within a predefined time period. Otherwise, the sender will take the network to be congested and will activate congestion control actions which may slow down or even stop the operation of TCP. Therefore, OppNet demands a more sophisticate message transfer mechanism other than store-carry-forward to prevent the network from interferenced by inappropriate congestion control or other protocol operations.

1.2 Mobile Agent

A *mobile agent* is a composition of computer software and data which is able to migrate from one computer to another autonomously and continue its execution on the destination computer [4,6]. When an agent leaves its home node, it acts as a delegate of the originator and is able to make decision by itself based on the built-in logic and the information it collected from the network. On the completion of its assigned task, it returned to the originator to deliver the result. Due to the ability to make decision autonomously, mobile agent technology is a good candidate to provide message transfer service for OppNet. Nevertheless, a mobile agent platform demands a fast search mechanism to facilitate the control of mobile agents. Based on a special OppNet, *CenWits system* [3], this paper proposes a control network framework as well as an associated search mechanism to facilitate rapid agent search.

1.3 CenWits System

CenWits Search and Rescue System is a special Wireless Sensor Network (WSN) that uses loosely coupled connection and witnesses between wireless sensor nodes (i.e., OppNet) to track locations of moving objects in the wilderness areas [3]. CenWits is comprised of GPS enabled mobile, in-situ sensors that are worn by subjects such as people and wild animals as well as access points (AP) that collect information from these sensors via wireless signals. CenWits records subjects' location/movement information as well as environment informaiton (e.g. weather) and conveys to the outside world. The system had been deployed in some places such as Rocky Mountain (US), Yosemite National Parks (US), and YuShan National Park (Taiwan). A typical application of this system is to narrow down the search area based on the recorded location and movement information once a rescue operation is launched for a lost hiker.

Conventional cellular networks may not be useful in the areas mentioned above due to the poor signal coverage. First, propagation of radio signals in mountain area is often interferenced by wavy terrain. Secondly, low user popularity and high deployment cost couldn't motivate cellular operators to increase the density of cell towers. Therefore, in such a special OppNet, a special short range wireless communication mechanism such as ZigBee is used to support node-to-node communications.

In the system deployed in YuShan National Park, called *YushanNet*, every participating hiker carries a sensor node, called *mobile node* (*MN*) in this paper, which is designed to collect time, positions, and demanded environment information periodically. When two mobile nodes meet together in the park, they exchange the collected information to each other. When a mobile node meets an AP, it delivers all collected information to the AP, which then forwards the information to the headquarter through a high speed network. The entire system forms an OppNet. This paper takes this system as the reference system.

1.4 Challenges of Mobile Agent on OppNet

In YushanNet, each mobile node is carried by a hiker so that its moving behavior is actually the same as the moving behavior of its hosting hiker. Assuming hikers are walking in similar speeds, mobile agents are not only moving slowly, but also having difficulty to hop forward. As a consequence, neither moving nor searching a mobile agent is an easy task. Taking YushanNet as an example, we illustrate this difficulty using the example shown in Fig. 1. Assuming a search agent is launched to search another agent that left the starting point one day earlier, it is very difficult for the search agent to reach the target via forward hopping because all intermediate nodes are walking in similar speeds.

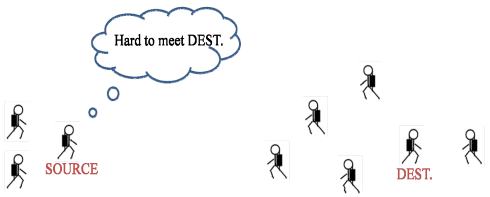

Fig. 1. Searching a mobile agent on OppNet

A real world system like YushanNet demands a fast agent search mechanism to support its control operations as well as urgent tasks, such as warning a hiking team member about a severe weather condition. However, most conventional agent search mechanisms are designed for conventional networks which are quite different from OppNet. Therefore, we propose to use a separated control network overlaid on top of a conventional high speed IP network as well as a search mechanism to facilitate rapid agent search on OppNet. A search agent can move to a control point near the target rapidly via the control network and then hop to a mobile node which then moves in walking speed toward the target.

