An Energy-Efficient Algorithm for Object Tracking in Wireless Sensor Networks

Frank Yeong-Sung Lin and Cheng-Ta Lee
Department of Information Management
National Taiwan University
Taipei, Taiwan, R.O.C.
e-mail: {yslin,d90001}@im.ntu.edu.tw

Yen-Yi Hsu
Data Communications Business Group
Chunghwa Telecom Co., Ltd.
Taipei, Taiwan, R.O.C.
e-mail: yenyi@cht.com.tw

***Abstract*—In recent years, due to the rapid growth of sensor technology and wireless communication, Wireless Sensor Networks (WSNs) have been applied to various applications. Nevertheless, sensor nodes are highly energy-constrained, because of the limitation of hardware and the infeasibility of recharging the battery under a harsh environment. Therefore, energy consumption of sensor nodes has become a popular issue.**

The purpose of this paper is to achieve energy-efficient object tracking for an arbitrary topology in WSNs. Object tracking typically contains two basic operations: update and query. Most research only considers the update cost during the design phase, or adjusts the structure by taking the query cost into consideration in the second round. We aim to construct an object tracking tree with minimum communication cost, including both update and query costs. This problem is formulated as an integer programming problem. The Lagrangean relaxation method is adopted to find an optimal solution and develop a heuristic algorithm for constructing an object tracking tree with minimum communication cost.

Keywords-Wireless sensor networks; Object tracking; Lagrangean relaxation

I. INTRODUCTION

In recent years, because of the rapid growth of wireless communication and inexpensive sensors capable of sensing environmental information, wireless sensor networks have been used in a wide range of applications, including military intrusion detection, wildlife animal monitoring, and civil applications [13][14]. Many factors need to be taken into account when designing WSNs, such as coverage, end-to-end delay, and lifetime. An important challenge in the design of WSNs is that the battery level is fixed and it is not feasible to recharge the battery. Sensor nodes are highly energy-limited due to the limitation of hardware and environment. Thus, more and more research focuses on the problem of how to prolong the lifetime, and a great many approaches have been proposed, such as sleeping scheduling, data aggregation tree [3][8][11][15], adding some powerful nodes into the WSNs, etc.

Object tracking is one of the key application issues of WSNs, and this can be used to track enemy vehicles, detect illegal border crossings, etc. The sensor nodes are required to sense and track the movement of mobile objects, and then report to the special node, the *sink*. Object tracking wireless sensor networks typically involve two basic operations to maintain and obtain the location of the target object [6], the first of which is updating. When an object is moving from one sensor to another, its location must be updated in order to provide up-to-date information for the WSNs. The cost caused by object moving is referred to as the "update cost". The second is Query. In wireless networks, the *sink* acts as a gateway between the wireless sensor network and the external network. A query for the location of the object is usually sent from the external network to the *sink*, and the *sink* forwards the query message to other nodes in the WSNs to collect information. The total cost of transmitting the query message is defined as the "query cost". These two operations are interleaved during the entire process. In order to prolong the system's lifetime with limitations, it is necessary to adopt an adequate method to minimize the total cost.

There are many ways to maintain an object's information while it is moving around in the WSNs and querying the location of the target object. There are two ways of storing the data, the first of which involves storing it in different sensor nodes as a distributed database. A simple way to deliver the query message is to flood the entire network. However, a great number of query messages are wasted even though no update message is sent. The other method is to store all of the information in one specific node, i.e. the *sink* node. Once the sensor node senses that the object is within its sensing range, the sensor node sends the updated message back to the *sink*. Once the query arrives at the *sink*, no query message should be sent in the WSNs. Even if the query cost is zero, the update cost, caused by the object moving, is still considerable when the frequency of the object movement is high. As already mentioned, how to strike a balance between the update cost and the query cost is an important issue in object tracking wireless sensor network. This paper focuses on the problem of building an energy-efficient wireless sensor network for object tracking by using an object tracking tree with a given arbitrary topology.

In prior studies [2][4][5][6][7][10], the focus has been on developing strategies for reducing energy consumption when reporting operations. For example, Figure 1 illustrates a scenario of an event-driven report, such as wildlife animal monitoring and tracking in an outdoor situation. Sensor *u* will detect the object and deliver the object's location information to *sink* node when object enters the sensor filed, and sensor *v* will only forward the new location information to communication node *c* when object moves from sensor *u* to sensor *v*. This scenario can be performed through the entire sensor field. Finally, sensor *z* will forward the leaving information to *sink* node when object leaves sensor field

978-1-4244-5849-3/10/$26.00 ©2010 IEEE

from sensor *z*. In [10], the authors propose a scalable message-pruning hierarchy tree called a DAB for a sensor tracking system. In [6], the authors propose two message-pruning tree structures called DAT and Z-DAT for object tracking. In [2], the authors propose a new data aggregation structure, a message-pruning tree with shortcuts. The approaches described above assume that there is only one *sink*. In [7], the authors extend the problem from a single *sink* to multiple *sink*s and propose two algorithms called MT-HW and MT-EO.

