A Flow-Aware Placement of Mobile Agent Control Network over Opportunistic Networks

Hsiao-Tzu You, Yao-Nan Lien
National Chengchi University, Taiwan, ROC
lien@cs.nccu.edu.tw

Jyh-Shyan Huang
Chunghwa Telecom, Taiwan, ROC
frank210@cht.com.tw

***Abstract*—**Transmitting data on an opportunistic network is much more difficult than that on a conventional network. We propose to use mobile agent 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. This paper enhances our previous control network placement models by taking into account the amount of agent flows. After proving it to be NP-hard, we designed a simple but efficient heuristic algorithms to solve the placement problem. In our simulation study, the new model outperforms our previous models easily.

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

I. Introduction

Data transmission over opportunistic network is not only much slower but also more unpredictable than conventional networks. We proposed to apply mobile agent technology on opportunistic network to facilitate its control operation [5]. 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. Our previously proposed control network technology will be able to fulfill this demand. This paper enhances our previous placement models by taking into account the amount of agent flows to achieve better services to mobile agents.

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 interfered by inappropriate congestion control as well as 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 cell tower densities. 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 a similar speed.

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 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*, then hop to a mobile node that is heading toward the target. The above procedure is illustrated in Fig. 2.

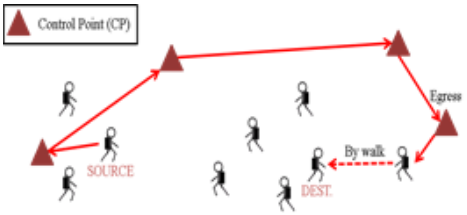

Fig 2. Use Control Network to assist agent search

3.2 Agent Search Based on Control Network

The paper [5] uses a simple agent search algorithm, which is designed by modifying the Basic Binary Search (BBS) algorithm we designed for conventional networks [4]. To improve search efficiency, we are designing a more efficient search algorithm by estimating agent's current location based on the prior statistics of traveling time over each trail segment. The details will not be discussed here because it is out of the scope of this paper.

3.3 Deployment of Control Network

To deploy a control network under resource constraint is a typical combinatorial optimization problem. In our previous research, we propose several optimization models addressing different objectives based on the YushanNet.

3.3.1 Environment Assumptions

Based on YushanNet, 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, in our previous paper [5] we designed several optimization models addressing to different objectives. This paper proposes a more comprehensive model. It is up to the user to select the model that is the most appropriate w.r.t. his/her own need. Furthermore, all models have a common available resource constraint.

3.3.3 Existing Placement Models

We proposed three placement models in our previous paper [5]. 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. CPSP-Flow model is proven to be isomorphic to the famous 0-1 Knapsack problem. 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.

To solve the uneven distribution problem, CPSP model was modified 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. All edges have the same unit weight. In reality, trail segments may have different weights. For instance, hikers may be easier to get lost on 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 model. *Maximum Utility Model* (*CPSP-Utility*) assumes each edge has a weight representing the priority of a trail segment. All three models are proven to be NP-hard. Heuristic algorithms (0-1 knapsack, NPF, and NPF-U) were proposed to solve them. Although all three models are useful, each of them only addresses one or two issues. A more comprehensive model that can address several issues is yet to be studied.

3.3.4 Enhanced Heuristic Algorithms

For comparison purpose, we proposes two algorithms, NPF2 and NPF-U2 for CPSP-Coverage and CPSP-Utility modes to improve the performances of NPF and NPF-U, respectively. Both NPF and NPF-U are greedy algorithms to select control nodes one by one based on two potential indices, node potential and tail potential. Node potential gets higher priority than trail potential, which is taken into consideration only when there is a tie on the node potential. NPF2 and NPF-U2 change the node selection rule into the linear combination of both indices. Thus both indices can be taken into account simultaneously. Only NPF-U2 is shown in Fig. 3, while NPF2 can be easily figured out from NPF.

