Optimal Data Cache Allocation for Mobile Devices in Sensor-Cloud

Yaling Tao[1,a)] Yongbing Zhang[1,b)] Yusheng Ji[2,c)]

Abstract: Recent years, many efforts have been made to overcome the limitation of storage and computing capability of wireless sensor networks (WSNs). With the computing paradigm of cloud computing characterized by massive storage and computing capability, larger market and more opportunities arise for WSNs. The *sensor-cloud* combing sensor networks into cloud can provide excellent data scalability, rapid visualization, and user programmable analysis. Our primary work in this paper is to allocate sensor data caches on the cloud to minimize the data access and maintenance costs. We present a data cache model and formulate three types of data allocation problems, called single-type (SDAP), uncapacitated multi-type (UCMDAP), and capacitated multi-type (CMDAP), respectively. We propose a Lagrangian relaxation algorithm to solve the SDAP, and analyze its usability for UCMDAP and CMDAP. We also examine the performance of our algorithm by numeral experiments.

1. Introduction

In the past few years, wireless sensor networks (WSNs) have been attracting increasing attention because of their potential of enabling of novel and attractive solutions in variety areas such as military, industrial automation, environment monitoring, healthcare, smart home, etc. Many efforts, such as topology control, energy efficiency technique, data aggregation, and sensor scheduling, have been made to overcome the limitation of storage and computing capabilities of WSNs [1], [2], [3], [4]. However, most studies focus only on the inside of WSNs. The gateway, called *sink*, connecting to the outside of WSN is usually assumed to have infinite storage and computing capabilities. Some work such as data analysis, data accessibility, and data utilization following the data collection at sink are left aside or pushed into other research areas. The uses of data from one specific WSN is limited to a small group of users due to the lack of share-ability, efficient maintenance, and elasticity.

At the same time, *cloud computing*, acting as one of the most popular concepts in recent years, has been attracting much unprecedented attentions from institutions and individuals. With the computing paradigm of cloud computing, larger market and more opportunities arise for WSNs. Data collected from WSNs can be efficiently maintained on cloud and easily shared by different groups of users. Cloud computing possessing massive computing and storage capability acts as a remedy for the limitation of WSNs. On the other hand, WSNs act as data sources providing various kinds of data for cloud in a continuous, pervasive, and real-time manner.

The concept of *Sensor-Cloud* that combines sensor networks with cloud computing can be found in recent works like [6], [7], [8], [9], [10], [11]. An

[1] University of Tsukuba, Tsukuba, Ibaraki 305-8573, Japan
[2] National Institute of Informatics, Japan
[a)] to.are@sk.tsukuba.ac.jp
[b)] ybzhang@sk.tsukuba.ac.jp
[c)] kei@nii.ac.jp

example of sensor-cloud is shown in Fig.1, different kinds of WSNs are combined into cloud, the connection points, such as S_1, S_2, and S_3, play the roles of sinks in WSNs collecting raw data and the roles of data centers in cloud processing and assigning data for users. All of data generated by WSNs can be accessed via cloud.

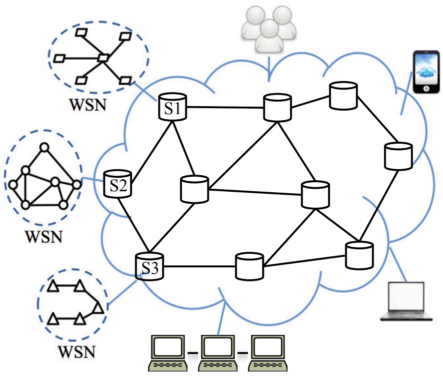

Fig. 1 data allocation on sensor-cloud

A common technique used to improve data access performance is caching, i.e., to cache data at some appropriate locations, so that data access requests from users can be responded efficiently. This benefit becomes especially important for the mobile devices with limited resources. Our work focuses on deciding cache nodes on sensor-cloud with the objective of minimizing the total system cost. This is a challenging problem because finding a strategy that minimizes costs (e.g., data updating cost, data assigning cost, data accessing cost, etc) is combinatorial problem, which includes decisions such as how many duplications to make, allocation of data items to cache nodes, storage capacity, and load balancing. Furthermore, there is significant uncertainty involved in a wide variety of data items and users.