Section 2 will discuss current mobile agent search technology. Section 3 will discuss the concept of control network and its optimal deployment model. Section 4 will analyze problem complexity and present several heuristic algorithms. The evaluation will be presented in Section 5.

II. Related Work and Challenges

Mobile agent search technology has been studied for years [4]. They can be roughly classified into two categories: report-based and non-report-based. In a report-based system, every mobile agent has to report its locations periodically to the originator so that its current location can be easily determined. On the other hand, in a non-report-based system, a mobile agent doesn't report its locations such that a special search mechanism is needed to estimate its current location. All of them only consider the mobility of agents, but not the mobility of hosting devices. Therefore, they are not adequate for OppNet for the following reasons:

- Message transfer on OppNet requires longer delay time such that reported agent locations are usually outdated turning a search operation into a hide-and-seek game.
- The prior knowledge of the path and schedule a mobile agent took may not be useful in estimating the current location of a mobile agent because its hosts are moving too.

Therefore, it is difficult to calculate or to estimate the exact location of a mobile agent.

III. Control Network for Mobile Agent

3.1 Basic Concept of Control Network

We assume that there is a conventional high speed IP network (e.g. Internet) deployed in the concerned area. We can construct a control network by installing *control points* (CP) over the concerned area and connecting them together using the IP network. (Actually, control points can also be embedded in YushanNet's APs.) A search agent can move rapidly on this network to the control point nearest to the target, called *Egress Control Point,* and then hop to a mobile node that is moving toward the target. The above procedure is illustrated in Fig. 2.

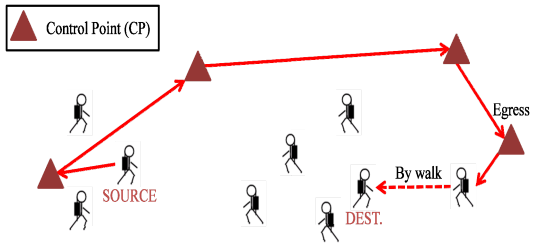

Fig 2. Use Control Network to assist agent search

3.2 Agent Search Based on Control Network

The following is a simple agent search algorithm, which is designed by modifying the Basic Binary Search (BBS) algorithm we designed for conventional networks [4]:

1. Sort the CPs along the path that the target agent took to create CPlist.
2. Visit the CP in the middle of CPlist. If the target had visited the selected CP, then delete the first half of CPlist. Otherwise, delete the other half.
3. Repeat Step 2 until either the target can be reached directly from the visited CP or only one CP remained on CPlist. Take it as the Egress CP.
4. If the target can be reached directly from Egress CP, then hop the search agent to the target. Otherwise, hop to a mobile node that is moving toward the target.

The selection of Egress CP may be inaccurate such that the above search procedure may have to be repeated several times to locate the target. Therefore, there is a need to design a more sophisticated search algorithm to improve the accuracy of search algorithms. It is beyond the scope of this paper, though.

3.3 Deployment of Control Network

To deploy a control network under resource constraint is a typical combinatorial optimization problem. We propose several optimization models addressing to different objectives based on the OppNet of YushanNet style.

3.3.1 Assumptions of Experiment Environment

Based on YushanNet style, we assume:

1. Each hiker moves in walking speed along a pre-determined trail.
2. All hikers move in approximately the same speed.
3. The deployment cost of a CP is location dependent.
4. Most candidate CPs are located at the end points and intersections of trails. They can also be deployed in the middle of a trail if the trail is too long.

3.3.2 Design Considerations and Objectives

Because resources are limited and deployment cost is location dependent, the selection of CPs to maximize the design objective becomes a combinatorial optimization problem, called *Control Point Selection Problem* (CPSP).

The best design objective might be "the average time to locate an agent". Thus, we define *MC_distance* (MCD) to be the distance (time) between a target agent and its Egress CP, which is the CP to be visited by the target agent in the nearest future.