Candidate ListV ← all nodes
Iterate until Committed Cost >= Available Resources
{
Delete the node from the candidate ListV with node_weight > Remaining Resources.
Select the node from the candidate ListV with the largest value of
α(total weights of unmarked edges /node cost)+ β(trail_potential[node]);
If tie Then select the node with the smallest value of cost[node].
Mark the edges and trails that are connected to the selected node to "covered".

Update the value of total weights of unmarked edges[node] and trail_potential[node] of every node by marking covered trails.
}
# trail_potential[node] is the (no. of unmarked trail /cost) w.r.t. the node.

Fig. 3. Node Potential First for Utility (NPF-U2) Algorithm

3.4 Maximum Utility-Flow Model

This paper proposes a more comprehensive CPSP model, *Maximum Utility-Flow Model* (*CPSP-UF*), which takes not only weighted edge coverage but also the amount of agent flows into account.

We first define the total profit for a given set of nodes, *M*, as follows:

$$P(M) = \{ \ \Sigma \ (z_{ij} \ \cdot \ f_k) \mid e_{ij} \in M', t_k \ \text{passes} \ e_{ij} \ \}$$

In above equation, z_{ij} is the weight of edge e_{ij} and M' is the set of edges connected to the selected nodes in *M*. $P(M)$ is then the total profit, the optimization objective. The model is defined as follows:

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

- $V=\{v_1,\ldots,v_n\}$ is the set of possible nodes for building control points,
- $E=\{e_{ij} \mid v_i, v_j \in V\}$ is the set of edges (trail segments) between v_i and v_j,
- $W=\{w_i \mid v_i \in V\}$ is the set of the cost (required resources) of building a control point on v_i,
- C is the available resources,
- $T=\{t_1,\ldots,t_m\}$ is the set of hiking trails, each of which is a path, a sequence of nodes, used for hiking, and
-
- $G=\{g_{ki} \mid v_i \in V, t_k \in T\}$ is a passing_trail matrix,

where $g_{ki} = \begin{cases} t_k & \text{if trail } t_k \text{ passes node } v_i \\ 0 & \text{otherwise,} \end{cases}$

- $Z=\{z_{ij} \mid v_i, v_j \in V\}$ is the weight of the edge e_{ij},
- $F=\{f_k \mid t_k \in T\}$ is the set of hiker flows, which is the number of hikers per time unit passing a trail,

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

maximize $P(S)$
subject to
$\bigcup \{g_{kj} \ \cdot \ x_i \} = T$ 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

Similar to our previous models, Maximum Utility-Flow Model can also be easily proven NP-Hard. To solve the problem, we propose a heuristic algorithm, called *Node Potential First for Utlity-Flow* (*NPF-UF*). As shown in Fig 4.

Candidate ListV ← all nodes
Iterate until Committed Cost >= Available Resources
{
Delete the node from the candidate ListV with node_weight > Remaining Resources.
Select the node from the candidate ListV with the largest value of
(α (total weights of unmarked edges /node cost)+ β (trail_potential[node]))*γ (flow_potential[node]);

If tie Then select the node with the smallest value of cost[node].
Mark the edges and trails that are connected to the selected node to "covered"

Update the value of node_potential and trail_hit_count of every node by marking covered trails.
}
# trail_potential[node] is the (no. of unmarked trail /cost) w.r.t. the node
# flow_potential [node] is the (no. of unmarked flow/ node_cost) w.r.t node

Fig. 4 Node Potential First for Utility-Flow (NPF-UF) Algorithm

V. Performance Evaluation

The performances of heuristic algorithms NPF-U, NPF-U2, NPF-UF 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) are unified metrics to evaluate the adequacy of the models because their objectives are all different. MCD of a mobile node is the hop counts from the node to the nearest control point. And, the average MCD is the average distance (time) from a mobile node to the nearest CP over all mobile nodes.