In this paper, we present a data cache model and formulate three types of data allocation problems. The first one is single-type data allocation problem (SDAP), in which only one WSN combines to cloud. The next one is uncapacitated multi- type data allocation problem (UCMDAP), in which multiple WSNs combine to cloud, and nodes on the cloud are assumed to have no capacity limitations. The last one is capacitated multi-type data allocation problem (CMDAP), in which multiple WSNs combine to cloud, and nodes on the cloud have capacity limitation. Then, we propose a Lagrangian relaxation algorithm to solve the SDAP, and analyze its usability for UCMDAP and CMDAP. We also examine the performance of our algorithm by computational experiments.

The remainder of this paper is organized as follows. In the next section, we list some related works and compare them with our work. In section 3, we describe our system model and give some prerequisites. The formal formulation, solution and computational experiment for SDAP are presented in section 4. And the UCMDAP and CMDAP are discussed in section 5. Finally, we conclude this paper and give our future work in section 6.

2. Related work

A series of studies, [6], [7], [8], [9], [10], [11], have recently been done on the sensor-cloud. Most of these studies primally focuses on the system architecture of sensor-cloud. Hassan et al. Alamri et al. [11] give a survey on sensor-cloud in aspects of architecture, applications, and approaches. Differing from above researches, our work focus on data allocation problems with the goal of minimizing data access and maintenance costs. Contrast to the conceptual descriptions in previous work, mathematical formulation and numerical algorithm are explicitly presented in our work.

Data cache strategy are studied in some work such as [12], [13], [14]. The number and location of replicas of distinct data items in cloud have a strong impact on systems performance. Tan et al. [12] address the problem of content placement in peer-to-peer systems, with the objective of maximizing the utilization of peers uplink bandwidth resources. Data items are divided into three different classes, named *Hot*, *Warm*, and *Cold*, according to their popularity ranking. Hot items are cached at all nodes, Warm items are cached at a fraction of nodes, and Cold items are node cached at all. Similarly, Bjorkqvist et al. [13] also partition items into three classes: *Gold*, *Silver*, and *Bronze*. With the objective of minimizing retrieval latency, gold items are always stored at the edge node, whereas bronze items as never stored, silver items are managed locally either by a collaborative LRU scheme or by a random discarding scheme.

In above work, data popularity plays a key role in determining the cache allocation, the bandwidth utilization or latency are used as the performance metrics. In our work, all costs of data process are converted into monetary cost and the objective is to minimize the total cost.

We consider the *data assigning cost* to transfer sensed data from each sink of WSN to cache nodes in cloud, *data placing cost* to place data at cache nodes, and *data accessing cost* to respond requests. Generally, as the number of caches of a data item increases, the corresponding assigning cost and placing cost will increase, and the accessing cost will decrease. The data allocation problem has some similarities on a facility location problem (FLP) [15], [16], [17]. In the conventional facility location problem, a set of facilities with facility-opening costs and a set of clients with demands are given, the objective is to open facilities and assign clients to open facilities so as to minimize facility-opening costs and client-assignment costs. In our problem, data caches represent the facilities, the data placing cost and data accessing cost model the facility-opening cost and client-assignment cost, respectively. Despite the similarity with FLP, factors like continuous data assignment from sinks and cache-capacity constraints make it difficult to apply the standard method for FLP.

3. Data Allocation Model

In our data allocation model, data items collected from different kinds of WSNs are stored and maintained on cloud, various kinds of users can access sensor data items via cloud conveniently. Nodes connecting WSNs and cloud, named *source nodes* are responsible of collecting raw data from WSNs and assigning data to other nodes on cloud. Some of other nodes, named *cache nodes*, are selected to store duplications of data to control data transition cost. And those nodes directly response to users are called *demand nodes*. Source nodes collect data from WSNs and send the latest data to cache nodes periodically, demand nodes can access data from cache node or source nodes directly. In our data transfer model, three types of cost are considered: the *assigning cost* generated as source nodes assign the latest data to cache nodes periodically, the *placing cost* generated as cache nodes erase old data and write new data, and the *accessing cost* generated as demand nodes access data from cache nodes (or source node). Our work focuses on deciding cache nodes for data items on sensor-cloud with the objective of minimizing the total system cost.