This study is an extension of the work in [4][5][6][10], expanding the previous studies to energy-efficient object tracking in wireless sensor networks. It focuses on the problem of constructing an energy-efficient wireless sensor network for object tracking services using an object tracking tree rooted at the *sink*. Therefore, we are motivated to propose a heuristic algorithm to resolve the problem of a given arbitrary sensor network topology as a directed graph, and we particularly consider the bi-directed object moving frequency of in-sensor field and incoming-outgoing sensor field, bi-directed link transmission cost, and nodal processing cost. The total communication cost can be computed and minimized by the object tracking tree during the planning stage.

Calculating the communication cost is different from that of prior studies [6][10]. Firstly, we consider the bi-directed moving objects with given frequencies for each pair of sensor nodes, because the round-trip traffic cost of each pair of sensor nodes is different. Secondly, we consider the link transmission cost, since each link transmission cost is also different. Thirdly, we further take the query cost into consideration. We use an approximate approach to calculate the query rate by using the Markov chain [15].

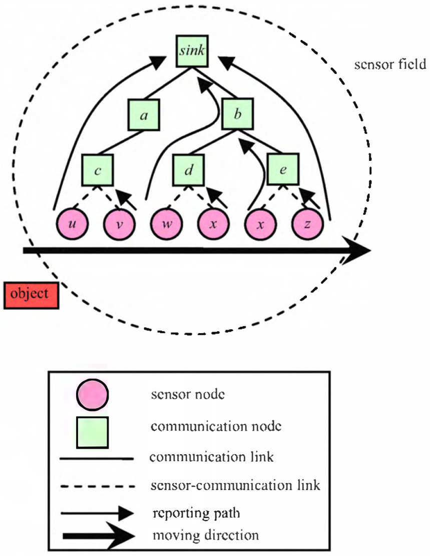

Figure 1. An example of object tracking.

Figure 2 illustrates an example of calculating the communication cost. The weight of each solid link represents the link transmission cost between a pair of adjacent communication nodes, or between a sensor node and communication node, and the weight of each dashed link represents the object moving frequency between a pair of adjacent sensors. For example, the communication cost is 5*8=40 when an object moves from sensor x to sensor y, and the communication cost is (3+2)*6=30 when an object moves from sensor y to sensor x. The probability under the sensor nodes is calculated by using a Markov chain, which denotes the query rates for each sensor node. The total query cost is (8+3)*60%*T+8*40%*T, where T is the number of query requests.

In this paper, we formulate the problem as a 0/1 integer-programming problem where the objective function is to minimize the total communication cost subject to routing, tree, and variable-transformation constraints. The object tracking tree in a weighted graph spans a given sensor and communication nodes, and the tree is used to minimize total communication cost. Therefore, constructing the object tracking tree is NP-complete problem [2]. A Lagrangean relaxation-based (LR-based) heuristic algorithm is used to resolve the sub-problem and obtain a primal feasible solution.

The problem is formulated as a linear optimization-based problem with three different decision variables: paths, tree links, and tracking links, and the Lagrangean relaxation method, which has been successfully adopted to resolve many famous NP-complete problems [1][9][12], is adopted to fulfill the timing and the quality requirements of the optimal decisions. In subsequent computational experiments, our proposed object tracking algorithm is expected to be efficient and effective in dealing with the complex optimization problem.

Having reviewed the papers, this study is found to differ from the prior works in three aspects [6][10], the first of which is that it considers the bi-directed moving objects with given frequencies for each pair of sensor nodes and link transmission cost. Secondly, it presents an LR mathematical model to describe the optimization problem and proposes an LR-based heuristic algorithm to resolve the problem. Thirdly, it considers the query cost as well as updating it at the same time.

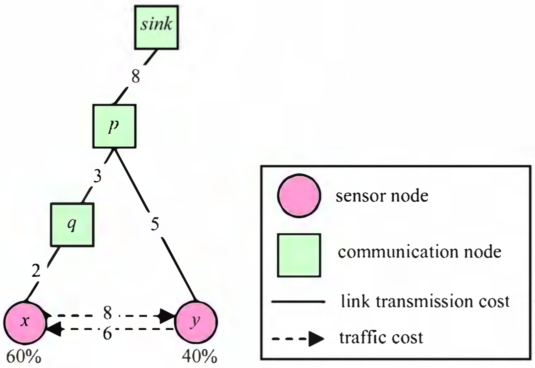

Figure 2. An example of calculating communication cost.

The rest of this paper is organized as follows. The problem and mathematic model are described in sections 2 and 3, respectively, a solution approach is presented in section 4, and the computational results are discussed in section 5. Finally, the conclusion of the paper is presented in section 6.