Mean_MCD = $$\frac{\sum_{\forall v_i \in V} MCD(v_i)}{|V|}$$

Table 1. Evaluation Metrics for CPSP

Name	Definition
flow coverage	total flows passing selected CPs
edge coverage	total no. of edges connected to the selected CPs
Weighted edge coverage	summation of the weight of the edges connected to the selected CPs
populated edge coverage	summation of the (no. of flow * 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 four 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	100
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 all CPSP models are NP-hard problems, we evaluate NPF-U, NPF-U2, and NPF-UF against optimal solutions on 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 results of NPF-U, NPF-U2, and NPF-UF experiments are showed in Table 3, 4, and 5.

Table 3. NPF-U vs. Optimal solutions

Instance	1	2	3
edge coverage	0	Diff = 2 Error = 1.01%	0
mean_ weighted_MCD	Diff = 0.08 Error = 2.48%	Diff = 0.07 Error = 2.22%	0
max_ weighted_MCD	Diff = 2 Error= 28.57%	Diff = 1 Error= 16.67%	0
Instance	4	5	
edge coverage	Diff = 4 Error = 1.34%	Diff. = 20 Err. =5.75%	
mean_ weighted_MCD	Diff = 0.01 Error = 0.34%	Diff = 0.01 Error = 0.33%	
max_ weighted_MCD	0	0	

Table 4. NPF-U2 vs. Optimal solutions

Instance	1	2	3
edge coverage	0	0	0
mean_ weighted_MCD	Diff = 0.05 Error = 1.55%	0	0
max_ weighted_MCD	Diff = 2 Error= 28.57%	0	
Instance	4	5	
edge coverage	Diff = 2 Error = 0.67%	Diff = 15 Error = 4.31%	
mean_ weighted_MCD	0	Diff = 0.01 Error = 0.33%	
max_ weighted_MCD	0	0	

Table 5. NPF-UF vs. Optimal solutions

Instance	1	2	3
populated edge coverage	Diff. =3 Err. = 0.4%	Diff. =9 Err. = 1.19%	Diff. = 6 Err. = 0.77%
mean_MCD	Diff. = 5.08 Err. = 24.78%	Diff. = 3.01 Err. = 11.24%	Diff. = 0.06 Err. = 0.2%
Instance	4	5	
populated edge coverage	Diff. = 11 Err. = 1.37%	Diff. = 10 Err. =1.23%	
mean_MCD	Diff. =4.34 Err. = 11.79 %	Diff. = 6.42 Err. = 17.5%	

5.3 Evaluation of Large Scale Problems

We randomly generated many large scale instances to evaluate NPF-U, NPF-U2 and NPF-UF. 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. Experiment results are showed in Fig 5, 6, 7, and 8.

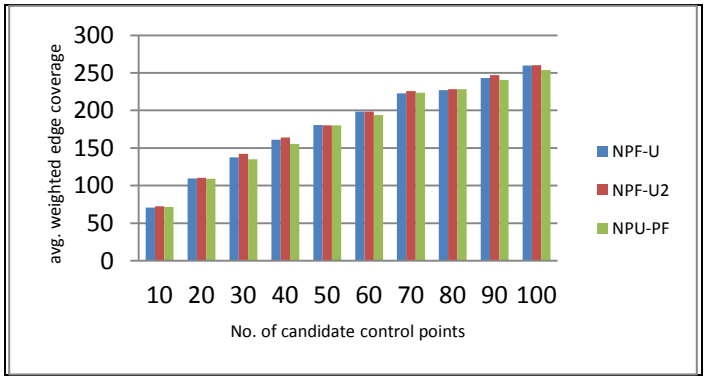

Fig. 5. Node Scale Sensitivity (avg. weighted edge coverage)