Formally, let $\mathcal{N} = \{i \mid i = 1, 2, ..., N\}$, $\mathcal{S} = \{S_k \mid k = 1, 2, ..., M\}$, $\mathcal{M} = \{k \mid k = 1, 2, ..., M\}$ represent respectively the sets of nodes excluding source nodes on sensor cloud, source nodes connecting M different WSNs and cloud, and data items generated from M different WSNs. And $\mathcal{N}' = \mathcal{S} \bigcup \mathcal{N}$ is the set of all nodes on cloud. The data transfer model in this paper is illustrated in Fig.2. Source node S_k collects data k from the kth WSN and sends the latest data to its cache node(s) j, demand node(s) i access data from cache node j or source node S_k. As data-collection in WSN is usually executed within a certain period of time, denoted by T_0, so the data transmission between source node and cache node is also considered to be periodically executed. On the other hand, data transmission between cache node j (or source node i) and demand node i is often assumed to be a Poisson process. The number of requests for data k from demand node i during the period T_0 can be known, denoted by h_i^k. The cost to place a data duplication on node j is f_j, and the costs to transfer unit data per distance from source node to cache node and from the cache node (or source node) to the demand node are α and β, respectively. As the transmission between source node and cache nodes can be scheduled to execute during system idle time, we can set $\alpha \leq \beta$. Data is routed along the shortest path, d_{js}, d_{ij}, and d_{is} are respectively the shortest distance between source node s and cache node j, the shortest distance between cache node j and demand node i, and the shortest distance between source node s_k and demand node i. We also define two decision variables $x_j^k (j \in \mathcal{N})$ and $y_{ij}^k (i \in \mathcal{N}, j \in \mathcal{N}')$. The allocation decision variable x_j^k is 1 if node j is selected as a cache node for data item k, and 0 otherwise. The access decision variable y_{ij}^k is 1 if demand node i accesses data item k from cache node j, 0 otherwise. Primal symbols used to define problems are listed in table 1.

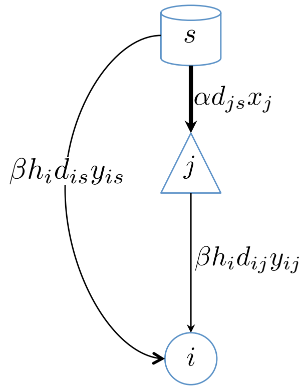

Fig. 2 data transfer model

Table 1 Symbols used in problem formulation

$\mathcal{N}$	the set of nodes on cloud (excluding sink nodes) $\mathcal{N} = \{i \mid i = 1, 2, ..., N\}$.
$\mathcal{M}$	the set of data items generated by M WSNs $\mathcal{M} = \{k \mid k = 1, 2, ..., M\}$.
$\mathcal{S}$	the set of nodes connecting WSNs and Cloud $\mathcal{S} = \{S_k \mid k = 1, 2, ..., M\}$.
$\mathcal{N}'$	the set of nodes including the sink of WSN $\mathcal{N}' = \mathcal{S} \bigcup \mathcal{N}$.
h_i^k	the number of requests for item k from demand node i during a certain period T_0.
f_j	fixed cost to create a data duplication at node j.
d_{ij}	distance from node i to node j.
α	cost to transfer unit item per unit distance from source node s to cache node j.
β	cost to transfer unit item per unit distance from cache node j to demand node i.
x_j^k	data allocation decision variable $x_j^k = \begin{cases} 1, \text{node } j \text{ caches node for data item } k \\ 0, \text{otherwise.} \end{cases}$
y_{ij}^k	data access decision variable $y_{ij}^k = \begin{cases} 1, \text{node } i \text{ accesses item } k \text{ from node } j \\ 0, \text{otherwise.} \end{cases}$

4. Single-type data allocation problem

4.1 Mathematical Formulation of SDAP

First of all, we consider the simple scenario that only one WSN connects to the cloud. For the sake of simplicity, we just use s to denote the single source node in single-type data allocation problem, and symbols like h_i^k defined in section 3 is simplified as h_i here. With the notation above all, we can formulate the single-type data allocation problem as follows.