II. PROBLEM DESCRIPTION

Our approach focuses on the construction of an object tracking tree, which is used to record the object's information and keep it up to date. The sensor field consists of sensor nodes and communication nodes. Sensor nodes are appointed to sense and track the mobile object and send the information back to the *sink*. Communication nodes are required to relay the updated message, and store and maintain a detected list. Figure 1 shows an example of object tracking.

The object tracking problem is modeled as a directed graph, $G(V, L)$ where V is a set of communication nodes and sensor nodes randomly deployed in a 2D sensor field. L is a set of links between the adjacent sensor nodes or connected to one communication node and one sensor node. Each link weight represents the distance between the sensor nodes. For example, the sensor sub-graph in Figure 3 illustrates a 2D sensor field with each edge connecting a pair of adjacent sensors. Each link weight is the object moving frequency of each pair of sensor nodes.

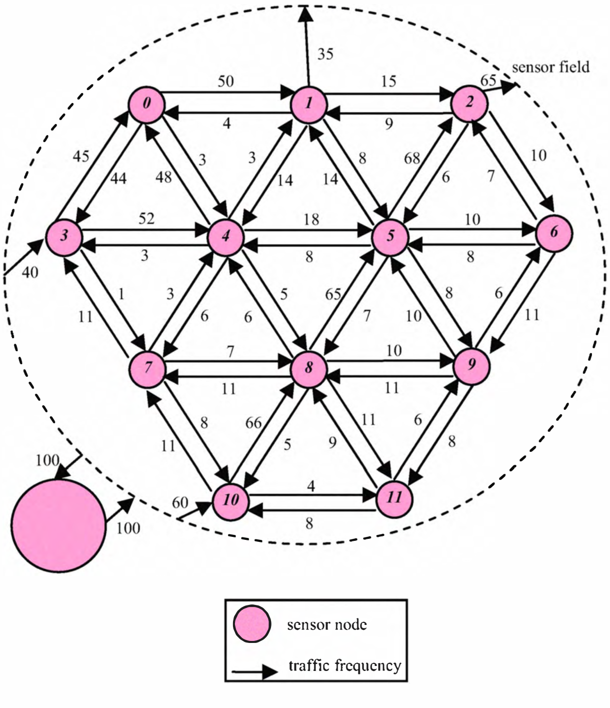

Figure 3. An example of 2D sensor sub-graph.

Figure 4 illustrates a 2D sensor field's routing sub-graph with each edge connecting a pair of adjacent communication nodes or sensor-communication nodes. Each link weight represents a link transmission cost.

We model the object movements as a stochastic process, the following property will be observed in the steady-state.

$$S^{(0)}P = S^{(1)} \quad \rightarrow \quad S^{(1)}P^{n-1} = S^{(0)}P^{n} = S^{(n)}$$

$$\rightarrow \quad S^{(n)}P = S^{(n)} \quad \rightarrow \quad \pi P = \pi \qquad (2.1)$$

$S^{(i)}$ denotes the network state at time i. For example, there are 5 sensors (sensor 1, 2, 3, 4, 5), only sensor 1 covers the target object at time 1, hence, $S^{1} = [1, 0, 0, 0, 0]$.

P is a $n \times n$ array, each elements p_{ij} in the array denotes a probability of objects moving from sensor i to sensor j.

We use π to indicate the network state at steady-state as (2.2), and the summation of every element in π should be equal to 1 as (2.3). Combine these two conditions as following:

$$\begin{cases} \pi P = \pi = \pi I & (2.2) \\ \pi_1 + \pi_2 + \ldots + \pi_n = 1 & (2.3) \end{cases}$$

We use $\pi A = e$ to represent (2.4), and then we try to find the result of π.

$$[\pi_1 \quad \pi_2 \quad \ldots \quad \pi_n] \times \begin{bmatrix} p_{11}-1 & p_{12} & \cdots & p_{1n} & 1 \\ p_{21} & \ddots & & \vdots & 1 \\ \vdots & & \ddots & \vdots & 1 \\ p_{m1} & \cdots & \cdots & p_{mn}-1 & 1 \end{bmatrix}$$