Although MC_distance is a good performance index, it is very difficult to be formulated in mathematical format. Therefore, we designed several optimization models addressing to different objectives. It is up to the user to select the model that is the most appropriate with respect to his/her own need. Furthermore, all models have a common available resource constraint.

3.4 Maximum Flow Model

In *Maximum Flow Model* (*CPSP-Flow*), the number of hikers entering a trail per time unit is a known constant. The objective is to maximize the total amount of hiker flows passing through the selected CPs:

Given a graph $G=(V,E)$, where

- $V=\{v_1,\ldots,v_n\}$ is the set of possible nodes for building control points,
- $E=\{e_{ij}\,|\,v_i,v_j\in V\}$ is the set of edges (trail segments) between v_i and v_j,
- $W=\{w_i\,|\,v_i\in V\}$ is the set of the cost (required resources) to build a control point on v_i,
- C is the total available resources,
- $T=\{t_1,\ldots,t_m\}$ is the set of hiking trails, each of which is a path, a sequence of nodes, and
- $F=\{f_k\,|\,t_k\in T\}$ is the set of hiker flows, which is the number of hikers per time unit passing a trail,

the *CPSP-Flow Problem* is to find the set of nodes $S\subseteq V$, to

maximize $\sum_{i\in V}\sum_{i\in t_k} f_k x_i$,

subject to

$\sum_{i\in V} w_i x_i \le C$, where $x_i=\begin{cases}1 & \text{if } v_i\in S,\\ 0 & \text{otherwise.}\end{cases}$

The advantage of CPSP-Flow model is its simplicity. However, it may lead to an uneven CP distribution. A good CSPS-Flow algorithm will inevitably choose the trails that have large hiker flows and neglect others. The mobile nodes that are hiking on less popular trails may never get a chance to meet any CP.

3.5 Maximum Coverage Model

To solve the uneven distribution problem, we modify CPSP model as follows:

1. Each trail must have at least one CP if resource is sufficient.
2. CPs have to be evenly distributed to the geographic areas.

However, since graph is composed of nodes and edges carrying no geographic information, the objective of *Maximum Coverage Model* (*CPSP-Coverage*) is set to maximize the number of covered edges, instead. The model is as follows:

Given a graph $G=(V,E)$, where

- the definition of V,E,W,C,T are the same as V,E,W,C,T in CPSP-Flow Problem, and
- $F=\{f_{ki} | v_i \in V, t_k \in T\}$ is a passing_trail matrix, where $f_{ki} = \begin{cases} t_k & \text{if trail } t_k \text{ passes node } v_i, \\ 0 & \text{otherwise,} \end{cases}$

the *CPSP-Coverage Problem* is to find $S \subseteq V$ to

maximize $\left|\bigcup\{e_{ij} \cdot x_i\}\right|$,

subject to

$\bigcup\{f_{ki} \cdot x_i\} = T$ and $\sum_{i \in V} w_i x_i \le C$, where

$x_i = \begin{cases} 1 & \text{if } v_i \in S, \\ 0 & \text{otherwise.} \end{cases}$

In this model, f_{ki} indicates whether node v_i is included in trail t_k; $\bigcup\{f_{ki} \cdot x_i\}$ is the union of the trails that pass node v_i; and $\left|\bigcup\{e_{ij} \cdot x_i\}\right|$ is the union of the edges covered by selected nodes.

This model is set to maximize edge coverage. All edges have the same unit weight. However, in reality, trail segments may have different weights. For instance, hikers may be easier to get lost at some trail segments. It makes more sense to raise the priority of the nodes near such trail segments. Therefore, the model is refined by taking edge priority into account in next section.