$(SDAP)$

$$\min \alpha \sum_{j \in N} d_{sj} x_j + \sum_{j \in N} f_j x_j + \beta \sum_i \sum_{j \in N'} h_i d_{ij} y_{ij} \quad (1)$$

$s.t.$

$$\sum_{j \in \mathcal{N}'} y_{ij} = 1, \forall i \in \mathcal{N} \quad (2)$$

$$x_j \geq y_{ij}, \forall i, j \in \mathcal{N} \quad (3)$$

$$x_j \in \{0, 1\}, \forall j \in \mathcal{N} \quad (4)$$

$$y_{ij} \in \{0, 1\}, \forall i \in \mathcal{N}, j \in \mathcal{N}' \quad (5)$$

The objective function (1) minimizes the total cost which is the sum of the assigning cost, the sum of placing cost, and the sum of accessing cost. Constraint (2) stipulates that each demand node i gets data from exactly one node (the source node or a cache node). Constraint (3) means that requests from demand node i cannot be responded by node j unless data is cached at node j. Constraints (4) and (5) define the nature of the decision variables. In the single-type data allocation problem, it is assumed that node capacity is large enough to store the data generated from one WSN during period T_0. So node capacity constraints are not considered in the formulation. From the mathematical formulation, if we ignore the assigning costs described by the first part of expression (1), our problem during a period T_0 is presented as an *uncapacity fixed charge facility location problem*[15], [16], [17].

4.2 A Lagrangian Relaxation Algorithm for SDAP

Firstly, we use Lagrangian Relaxation method to get an approximate optimal solution for the single-type data allocation problem, and use this solution to benchmark heuristic approach proposed later. We consider relaxing constraint (2), $\sum_{j \in N'} y_{ij} = 1$, to obtain the following problem:

$(LR - SDAP)$

$$\max_{\lambda} \min_{x,y} \alpha \sum_{j \in N} d_{sj} x_j + \sum_{j \in N} f_j x_j + \beta \sum_i \sum_{j \in N'} h_i d_{ij} y_{ij}$$

$$+\sum_i \lambda_i[1 - \sum_{j\in\mathcal{N}'} y_{ij}]$$
$$= \sum_{j\in\mathcal{N}}(\alpha d_{sj} + f_j)x_j + \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}'}(\beta h_i d_{ij} - \lambda_i)y_{ij}$$
$$+ \sum_i \lambda_i \quad (6)$$

$s.t.$

$$y_{ij} \le x_j, \forall i, j \in \mathcal{N} \quad (3)$$
$$x_j \in \{0,1\}, \forall j \in \mathcal{N} \quad (4)$$
$$y_{ij} \in \{0,1\}, \forall i \in \mathcal{N}, j \in \mathcal{N}' \quad (5)$$
$$\lambda_i \in \Re, \forall i \quad (7)$$

For fixed values of the Lagrange multipliers λ_i, we want to minimize expression (6). Consider first the problem involving the access decision variables, y_{ij}. If $\beta h_i d_{ij} - \lambda_i \ge 0$, we can set $y_{ij} = 0$, and if $\beta h_i d_{ij} - \lambda_i < 0$, we would like to set $y_{ij} = 1$. On the other hand, as shown in expression (3), y_{ij} is constrained to be no more than x_j. Thus, for each j, we compute $V_j = \alpha d_{sj} + f_j + \sum_i min\{0, \beta h_i d_{ij} - \lambda_i\}$, if we set $x_j = 1$, the objective function expression (6) will change by the V_j. If $V_j < 0$, it is advantageous to set $x_j = 1$; otherwise, set $x_j = 0$. We note that for $j = s$, d_{sj} and f_j are set to be 0. The objective value of expression (6) is a *lower bound*, denoted by$LB(\lambda)$, on the primal objective function expression (1). On the other hand, as the constraint (2) is relaxed in the Lagrangian Relaxation Approach, the obtained solution is likely to violate the constraint, i.e., some demand nodes may not get any data and others may get the same data from multiple nodes. Suppose that we set all those nodes $x_j = 1$ to be cache nodes, and then route all demand nodes to access the nearest cache node. By this way, we can get a feasible solution to the primal problem, this solution is an *upper bound*, denoted by $UB(\lambda)$, on the objective function expression (1).

