Al-Aqsa University Journal (Natural Sciences Series),Vol.21, Pages 90-119,Jan.2017
ISSN 2521-893X

Light Allocation of Tasks in Clustered-based Wireless Sensor Networks

Dr. Yousef E. M. Hamouda*

الملخص

التخصيص الخفيف للمهام في شبكات الاستشعار اللاسلكية المقسمة إلى مجموعات

تهدف مشكلة الدراسة إلى تخصيص وتوزيع المهام في شبكات الاستشعار اللاسلكية (WSNs)؛ ليتم بشكل تعاوني تنفيذ تطبيق معقد يمكن تقسيمه إلى مهام تعتمد على بعضها باستخدام الرسم البياني الموجه(DAG). تم اقتراح خوارزمية التخصيص الخفيف للمهام(LAT)؛ وذلك لتوزيع وتخصيص مهام التطبيق على نقاط الاستشعار، بحيث يتم تعزيز كفاءة الطاقة، وعمر الشبكة ووقت التنفيذ للتطبيق بحيث يخضع الحل لتلبية الموعد النهائي لتنفيذ التطبيق. وتعد خوارزمية التخصيص الخفيف للمهام(LAT) طريقة إرشادية تؤدي إلى التقليل من وقت تنفيذ التطبيق من خلال جدولة المهام الأكبيرة، وترتيب مستويات(DAG) تنازلياً وفقاً لعدد المهام في المستوى، والتي يمكن تنفيذها بالتوازي حيث يتم تحسين عمر الشبكة أيضا عن طريق اختيار نقاط الاستشعارلتعيين المهام بحيث تضم هذه النقاط العدد الأكبر من النقاط المتجاورة والطاقة المتبقية الكبرى.

وتظهر نتائج المحاكاة أن وقت تنفيذ التطبيق وعمر الشبكة تحسنتا في خوارزمية(LAT)، مقارنة مع غيرها من خوارزميات تخصيص المهام المعروفة.

Abstract

The problem of task allocation in Wireless Sensor Networks (WSNs) is addressed to cooperatively execute a complex application that can be divided into dependent tasks using Directed Acyclic Graph (DAG). Light Allocation of Tasks (LAT) algorithm is introduced to map and schedule the application tasks among the sensor nodes so that the energy efficiency, network lifetime and application execution time are enhanced subject to meet the application deadline. The LAT algorithm is a heuristic approach that minimizes the execution time of the application by scheduling the bigger tasks and sorting the DAG level in non-increasing order according to the number of tasks in the level that can be parallelized. The network

* **Computer Department _Faculty of Applied Sciences_ Al-Aqsa University _ Gaza _ Palestine.**

lifetime is also improved by selecting the mapping sensor nodes with the highest number of neighbors and energy remaining.
The simulation results show that the application execution time and the network lifetime are improved in LAT algorithm compared with other well-known task allocation algorithms.

1. Introduction:

WSNs have recently received a clear interest in both civil and military applications due to its interactions with the environments where it provides virtual snapshots of the physical world by interpreting the physical events. WSNs consist of hundreds or thousands of small-size, low-cost sensor nodes connected to each other using wireless communication systems (Akyildizet al., 2002). Each sensor node has four capabilities which are the sensing capability to sense the physical phenomena, processing power with memory and storage to execute the applications and store the developed algorithms, wireless communication unit to exchange messages with other sensor nodes, and electrical power unit to deliver the energy to its components (Potdaret al., 2009). WSNs are easy to be deployed using random or planned deployment manners. For instance, in military ad-hoc applications or disaster events such as earthquake, WSNs can be formed by randomly deploying the sensor nodes from the airplane. On the other hand, in smart home or fire alarm systems, a fixed deploying of sensor nodes is required to set the sensor node in the appropriate place (Estrinet al., 1999).

WSNs have many civil, commercial and industrial applications such as healthcare, health monitoring, weather monitoring, pollution monitoring, environmental monitoring, structural monitoring, machine condition monitoring, wildlife habitat monitoring, navigation, forest fire monitoring, manufacturing job flow, control of moving vehicles, smart homes, inventory control, decision support systems, disaster management, surveillance of people or vehicle, bio-chemical material detection and agriculture. WSNs can also be employed in battlefield and military applications including target field imaging, intrusion detection, detecting illegal crossings, and security / tactical surveillance (Arampatziset al., 2005; Akyildizet al., 2002).