$$= [0 \quad 0 \quad \cdots \quad 1] \qquad (2.4)$$

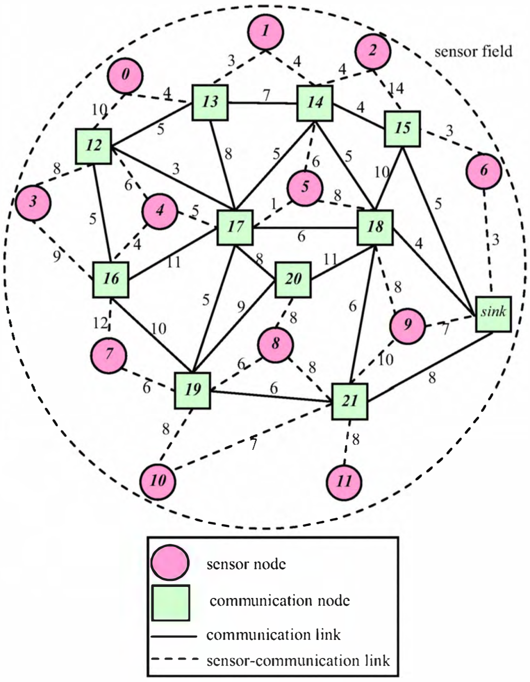

Figure 4. An example of 2D routing sub-graph.

Step 1: $\pi AA' = eA'$

Step 2: $\pi AA'(AA')^{-1} = eA'(AA')^{-1}$

$AA'(AA')^{-1}$ is a identity matrix, hence we get $\pi = eA'(AA')^{-1}$, π_x as the probability of the object which is in the sensing range of sensor node *x*. We further let π_x multiplied by *T* as the query rate of node *x* at a given period of time, *T* is the total number of queries in a unit time.

Since upward links and downward links may have a different transmission cost. This approach can keep a certain ratio between the upward link cost and the downward link cost. Therefore, the transmission cost can be defined as the power consumption of transmitting the data, which is measured as $r^{\alpha} + c$, where α is a signal attenuation constant (usually between 2 to 6), and *c* is a positive constant which represents signal processing and *r* is the Euclidean distance between any nodes.

This study considers a given arbitrary sensor network topology as a directed graph, a bi-directed object with given moving frequency of the in-sensor field and the incoming-outgoing sensor field, a bi-directed link transmission cost, and the nodal processing cost. A tree-based architecture is employed, with sensor nodes as leaf nodes, sending data to its ancestor which is an adjacent communication node.

A good tracking method is characterized by a low total communication cost [10] and given an arbitrary graph the total communication cost can be computed. Thus, the total communication cost for graph *G* is calculated as follows:

Total communication cost (*G*) = total update cost + total query cost, as (2.5).

$$\begin{aligned}&\sum_{x\in S}\sum_{y\in S}\sum_{(i,j)\in L} t^{xy}_{(i,j)} r_{xy}(ua_{(i,j)} + d^{w}_{j})\\ &+\sum_{s\in S}\sum_{(i,j)\in L} z^{s}_{(i,j)}(r_{os} + r_{so})(ua_{(i,j)} + d^{w}_{j})\\ &+\sum_{s\in S}\sum_{(i,j)\in L} [w_{(i,j)} z^{s}_{(i,j)} Q_s T(va_{(j,i)} + d^{r}_{i})\\ &+w_{(i,j)} z^{s}_{(i,j)} Q_s T(ua_{(i,j)})]\end{aligned} \tag{2.5}$$

Where *S* is the set of all sensor nodes and *L* is the set of all links. The decision variable $t^{xy}_{(i,j)} = 1$ if $z^{x}_{(i,j)} = 0 \cap z^{y}_{(i,j)} = 1$ (reporting object's location uses the link (*i*, *j*) when object moves from sensor *x* to sensor *y*) and 0 otherwise. The decision variable $z^{s}_{(i,j)} = 1$ if the sensor node s uses the link (*i*, *j*) to reach the *sink* node and 0 otherwise. $a_{(i,j)}$ is the transmission cost associated with link (*i*, *j*). r_{xy} is the frequency of object movement from sensor *x* to sensor *y*. r_{os} is traffic frequency while entering the sensor field, and r_{so} is traffic frequency when leaving the sensor field. *u* is the coefficient of the upward links and *v* is the coefficient of the downward links. Q_s represents the probability of the object which is within the sensing range of sensor *s*. *T* is the total number of query requests in a given time. $Q_s \times T$ represents the query rate of sensor node s. $w_{(i,j)} = 1$ if the link connects with the two communication nodes, and 0 otherwise. d^{w}_{i} and d^{r}_{i} are the nodal processing costs of wiring and reading the detected list for node *i*.

III. PROBLEM FORMULATION

The notations used to model the problem are listed in the following table.