To derive bounds using Lagrange relaxation, a sequence of Lagrange multipliers can be revised using a standard subgradient oprimiztion procedure as follows:

$$\lambda_i^{n+1} = \max\{0, \lambda_i^n - t^n(\sum_j y_{ij}^n - 1)\}, \quad (8)$$

where t^n is the nth iteration of the Lagragian procedure, named stepsize, and

$$t^n = \frac{\gamma^n(UB(\lambda^n) - LB(\lambda^n))}{\sum_i(\sum_j y_{ij}^n - 1)^2}. \quad (9)$$

y_{ij}^n is the optimal value of the access decision variable on the nth iteration. γ^n is a constant on the nth iteration, started with $\gamma^1 = 2$ and cut by half every time $LB(\lambda)$ fails to increase after a certain number of iterations. $UB(\lambda^n)$ and $LB(\lambda^n)$ are respectivly the upper and lower bound on the objective function. Furthermore, the initial multipliers are given by

$$\lambda_i^1 = \min_{(i,j)\neq(i,i)} \beta h_i d_{ij}. \quad (10)$$

Note that Lagrangian Relaxation doesn't always converge to the optimal solution. We can stop the procedure when the difference between the bounds is less than a tolerance error ϵ_0 (e.g. 1) or after a certain number of iterations (e.g. 1000). In order to evaluate the performance of a solution, We define an evaluation metric called *optimal gap* according to the lower bound (LB) and objective value (OV) calculated by the solution as follows.

$$optimal\ gap = \frac{OV - LB}{LB}. \quad (11)$$

The optimal gap indicates how close the solution we obtained to the theoretical lower bound. The lower optimal gap is, the better the solution. The Lagrangian Relaxation Approach is detailed in algorithm 1.

4.3 Numerical Experiments for SDAP

To illustrate the procedure of Algorithm 1, we consider a simple network with 8 nodes shown in Figure 3. Node 0 is the only source node in the network. Our work is determining some cache nodes to bake up data for node 0, so that the total costs can be minimized. All demands (h_i) for the data from node i , and all fixed cost f_j for creating a duplication at each node are set equal to 1, while the unit transfer costs α (from source node to cache node) and β (from cache node to demand node) are set to 0.6 and 0.8, respectively. Distance between node i and j is the length of the shortest path between them times a constant $dis(= 10)$. Beginning with all Lagrange multipliers set equal to the initial values of $\min_{(i,j)\neq(i,i)} \beta h_i d_{ij}$ according to equation (10), the Lagrangian procedure converges in 12 iterations to a solution with a total cost of 59.0. Key values at iterations are listed in table 2. Three duplications are

Algorithm 1 A Lagrangian Relaxation Algorithm for SDAP

1: $n = 0, UB(\lambda^n) = \infty, LB(\lambda^n) = 0$
2: **while** $UB(\lambda^n) - LB(\lambda^n) \geq \epsilon_0$ **do**
3: n=n+1
4: Set Lagrange multipliers λ_i^n according to (8) or (10).
5: Solve the relaxed problem for values of the Lagrange multipliers got from step 4.
5.1 for each candidate node j, compute V_j .
5.2 Set $x_j = \begin{cases} 1, \text{if } V_j < 0, \\ 0, \text{if not} \end{cases}$
5.3 Set
$y_{ij} = \begin{cases} 1, \text{if } x_j = 1 \text{ and } \beta h_i d_{ij} - \lambda_i < 0, \\ 0, \text{if not} \end{cases}$
5.4 Calculate the lower bound $LB(\lambda^n)$
6: Convert the relaxed solution into a primal feasible solution and get the upper bound $UB(\lambda^n)$
7: **end while**

cached at node 1, 2, and 4.

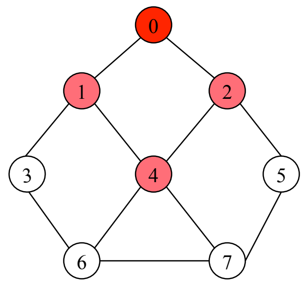

Fig. 3 An example of network used to illustrate algorithm 1 for SDAP.