One of the most drawbacks of WSNs is its limited resources in term of processing capability and battery power. Furthermore, WSNs are usually deployed in harsh environments such as space, forests and battlefields. Therefore, it is difficult to physically access the wireless sensor nodes after deployment. In many cases, it is impossible to change or recharge the depleted sensor node battery (Ali et al., 2002). Therefore, maintaining battery life and the energy-efficiency are ones of the most crucial issues in WSNs because it increases the useful network lifetime (Heinzelman et al., 2000). Energy is consumed from the battery during sensing, communication and processing. However, other operational factors can waste energy. For example, sensor nodes waste energy due to collisions, idle listening, over-hearing, protocol overhead and over-emitting (Demirkol et al., 2006).

Complex applications such as camera-based applications require highly computational capability and real time execution (Rinner and Wolf, 2008). In other words, the execution time of the application in WSNs has to meet the application deadline after which the execution of the application will not be useful anymore. Accordingly, parallel systems (Grama et al., 2003) are considered in which multiple processors are concurrently and cooperatively used to execute a single application so that the execution of applications is speeded up. In parallel systems, task mapping assigns resources to tasks and task scheduling determines the execution sequence of the tasks (Sinnen, 2007). It is well known that task mapping is an NP-complete problem where the time required to find the optimal solution increases with the problem size (Fernandez-Baca, 1989).

This paper introduces a heuristic energy efficient task allocation approach called Light Task Allocation (LAT) that aims to execute the complex application using a group of sensor nodes so that the execution time and energy consumption are improved. Many researchers are engaged in developing new approaches which work efficiently and effectively with sensor nodes requirements in order to conserve their power (Tian et al., 2005).

The paper is organized into seven sections including this one. After this introduction, Section 2 critically explores and analyzes the related work into task mapping and scheduling using WSNs. The problem definition and LAT framework are formalized in Section 3 and Section 4, respectively. The details of proposed LAT algorithm are introduced in Section 5. A simulation-based evaluation is presented in Section 6 and, finally, Section 7 summarizes the paper.

2. Related Work:

Task mapping and scheduling are deeply investigated in traditional parallel computing including high performance computing, heterogeneous computing, grid computing and distributed computing systems (Hagras and Janecek, 2003). Nevertheless, the design objectives are different from using the parallel systems in WSNs. In (Braun et al., 2001; Iverson et al., 1995), the main objective is the minimizing of the application execution time. Alternatively, the application execution time in case of WSNs should not exceed the application deadline. Furthermore, energy consumption and wireless communication constraints are not considered in traditional parallel processing systems. Therefore, the current approaches of task mapping and scheduling in traditional parallel processing systems can not directly apply to WSNs. A number of researchers have already considered the task mapping and scheduling in WSNs.

In (Jin et al., 2012; SHAMS and KHAN, 2012), Genetic Algorithm (GA) is entailed to provide well-performing task allocation. A modified binary version of the basic Particle Swarm Optimization algorithm (MBPSO) is introduced in (Yang et al., 2004) to find the optimal task allocation solution. In (Ferjani et al., 2016), logic gate-based evolutionary algorithm is used to solve the problem of task allocation in WSNs. However, all the researches into task allocation in WSNs using evolutionary techniques require high processing and time to be executed, which is not suitable in limited resources WSNs.

In(Papataxiarhis, 2016), integer linear programming is adopted to optimally assign complex tasks to sensor nodes so that the total energy consumption is minimized. Task allocation is explored also in (Abdelhak et al., 2010) so that the energy consumption and network life time are improved. However, the execution time is not considered in (Papataxiarhis, 2016) and (Abdelhak et al., 2010), which causes the application to take long time to be executed.

In(Giannecchini et al., 2004), a fast online collaborative allocation algorithm (CoRAl) is proposed to dynamically reconfigure WSNs according to the activity changes of sensor node (i.e. sleep versus active modes) or new hot spots occurring (e.g. new target is detected). However, CoRAl does not consider the battery level as a part of sensor node resources and it does not address the energy consumption problem.

In(Shivle et al., 2004), six heuristic task mapping and scheduling techniques are compared and evaluated in heterogeneous ad hoc grid environment. The simulation results in (Shivle et al., 2004) show that GA gives the best performance which is defined as the summation of the percentage of energy consumed by each sensor node to complete the mapped tasks, averaged across all sensor nodes. On the other hand, the time required to perform GA is high compared to Min-Min. However, unlike the case of WSNs, the heuristic techniques in (Shivle et al., 2004) assume individual channels for each sensor node and each sensor node can transmit and receive data at the same time. Moreover, (Shivle et al., 2004) ignores the energy consumption to receive a data item and the cost of the initial data item.