TABLE I. GIVEN PARAMETERS

Notation	Description
S	Set of all sensor nodes
C	Set of all communication nodes, including *sink* node
R	Set of the frequency (r_{xy}) of object movement from *x* to *y*, $\forall x, y \in S \cup \{o\}$, $x \neq y$
L	Set of all links, $(i, j) \in L$, $i \neq j$
A	Set of transmission costs $a_{(i,j)}$, associated with link (*i*, *j*)
P	Set of all candidate paths *P* between any pair (*s*, *sink*) $\forall s \in S$
Q_s	Probability of nodes *s* that object has in its sensing range, $\forall s \in S$
T	Total number of queries per unit time
o	Artificial node outside sensor field
u	Coefficient of upward links
v	Coefficient of downward links
d^{w}_{c}	Nodal processing cost of writing operation of the communication node *c*
d^{r}_{c}	Nodal processing cost of reading operation of the communication node c

TABLE II. INDICATED PARAMETERS

Notation	Description
$\delta_{p(i,j)}$	Indicator function is 1 if link (*i*, *j*) is on path *p* and 0 otherwise
$w_{(i,j)}$	1 if $i, j \in C$ 0 otherwise

TABLE III. DECISION VARIABLES

Notation	Description
x_{sp}	1 if the sensor node *s* uses the path *p* to reach the *sink* node and 0 otherwise, $\forall s \in S, p \in P$
$z^{s}_{(i,j)}$	1 if the sensor node *s* uses the link (*i*, *j*) to reach the *sink* node and 0 otherwise
$t^{xy}_{(i,j)}$	1 if $z^{x}_{(i,j)} = 0 \cap z^{y}_{(i,j)} = 1$ (reporting object's location uses the link (*i* ,*j*) when object moves from sensor *x* to sensor *y*) and 0 otherwise, $x \neq y$

Problem (IP):
Objective function:

$$
\begin{aligned}
Z_{IP} = min \sum_{x\in S}\sum_{y\in S}\sum_{(i,j)\in L} & t^{xy}_{(i,j)} r_{xy}(ua_{(i,j)} + d^{w}_{j}) \\
+ \sum_{s\in S}\sum_{(i,j)\in L} & z^{s}_{(i,j)}(r_{os} + r_{so})(ua_{(i,j)} + d^{w}_{j}) \\
+ \sum_{s\in S}\sum_{(i,j)\in L} & [w_{(i,j)} z^{s}_{(i,j)} Q_s T(va_{(j,i)} + d^{r}_{i}) \\
& + w_{(i,j)} z^{s}_{(i,j)} Q_s T(ua_{(i,j)})]
\end{aligned}
\tag{IP}
$$

Subject to:

$$\sum_{p\in P_s} x_{sp} = 1 \qquad \forall s \in S \tag{3.1}$$

$$\sum_{j\in C} z^{s}_{(i,j)} \le 1 \qquad \forall s \in S, i \in S \bigcup C \tag{3.2}$$

$$\sum_{p\in P_s} x_p \delta_{p(i,j)} \le z^{s}_{(i,j)} \qquad \forall s \in S, (i,j) \in L \tag{3.3}$$

$$2t^{xy}_{(i,j)} \le z^{y}_{(i,j)} - z^{x}_{(i,j)} + 1 \qquad \forall x, y \in S, (i,j) \in L \tag{3.4}$$

$$z^{y}_{(i,j)} - z^{x}_{(i,j)} + 1 \le t^{xy}_{(i,j)} + 1 \qquad \forall x, y \in S, (i,j) \in L \tag{3.5}$$

$$\sum_{j\in C} t^{xy}_{(y,j)} \ge 1 \qquad \forall x, y \in S \tag{3.6}$$

$$\sum_{j\in C} z^{s}_{(s,j)} = 1 \qquad \forall s \in S \tag{3.7}$$

$$x_{sp} = 0 \; or \; 1 \qquad \forall s \in S, p \in P_s \tag{3.8}$$

$$z^{s}_{(i,j)} = 0 \text{ or } 1 \qquad \forall s \in S, (i,j) \in L \tag{3.9}$$

$$t^{xy}_{(i,j)} = 0 \text{ or } 1 \qquad \forall x, y \in S, (i,j) \in L. \tag{3.10}$$

The objective is to minimize the total cost of constructing an object tracking tree, and the total cost is defined as being a combination of update and query costs.

Constraint (3.1):	Routing constraint: For each sensor node *s*, only one path exactly exists between the *s* and the *sink*.
Constraint (3.2):	To avoid a cycle, we enforce that any nodes' outgoing link to the communication node should be equal to 1 on the object tracking tree, except for the *sink* node.
Constraint (3.3):	If path x_p has been chosen, and link (*i* ,*j*) is on the path, link (i, j) should be chosen, i.e. decision variable $z^{s}_{(i,j)}$ should be enforced to equal 1
Constraints (3.4- 3.5):	These two constraints are variable-transformation constraints. When the object moves from sensor node *x* to sensor node *y* using the link (*i*, *j*) to report the object's location to the *sink*, i.e. $z^{x}_{(i,j)} = 0 \bigcap z^{y}_{(i,j)} = 1$, $t^{xy}_{(i,j)}$ must be enforced to equal 1 and 0 otherwise.
Constraint (3.6):	This is a redundant constraint, and it is used to guarantee that, when an object is moving within the WSNs, at least one sensor node *s* can detect the object, and that one link exists for the sensor node s to transmit a message to the *sink*.
Constraint (3.7):	This is a redundant constraint, and it is used to guarantee that all of the sensor nodes choose at least one, and only one, link to transmit a message to the *sink*.
Constraints (3.8-3.10):	The integer constraints for decision variables x_p, $z^{s}_{(i,j)}$, and $t^{xy}_{(i,j)}$ must equal 0 or 1.