As shown in table 2, the lower bound and upper bound obtained from the Lagrangian procedure in this example are exactly equal to each other. We get the optimal gap equals 0 for the objective value of 59.0 at 12th iteration, the solution we obtained is optimal. From the distribution of cache data in the network, we can see that caches tend to locate close to the source node because the assigning costs used to assign the latest sensed data to each cache node are also considered in our model.

5. Multi-type Data Allocation Problem

5.1 Uncapacitated Multi-type Data Allocation Problem

We consider two versions, uncapacitated and capacitated, of multi-type data allocation problem. For the uncapacitated Multi-type Data Allocation Problem, the allocation of arbitrary data item k will not affect the allocation of another data item k' at all. The M types data items are completely independent with each other as nodes have no capacity constrains. In other words, the Multi-type Data Allocation Problem with no capacity constrains is exactly an *M-single-type* data allocation problem. It can be described as:

$(UMDAP)$

$$\min \alpha \sum_{k \in \mathcal{M}} \Big(\sum_{j \in \mathcal{N}} d_{sj} x_j^k + \sum_{j \in \mathcal{N}} f_j x_j^k + \beta \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}'} d_{ij}^k h_i^k y_{ij}^k \Big) \quad (12)$$

$$= \min \alpha \sum_{j \in \mathcal{N}} d_{sj} x_j^1 + \sum_{j \in \mathcal{N}} f_j x_j^1 + \beta \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}'} d_{ij}^1 h_i^1 y_{ij}^1$$
$$+ \min \alpha \sum_{j \in \mathcal{N}} d_{sj} x_j^2 + \sum_{j \in \mathcal{N}} f_j x_j^2 + \beta \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}'} d_{ij}^2 h_i^2 y_{ij}^2$$
$$\vdots$$
$$+ \min \alpha \sum_{j \in \mathcal{N}} d_{sj} x_j^M + \sum_{j \in \mathcal{N}} f_j x_j^M + \beta \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}'} d_{ij}^M h_i^M y_{ij}^M \quad (13)$$

$s.t.$

$$\sum_{j \in \mathcal{N}'} y_{ij}^k = 1, \forall i \in \mathcal{N}, k \in \mathcal{M} \quad (14)$$

$$x_j^k \geq y_{ij}^k, \forall i, j \in \mathcal{N}, k \in \mathcal{M} \quad (15)$$

$$x_j^k \in \{0, 1\}, \forall j \in \mathcal{N}, k \in \mathcal{M} \quad (16)$$

$$y_{ij}^k \in \{0, 1\}, \forall i \in \mathcal{N}, j \in \mathcal{N}', k \in \mathcal{M} \quad (17)$$

This problem can be solved by executing M times Lagrangian Relaxation Algorithm for Single-type Data Allocation proposed in section 4.2.

5.2 Capacitated Multi-type Data Allocation Problem

The capacitated version of multi-type data allocation problem has the same objective function(12) with the uncapacitated one. However, the total number of items stored at node j is bound by its capacity Q_j. Thus, the corresponding mathematical formulation can be written as:

$(CMDAP)$

Table 2 Iterations of Algorithm 1 for the Network of Figure 3
($\alpha = 0.6, \beta = 0.8, \epsilon = 1.0E - 5$)