3.6 Maximum Utility Model

Maximum Utility Model (*CPSP-Utility*) assumes each edge has a weight representing the priority of a trail segment. We first define the total profit (utility) for a given set of nodes, *M,* as follows:

$$P(M) = \{ \Sigma\ z_{ij} \mid e_{ij} \in M \}$$

In above equation, z_{ij} is the weight of edge e_{ij}. If *M* is the selected nodes, *P*(*M*) is the total profit, the optimization objective. The model is defined as follows:

Given a graph $G=(V,E)$, where

- the definition of V,E,W,C,T,F are the same as V,E,W,C,T,F in CPSP-Coverage Model, and
- $Z = \{ z_{ij} \mid v_i, v_j \in V \}$ is the set of edge weights,

the *CPSP-Utility Problem* is to find $S \subseteq V$, to

maximize $\mathrm{P}\left(\bigcup\{e_{ij} \cdot x_{ij}\}\right)$,

subject to

$\bigcup\{f_{ki} \cdot x_i\} = T$ and $\sum_{i \in V} w_i x_i \le C$, where

$x_i = \begin{cases} 1 & \text{if } v_i \in S, \\ 0 & \text{otherwise.} \end{cases}$

IV. Complexity Analysis and Solutions

We can easily transform CPSP-Flow problem into a NP-Hard 0-1 knapsack problem by letting the deployment cost and the total amount of flows passing a node to be the weight and the profit of an object respectively in the knapsack problem. 0-1 knapsack problem has a good heuristic solution, which is to select objects in descending order of profit density. The other two problems can also be easily proved NP-hard. The details are shown in [5].

To solve CPSP-Coverage problem, we propose a heuristic algorithm, called *Node Potential First* (*NPF*). As shown in Fig 3.

Candidate ListV ← all nodes
Iterate until **Committed Cost >= Available Resources**
{
1. Delete the nodes from the candidate ListV if cost[node] > Remaining Resources.
2. Select the node from the candidate ListV with the largest value of edge_potential[node];
   i. If (tie) Then select the node with the largest value of trail_potential[node];
   ii. If (tie) Then select the node with the smallest value of cost[node].
3. Mark the edges and trails that are connected to the selected node to "covered"
4. Update the value of edge_potential[node] and trail_potential[node] of all nodes in ListV

}
# edge_potential [node] is the (no. of unmarked edges/cost) w.r.t. the node
# trail_potential[node] is the no. of unmarked trail w.r.t. the node

Fig. 3. Node Potential First (NPF) Algorithm

To solve CPSP-Utility problem, we modify NPF into another algorithm, *NPF-U*, by adding edge weights to the formula of edge_potential. The new formula is the total weight of unmarked edges divided by deployment cost of a node.

V. Performance Evaluation

The performance of three models and their solution algorithms (0-1 knapsack, NPF, and NPF-U) were evaluated using simulative experiments on a regular personal computer.

5. 1 Evaluation Metrics

Table 1 illustrates the evaluation metrics used in our experiments. Among them, MC_distance (MCD) is a unified metric to evaluate the adequacy of three models because their objectives are all different. To compute the average MCD of a control network placement, we assume that there are two mobile nodes moving in opposite directions on every edge. Thus, the average MCD is the average distance (time) from a mobile node to its nearest CP over all mobile nodes.

Table 1. Evaluation Metrics for CPSP

Name	Definition
flow coverage	total flows passing the selected CPs
edge coverage	total no. of edges connected to the selected CPs
weighted edge coverage	total weight of the edges connected to the selected CPs
mean_MCD	average MCD over all mobile nodes
mean_weighted_ MCD	average weight_MCD over all mobile nodes

Note that minimization of mean_MCD is not the objective of all three models such that the mean_MCD calculated by a heuristic solution may be smaller than that is calculated by an optimal solution. Further, the granularity of MCD is an edge, which is quite large since it may require several hours to walk through an edge.

In each of following experiments, problem instances were randomly generated using the parameters listed in Table 2.

Table 2. Parameters for Problem Generation

Parameter	Small Scale	Large Scale Node Scale Sensitivity	Large Scale Trail Scale Sensitivity
Number of candidate CPs	20	10~100	10
Number of trails	8	8	4 ~ 40
Available resources	12	12	12
CP deployment cost	1 ~ 5	1 ~ 5	1 ~ 5
Edge weight	1 ~ 5	1 ~ 5	1 ~ 5
Flow size on trail	N.A.	1 ~ 10	1 ~ 10

5.2 Evaluation of Small Scale Problems

Because both CPSP-Coverage and CPSP-Utility are NP-hard problems, we evaluated three algorithms against optimal solutions only in small scale problems. Five problem instances were randomly generated using the parameters listed in the second column of Table 2.