IV. SOLUTION APPROACH

A. Lagrangean Relaxation

By adopting the Lagrangean relaxation method, the primal problem can be transformed into the following Lagrangean relaxation problem by relaxing constraints (3.3), (3.4), and (3.5). The Lagrangean relaxation problem is presented for a vector of non-negative multipliers, as shown below.

Problem (LR):
Objective function:

$$
\begin{aligned}
& Z_{LR}(u^{1}_{s(i,j)}, u^{2}_{xy(i,j)}, u^{3}_{xy(i,j)}) = \\
& \min \sum_{x\in S}\sum_{y\in S}\sum_{(i,j)\in L} t^{xy}_{(i,j)} r_{xy}(ua_{(i,j)} + d^{w}_{j}) \\
& + \sum_{s\in S}\sum_{(i,j)\in L} z^{s}_{(i,j)}(r_{os} + r_{so})(ua_{(i,j)} + d^{w}_{j}) \\
& + \sum_{s\in S}\sum_{(i,j)\in L} [w_{(i,j)} z^{s}_{(i,j)} Q_s T(va_{(j,i)} + d^{r}_{i}) \\
& + w_{(i,j)} z^{s}_{(i,j)} Q_s T(ua_{(i,j)})] \\
& + \sum_{s\in S}\sum_{(i,j)\in L} u^{1}_{s(i,j)}(\sum_{p\in P_s} x_p \delta_{p(i,j)} - z^{s}_{(i,j)}) \\
& + \sum_{x\in S}\sum_{y\in S}\sum_{(i,j)\in L} u^{2}_{xy(i,j)}(2t^{xy}_{(i,j)} - z^{y}_{(i,j)} + z^{x}_{(i,j)} - 1) \\
& + \sum_{x\in S}\sum_{y\in S}\sum_{(i,j)\in L} u^{3}_{xy(i,j)}(z^{y}_{(i,j)} - z^{x}_{(i,j)} - t^{xy}_{(i,j)})
\end{aligned}
\tag{LR}
$$

Subject to: (3.1), (3.2), (3.6), (3.7), (3.8), (3.9), and (3.10)

This LR problem can be further decomposed into following four independent sub-problems, according to different decision variables, and easily solvable optimization sub-problem.

$$Z_{LR} = Z_{sub1} + Z_{sub2} + Z_{sub3} + Z_{sub4} \tag{3.11}$$

B. Getting Primal Feasible Solutions

An LR-based primal heuristic algorithm is listed in Figure 5 and the complete object tracking tree algorithm is listed in Figure 6.

```
Algorithm Primal_Heuristic
Step 1 Use the shortest path tree algorithm (SPT) to
       determine the initial primal value
Step 2 Adjust arc weight $c_{s(i,j)} = \sum_{s \in S} u^{1}_{s(i,j)}$ for each
       $(i,j) \in L$ and then run the Dijkstra algorithm to
       obtain the solution set of $\{ x_{sp} \}$
Step 3 Once $\{ x_{sp} \}$ is determined, $t^{xy}_{(i,j)}$ and $z^{s}_{(i,j)}$ are also
       determined
Step 4 We can have an object tracking tree now, and then
       iteratively execute Steps 2~3 with LR multipliers
       which can be updated from a dual mode problem
```

Figure 5. LR-based primal heuristic algorithm.

```
Algorithm Object_Tracking_Tree
begin
    Initialize the Lagrangean multiplier vector $(u_1,u_2,u_3)$
    to be zero vectors
    UB:=total communication cost of shortest path tree
     LB:=very small number
     improve_counter:=0; step_size_coefficient:=2
     for iteration:=1 to Max_Iteration_Number do
     begin
          run sub-problem(SUB1)
          run sub-problem(SUB2)
          run sub-problem(SUB3)
          run sub-problem(SUB4)
          calculate $Z_D$
       if $Z_D > LB$ then LB:= $Z_D$ and
            improve_counter:=0
       else improve_counter:=
            improve_counter+1;
          if improve_counter= improve_Threshold
          then improve_counter:=0; $\lambda := \lambda / 2$
          run Primal_Heuristic Algorithm
          if ub<UB then UB:=ub
          /* ub is the newly computed upper bound */
          run update-step-size
          run update-Lagrangean-multiplier
     end
end
```

Figure 6. Object tracking tree algorithm.