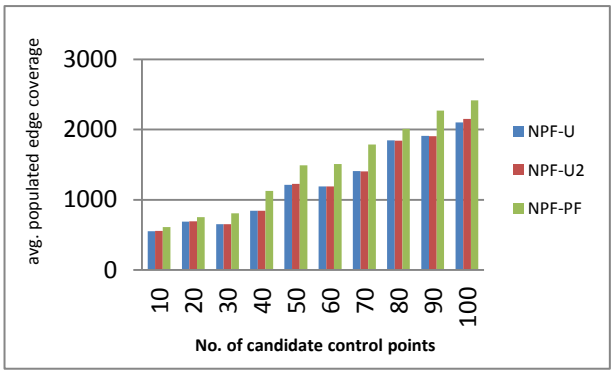

Fig. 6. Node Scale Sensitivity (avg. populated edge coverage)

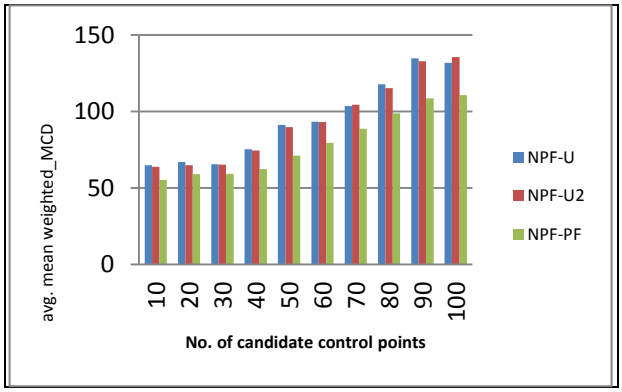

Fig. 7. Node Scale Sensitivity (avg. mean_weighted_MCD)

From Fig. 5 to 8, we can see that NPF-U2 outperforms NPF-U when weighted coverage is taken as objective. NPF-UF performs worst since weighted coverage is not its optimization objective. When populated edge coverage is taken as the objective, NPF-PF performs the best and NPF-U2 performs next. Because NPU-UF takes both the weight of edges and flows into considerations, solutions provided by NPU-UF are better than NPU-U2 and NPF-U.

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 performance in terms of weighted edge coverage and populated edge coverage is similar to the experiments of node scale sensitivity. Therefore, only mean_weight_MCD and mean_MCD are shown in Fig. 9.

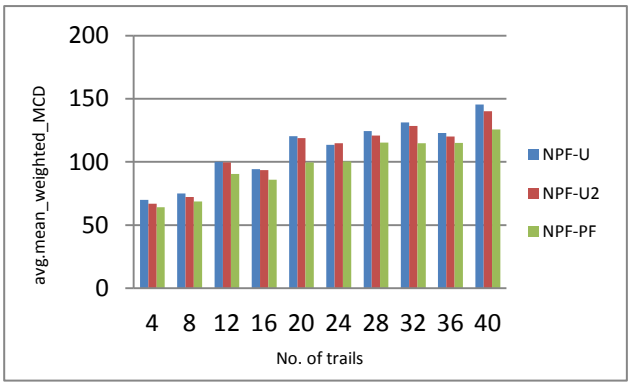

Fig. 8. Trail Scale Sensitivity (avg. mean_weighted_MCD)

The results (see Fig. 8) show that the performance of three algorithms in terms of avg. mean_weighted MCD. NPF-PF performs the best in terms of avg. mean MCD but not avg. mean_weighted_MCD. Although NPF-PF performs the best in terms of avg. mean_weighted_MCD, the margin is nominal. Both indices increase proportional to the number of trails and number of candidate control points.

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 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. This paper proposes a more comprehensive model, CPSP Utility-Flow, that taking not only weighted edge coverage but also the amount of agent flows into account. The simulative experiments show that CPSP Utility-Flow model is better than previous model. 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.

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] Yao-Nan Lien and Yi-Shiuan Lin, "Placement of Control Network for Mobile Agents over Opportunistic Networks", *Proc. of 8th International Workshop on Mobile Peer-to-Peer Computing*, Mar. 2012, Lugano, Switzerland.

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