Ite.	Key Values																									
	λ_i								V_j								X_j								UB	LB
1	0.0	8.0	8.0	8.0	8.0	8.0	8.0	8.0	0.0	-1.0	-1.0	5.0	5.0	5.0	11.0	11.0	1	1	1	0	0	0	0	0	70.0	54.0
2	0.52	8.0	8.0	8.52	8.52	8.52	8.52	8.52	-0.52	-2.55	-2.55	3.45	2.94	3.45	8.42	8.42	1	1	1	0	0	0	0	0	70.0	53.48387
3	0.0	8.0	8.0	8.52	7.92	8.52	9.12	9.12	0	-1.52	-1.52	3.37	2.85	3.37	8.25	8.25	1	1	1	0	0	0	0	0	70.0	56.14897
4	0.46	8.0	8.0	8.52	8.38	8.52	9.58	9.58	-0.46	-2.36	-2.36	2.44	1.00	2.44	6.49	6.49	1	1	1	0	0	0	0	0	70.0	55.85618
5	0	8.0	8.0	8.52	7.86	8.52	10.09	10.09	0.06	-1.52	-1.52	2.39	0.95	2.39	6.30	6.30	1	1	1	0	0	0	0	0	70.0	58.04839
6	0.40	8.0	8.0	8.52	8.26	8.52	10.49	10.49	-0.40	-2.18	-2.18	1.59	-0.64	1.59	4.84	4.84	1	1	1	0	1	0	0	0	59.0	57.28222
7	0.20	8.0	8.0	8.52	8.13	8.52	10.49	10.49	-0.20	-1.84	-1.84	1.80	-0.31	1.80	5.18	5.18	1	1	1	0	1	0	0	0	59.0	58.15795
8	0.10	8.0	8.0	8.52	8.06	8.51	10.49	10.49	-0.10	-1.67	-1.67	1.90	-0.14	1.90	5.34	5.34	1	1	1	0	1	0	0	0	59.0	58.58723
9	0.05	8.0	8.0	8.52	8.03	8.51	10.49	10.49	-0.05	-1.59	-1.59	1.94	-0.06	1.94	5.42	5.42	1	1	1	0	1	0	0	0	59.0	58.79766
10	0.02	8.0	8.0	8.52	8.01	8.52	10.49	10.49	-0.02	-1.55	-1.55	1.96	-0.02	1.97	5.46	5.46	1	1	1	0	1	0	0	0	59.0	58.90081
11	0.01	8.0	8.0	8.52	8.00	8.52	10.49	10.49	-0.01	-1.53	-1.53	1.98	4.07	1.98	5.48	5.48	1	1	1	0	0	0	0	0	70.0	58.95133
12	0.0	8.0	8.0	8.52	7.60	8.52	10.89	10.89	-0.01	-1.52	-1.52	1.59	-0.39	1.59	4.70	4.70	1	1	1	0	1	0	0	0	59.0	59.0

$$\min \alpha \sum_{k\in\mathcal{M}}\sum_{j\in\mathcal{N}} d_{sj}x_j^k + \sum_{k\in\mathcal{M}}\sum_{j\in\mathcal{N}} f_j x_j^k$$

$$+\beta \sum_{k\in\mathcal{M}}\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}'} d_{ij}^k h_i^k y_{ij}^k \quad (12)$$

$s.t.$

$$(14), (15), (16), (17),$$

$$\sum_{k\in\mathcal{M}} x_j^k \le Q_j, \forall j \in \mathcal{N}. \quad (18)$$

Constraint(18) means that the amount of data items cached in node j can not exceed the capacity of j.

Relax capacity constraint(18), we have

$$\max_{\mu} \min_{x,y} \sum_{j\in\mathcal{N}}\sum_{k\in\mathcal{M}} (\alpha d_{sj} + f_j - \mu_j) x_j^k$$

$$+ \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}'}\sum_{k\in\mathcal{M}} \beta h_i^k d_{ij} y_{ij}^k + \sum_{j\in\mathcal{N}} \mu_j Q_j \quad (19)$$

$$s.t. \quad \mu_j \ge 0, \forall j \in \mathcal{N}' \quad (20)$$

substituting $\hat{f}_j = (\alpha d_{sj} + f_j - \mu_j)$ and $\hat{Q} = \sum_j \mu_j Q_j$, we obtain the following objective function for the fixed values of μ_j:

$(LR - CMDAP)$

$$\max_{\mu} \min_{x,y} \sum_{j\in\mathcal{N}}\sum_{k\in\mathcal{M}} \hat{f}_j x_j^k + \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}'}\sum_{k\in\mathcal{M}} \beta h_i^k d_{ij} y_{ij}^k + \hat{Q}_j \quad (21)$$

$s.t.$

$$(14), (15), (16), (17)$$

This problem is exactly the structure of uncapacitated problem (UMDAP) (12) presented in 5.1. For fixed values of $\hat{f}_j$, $\hat{Q}_j$, and μ, the optimal solution of the problem LR-CMDAP can be seen as the lower bound of problem CMDAP. On the other hand, the obtained solution is likely to violate the capacity constraints. As noted in sectione 4.2, Lagrangian Relaxation doesn't always coverage to the optimal solution. So it is not easy to obtain the lower bound of CMDAP. Developing an efficient and acurate algorithm for the UMDAP becomes our future work.