Two indices are defined: *Diff* is the difference between heuristic and optimal solutions and *Error* is the ratio of *Diff* over optimal solution. The result of CPSP-Coverage experiment is that, in all instances but instance 4, heuristic solutions are the same as optimal solutions. In instance 4, edge_coverage of NPF is less than the optimal solution by only 1, or 2.22% and mean_MCD is only 0.02 more edge (or 1.25%) than the optimal solution. The result of CPSP-Utility experiment is that, in all instances but instance 3 and 5, heuristic solutions are the same as optimal solutions. In instance 3 and 5, the maximum errors of NPF-U in weighted_edge_coverage and mean_MCD are no more than 3.74% and 0.6%, respectively.

5.3 Evaluation of Large Scale Problems

We randomly generated many large scale instances to evaluate PDF and PDF-U. Because optimal solutions are difficult to compute, we only evaluated their sensitivities to the node scale and to the trail scale.

5.3.1 Experiments of Node Scale Sensitivity

In this experiment, 50 graphs were randomly generated with the number of nodes between 10 and 100. The parameters are listed in the third column of Table 2.

0-1 knapsack algorithm performs poorly in avg. edge-coverage, avg. weighted edge coverage, avg. mean_MCD, and avg. mean_weighted_MCD. On the other hand, NPF performs better in CPSP-Coverage model and NPF-U performs better in CPSP-Utility model. These results are consistent with the original goals of these models. Both indices increase proportional to the number of nodes. Due to the space limit, only part of results is shown in Fig. 4 and 5.

From experimental results we can conclude that CPSP-Utility model is better if the importance of trail segment is taken into consideration. Otherwise, CPSP-Coverage will be better than the other two models.

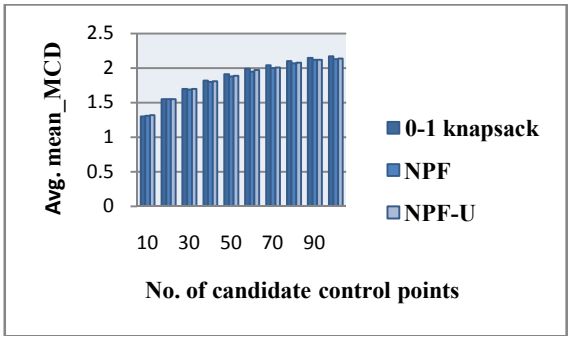

Fig. 4. Node Scale Sensitivity (avg. mean_MCD)

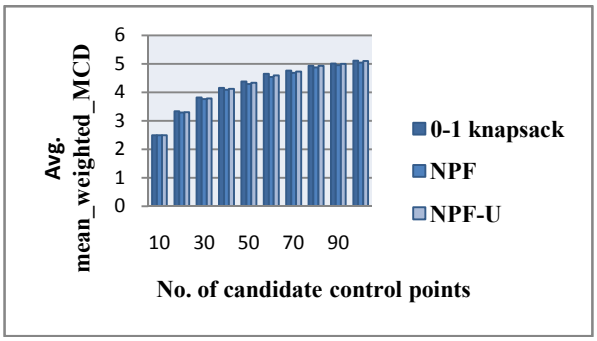

Fig. 5. Node Scale Sensitivity (avg. mean_weighted_MCD)

5.3.2 Experiments of Trail Scale Sensitivity

In this experiment, 50 graphs were randomly generated with the number of trails between 4 and 40. The parameters are listed in the fourth column of Table 2.

The results show that the all three algorithms perform very closely in terms of avg. edge coverage. NPF-U outperforms others in terms of avg. weighted edge coverage. Both indices increase proportional to the number of trails. Both avg. mean_MCD and avg. mean_weighted_MCD decreases gradually in all three algorithms as the number of trails increases. The performance of 0-1 knapsack is the best probably due to an increase in the number of intersections as the number of trails increases such that 0-1 knapsack model will favor intersections to build CP, which in turn will reduce mean_MCD.