V. COMPUTATIONAL EXPERIMENTS

An experiment was conducted to evaluate the performance of the proposed algorithm, which was assessed in terms of the total communication cost.

A. Scenario

The proposed algorithm for constructing object tracking trees was coded in programming language C and executed on Windows XP and Visual C++ 6.0. The program was run on a notebook with Intel Core2 Duo 2.20G CPU and 2GB RAM. The algorithm was tested on a 2D sensor field. Sensor nodes and communication nodes were distributed in a sensor field.

B. Experimental Results

In order to evaluate our proposed heuristic algorithm, it was compared with two heuristic algorithms. Dijkstra's algorithm (shortest path tree, SPT) was implemented as Simple Algorithm 1 (SA1), and a spanning tree-like (ST-like) algorithm as Simple Algorithm 2 (SA2). Since the regular spanning tree algorithm, Kruskal, may regard the sensor node as an intermediate node, this would violate our assumption. Therefore, all of the communication nodes, including the *sink* node, are spanned by using Kruskal's Algorithm prior to finding the shortest path to the spanning tree of each sensor node.

TABLE IV. and TABLE V. respectively show the total transmission cost calculated by different algorithms under a different number of nodes. It can be seen that the heuristic proposed in section 5 outperforms the other two simple algorithms. We denote the dual solution as "Z_{du}", and Lagrangean relaxation-based heuristic as "Z_{IP}". "*Gap*" is used to evaluate our solution quality. $Gap = |(Z_{IP} - Z_{du}) / Z_{du}| * 100\%$.

Figure 7 shows an example of a trend line for obtaining the primal problem solution values (*UB*) and dual mode problem values (*LB*). The *UB* curves tend to decrease to acquire the minimum feasible solution. In contrast, the *LB* curves tend to increase and converge rapidly to reach the optimal solution. The LR-based method ensures optimization results between *UB* and *LB* so that the duality gaps can be kept as small as possible in order to improve the solution quality and achieve near optimization.

Figure 8 show object tracking trees found by using the proposed LR-based algorithm under *T*=450. Since most of the updates usually happen on the links, which are further from the *sink*, most of the queries contrarily happen on the links, which are closer to the *sink*. Yet, the tree structure formed by the latter links becomes much like the tree structure formed by shortest path tree, when *T* becomes larger.

TABLE IV. EVALUATION OF GAP BY GIVEN DIFFERENT NUMBER OF NODES AND DIFFERENT QUERY RATES

Number of nodes		Z_{du}	Z_{IP}	*Gap* *(%)*
23	*T*= 0	20255	23338	15.2
	T= 450	33555	35802	6.7
	T= 960	46481	48182	3.7
36	*T*= 0	7180	9816	36.7
	T= 746	23487	27971	19.0
	T=1100	30397	35819	17.8
57	*T*= 0	51057	68619	34.4
	T=1400	106995	113410	6.0
	T=2856	152937	158661	3.7
87	*T*= 0	58387	89410	53.1
	T=1000	96374	126349	31.1
	T=3726	178029	228062	28.1

TABLE V. EVALUATION OF IMPROVEMENT RATIO (%) BY GIVEN DIFFERENT NUMBER OF NODES AND DIFFERENT QUERY RATE

Number of nodes		SA1	Improvement Ratio to SA1(%)	SA2	Improvement Ratio to SA2(%)
23	$T=0$	25546	9.5	25483	9.6
	$T=450$	37915	5.9	45632	27.5
	$T=960$	51933	7.8	68468	42.1
36	$T=0$	12684	29.2	12436	26.7
	$T=746$	28873	3.2	54048	93.2
	$T=1100$	36511	1.9	78561	119.3
57	$T=0$	75030	9.3	84463	23.1
	$T=1400$	119466	5.3	279920	146.8
	$T=2856$	165680	4.4	483195	204.5
87	$T=0$	94855	6.1	106002	18.6
	$T=1000$	131435	4.1	175987	39.3
	$T=3726$	231152	1.4	366766	60.8

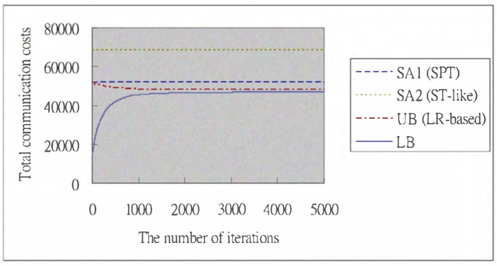

Figure 7. The execution result of LR based algorithm (Number of nodes = 23 and total query number=960).