6. Conclusions and future work

This paper formulates tree types of data allocation problems, SDAP, UCMDAP, and CMDAP, based on a novel data cache model considering data assigning cost, data placing cost, and data accessing cost in sensor-cloud. A Lagrangian Relaxation algorithm is proposed to solve the SDAP, and a computational experiment is executed to illustrate the procedure of algorithm. This algorithm can be also used in the UCMDAP to get an approximate solution. Furthermore, it is also important for our future work developing an efficient and acurate algorithm for the UMDAP.

References

[1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless Sensor Networks a Survey," *Computer Networks*, Vo1. 38, No. 4, pp. 393-422, March 2002.

[2] A.A. Abbasi and M. Younis, "A Survey On Clustering Algorithms for Wireless Sensor Networks," *Computer Communications*, Vol. 30, No. 14, pp. 2826-2841, October 2007.

[3] D. Kumar, T.C. Aseri and R.B. Patel. "EECDA: Energy Efficient Clustering and Data Aggregation Protocol for Heterogeneous Wireless Sensor Networks," *Int. J. of Computers, Communications and Control*, Vol. VI, pp. 113-124, March 2011.

[4] Y.L. Tao, Y.B. Zhang, and Y.S. Ji, "Flow-balanced routing for multi-hop clustered wireless sensor networks," *Ad*

Hoc Network, Vol. 11, No. 1, pp. 541-554, January 2013.
[5] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," *National Institute of Standards and Technology (NIST)*, Vol. 145, No. 6, pp. 1-7, September 2011.
[6] Sensor-Cloud, http://sensorcloud.com/system-overview.
[7] M. M Hassan, B. Song, and E. Huh, "A Framework of Sensor - Cloud Integration Opportunities and Challenges," *Proc. 3rd Int. Conf. Ubiquitous Information Management and Communication (ICUIMC)*, pp. 618-626, January 2009.
[8] N. Mitton, S. Papavassiliou, A. Puliafito, and K.S. Trivedi, "Combining Cloud and sensors in a smart city environment," *EURASIP Journal on Wireless Communications and Networking*, No. 247, 2012.
[9] N. Poolsappasit, V. Kumar, S. Madria, and S. Chellappan, "Challenges in Secure Sensor-Cloud Computing," *Lecture Notes in Computer Science*, Vol. 6933, pp. 70-84, September 2011.
[10] M. Kurz, G. Holzl, and A. Ferscha, "Goal-Oriented Opportunistic Sensor Clouds," *Lecture Notes in Computer Science*, Vol. 7566, pp. 602-619, 2012.
[11] A. Alamri, W.S. Ansari, M.M. Hassan, M. S. Hossain, A. Alelaiwi, and M.A. Hossain, "A Survey on Sensor-Cloud: Architecture, Applications, and Approaches," *International Journal of Distributed Sensor Networks*, Vol. 2013, ID 917923, 18 pages, 2013.
[12] B. Tan and L. Massoulie, "Optimal content placement for peer-to-peer video-on-demand systems " *Proc. IEEE INFOCOM*, pp. 694-702, April 2011.
[13] M. Bjorkqvist, L. Y. Chen, M. Vukolic, and X. Zhang, "Minimizing Retrieval Latency for Content Cloud" *Proc. IEEE INFOCOM*, pp. 1080-1088, April 2011.
[14] W. Gao, G. Cao, A. Iyengar, and M. Srivatsa, "Supporting Cooperative Caching in Disruption Tolerant Networks" *Proc. 31st International Conference on Distributed Computing Systems (ICDCS)*, pp. 151-161, June 2011.
[15] R. Z. Farahani and M. Hekmatfar, "Facility Location: Concepts, Models, Algorithms and Case Studies" *Springer-Verlag Berlin Heidelberg*, pp. 177-242, 2009.
[16] Z. Drezner and H. W. Hamacher, "Facility Location: Applications and Theory" *Springer-Verlag Berlin Heidelberg New York*, pp. 37-202, 2001.
[17] M. S. Daskin, "Network and Discrete Location: Models, Algorithms, and Applications" *Wiley-Interscience Series in Discrete Mathematics and Optimization*, pp. 247-303, May 1995.