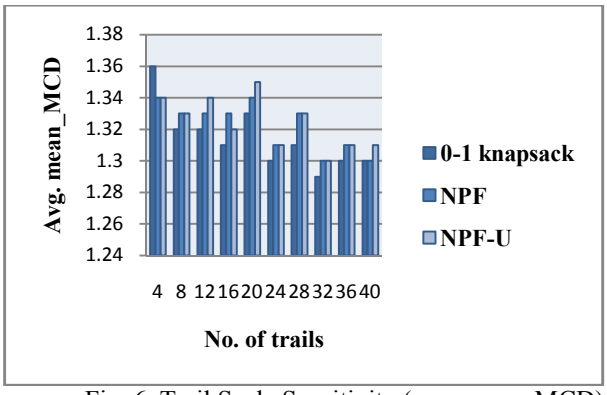

Fig. 6. Trail Scale Sensitivity (avg. mean_MCD)

VI. Concluding Remarks

Transmitting data on an opportunistic network is much more difficult than that on a conventional network. We propose to use mobile agent technology to enhance its message transfer efficiency. We investigated the application of mobile agent on the opportunistic network characterized by "CenWits Search and Rescue System" applied in YuShan National Park. We propose to construct a control network using a high speed IP network for search agents to travel in high speed. Under different objectives and constraints, we propose several placement models for the placement of control network. After proving them to be NP-hard, we propose several simple but efficient heuristic algorithms to solve them. The simulative experiments show that these models can facilitate rapid agent search and the proposed algorithms are quite efficient. In our experimental environment, the average distance, or hiking time, from a mobile node to the nearest control point is no more than two trail segments.

In the future, there is a need to design a more sophistiacted search algorithm based on the proposed control network to improve search accuracy.

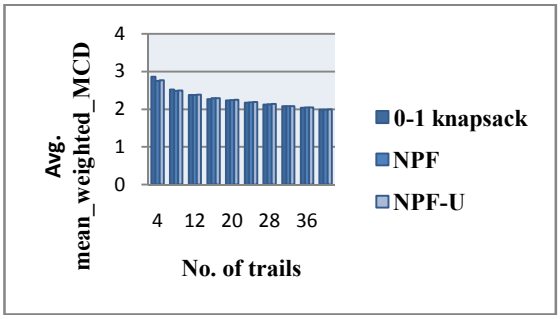

Fig. 7. Trail Scale Sensitivity (avg. mean_weighted_MCD)

References

1] C.-M. Huang, K.-c. Lan, and C.-Z. Tsai, "A survey of opportunistic networks," in *Proc. of the 22nd International Conference on Advanced Information Networking and Applications*, 2008, pp. 1672-1677.

[2] Yu-Te Huang, Yi-Chao Chen, Jyh-How Huang, Ling-Jyh Chen, Polly Huang, "YushanNet: A Delay-Tolerant Wireless Sensor Network for Hiker Tracking in Yushan National Park," *Proc. of Int'l conf. on Mobile Data Management (MDM'09)*, Taipei, Taiwan, 2009.

[3] J. H. Huang, S. Amjad, and S. Mishra, "Cenwits: a sensor-based loosely coupled search and rescue system using witnesses," in *Proc. of the 3rd International Conference on Embedded Networked Sensor Systems*, 2005, pp. 180-191.

[4] Y. N. Lien and C. W. R. Leng, "On the search of mobile agents," in *Proc. of the 7th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, 1996, pp. 703-707.

[5] Yi-Shiuan Lin, "Mobile Agent Tracking Technology over Opportunistic Network", Master Thesis, Dept of Comp. Sci., National Chengchi Univ., June 2010.

[6] D. S. Milojicic, F. Douglis, and R. Wheeler. *Mobility: processes, computers, and agents*. New York, NY: ACM Press/Addison-Wesley Publishing Co., 1999.