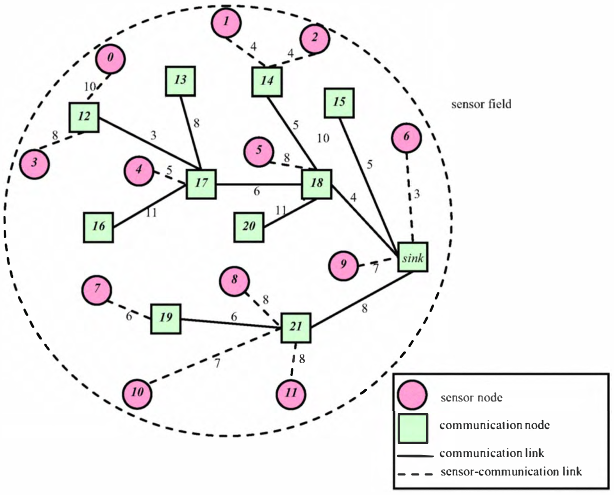

Figure 8. An example of a LR-based object tracking tree under T=450.

VI. CONCLUSION

This paper firstly proposes a mathematical formulation to model an object tracking tree construction problem as an 0/1 integer programming problem, and apply Lagrangean relaxation and a sub-gradient method to resolve it. Then, a heuristic approach is designed to obtain a feasible solution. Finally, several experiments are conducted on different cases. According to the results of these experiments, it can be claimed that our Lagrangean relaxation-based algorithm not only outperforms other heuristics, such as shortest path tree and spanning tree-like, but also that the gap is small. The results show that the proposed LR-based algorithm can achieve energy-efficient object tracking and, furthermore, that it is very efficient and scalable in terms of the solution time.

It is planned to further take the load balancing and residual energy capacity into consideration to prevent the "hot spot" letting the tree fail to work. In addition, it is intended to extend the model to multiple *sink*s object tracking tree in near future, since the multiple *sink*s can provide load balancing and failure tolerance.

REFERENCES

[1] A.M. Geoffrion, "Lagrangean Relaxation and its Use in Integer Programming," *Mathematical Programming Study*, vol. 2, pp. 82-114, 1974.

[2] B.H. Liu, W.C Ke, C.H. Tsai, and M.J. Tsai, "Constructing a Message-Pruning Tree with Minimum Cost for Tracking Moving Objects in Wireless Sensor Networks Is NP-Complete and an Enhanced Data Aggregation Structure," *IEEE Transactions on Computers*,pp. 849-863, June 2008.

[3] Y.F. Wen, F.Y.S. Lin and W.C. Kuo, "A Tree-based Energy-efficient Algorithm for Data-Centric Wireless Sensor Networks," *Proc. IEEE AINA*, 2007.

[4] C.T. Lee, F.Y.S. Lin, and Y.F. Wen, "An Efficient Object Tracking Algorithm in Wireless Sensor Networks," *Proc. JCIS'06*, 2006.

[5] C.T. Lee and F.Y.S. Lin, "An Energy-Efficient Lagrangean Relaxation-based Object Tracking Algorithm in Wireless Sensor Networks," *20th International Conference on Information Management (ICIM)*, 2009.

[6] C.Y. Lin, W.C. Peng, and Y.C. Tseng, "Efficient In-Network Moving Object Tracking in Wireless Sensor Networks," *IEEE Transactions on Mobile Computing*, pp. 1044-1056, August 2006.

[7] C.Y. Lin, Y.C. Tseng, T.H. Lai, and W.C. Peng, "Message-efficient In-network Location Management in a Multi-*sink* Wireless Sensor Network," *International Journal of Sensor Networks*, pp. 496-505, 2008.

[8] S. Bhatti and X. Jie, "Survey of Target Tracking Protocols Using Wireless Sensor Network," *International Conference on in Wireless and Mobile Communications, ICWMC* '09. Fifth, pp. 110-115, 2009.

[9] M.L. Fisher, "An Application Oriented Guide to Lagrangean Relaxation," *Interfaces*, vol. 15, no. 2, pp. 10-21, April 1985.

[10] H.T. Kung and D. Vlah, "Efficient Location Tracking Using Sensor Networks," *Proceedings of 2003 IEEE Wireless Communications and Networking Conference (WCNC)*, 2003.

[11] H.H. Yen, F.Y.S. Lin, and S.P. Lin, "An Energy-Efficient Data-Central Routing Algorithm in Wireless Sensor Networks," *Proc. IEEE ICC*, 2005.

[12] M.L. Fisher, "The Lagrangian relaxation method for solving integer programming problems," *Management Science*, vol. 27, no. 1, pp. 1-18, 1981.

[13] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless Sensor Networks: a Survey," *Elsevier Journal of Computer Networks*, vol. 38, pp. 393-422, March 2002.

[14] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks," *IEEE Communications Magazine*, pp. 102-114, August 2002.

[15] L.H. Yen and C.C. Yang, "Mobility Profiling Using Markov Chains for Tree-Based Object Tracking in Wireless Sensor Networks," *Proc. IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing vol 2*, pp. 220-225, June 2006.