In(Yu and Prasanna, 2005), a task allocation heuristic algorithm that consists of three operational phases has been developed to provide energy-balanced task allocation in a single-hop cluster of homogeneous sensor nodes. However, (Yu and Prasanna, 2005) assumes the energy consumption to transmit a data item is the same in the sender and receiver, which is not realistic. Additionally, (Yu and Prasanna, 2005) does not employ the broadcast nature of WSNs where sensor nodes are equipped with Omni-directional antennas.

In (Tian et al., 2005), Energy-Constrained Task Mapping and Scheduling (EcoMapS) algorithm is implemented for energy-constrained application in single-hop clustered. However, EcoMapS does not guarantee the in-time completion of the application before the application deadline. In (Tian et al., 2006), a real-time task mapping and scheduling (RT-MapS) algorithm is proposed for collaborative in-network processing in single-hop cluster WSN with enabling Dynamic Voltage Scaling (DVS) feature. In (Tian and Ekici., 2007), Multihop Task Mapping and Scheduling (MTMS) solution is presented to map and schedule application tasks in multi-hop cluster WSN. However, MTMS and RT-MapS do not allow mapping the task to its immediate predecessors, which causes to use more sensor nodes for mapping. Additionally, they involve all sensor nodes in the task mapping decision-making. Moreover, the Min-Min algorithm adopted by them is initially introduced in traditional parallel computing for mapping and scheduling independent tasks. Therefore, there are no any dependencies among tasks and in turn there is no communication cost between the processors. In Min-Min, the fitness value for each task is calculated across all sensor nodes and thus the sensor node that has a minimum fitness value is temporality selected and stored with the corresponding task in a pair. Among all node/task pairs, the pair that has the minimum fitness value is selected for mapping. Therefore, in Min-Min approach, only the selected pair will be permanently mapped and the procedures will be repeated to map other tasks. In case of an application that can be divided into tasks with dependencies, the first pair is permanently selected based on the other pairs and communication between the pairs to exchange the dependencies. If the same procedures are repeated to permanently map the next pair, the calculations that are used to map the first pair will not be valid anymore because the pairs that are produced to permanently map the second pair may not be the same as the pairs generated to permanently map the first pair.

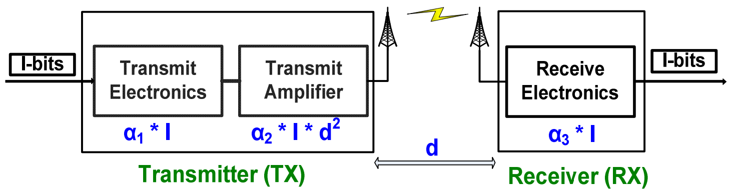
In (Hamouda and Phillips, 2009), Biological Task Mapping and Scheduling (BTMS) approach is introduced where the application is executed by a group of sensor nodes so that the execution time and energy consumption are improved. In (Bolla et al., 2017), the Topology-Aware Task Allocation and Scheduling (TATAS) is a heuristic approach to map and schedule the tasks to the sensor nodes. However, TATAS does not consider the dependencies among application tasks. In (Devi and Muthuselvi, 2016), Task Level Parallelism (TLP) is introduced to parallelize the execution of different health parameters in smart health care applications using WSN so that the processing time is reduced. Nevertheless, scheduling the task execution is not considered. An energy efficient Complicated Task Solution scheme for real-time task processing based on node Cooperation (CTSC) is presented in (Jiang et al., 2016) to allocate more tasks to sensor nodes with a higher energy-level. However, CTSC maps are all dependent tasks to the same sensor nodes which could cause an exhausting for the energy level of sensor node.

In this paper, a heuristic approach for task mapping and scheduling in WSNs called LAT algorithm is introduced to collaboratively process the application tasks using multiple sensor nodes so that the total energy consumption is minimized and the network lifetime is improved subject to meet the application deadline.

3. Problem Formalization and Definition

3.1 Energy Consumption Model

Energy is consumed during sensing, communication and processing activities. As in (Heinzelman et al., 2002; Wang and Chandrakasan, 2002), the transmitter consumes energy to run the radio electronics and the power amplifier while the receiver consumes energy to run the radio electronics. This is shown in Fig 1.



Energy Consumption Model

Therefore, the energy consumption of sensor s_i to transmit l-bit message to the sensor s_j over a distance d_{ij} is:

$$\psi_{TX}(l, d_{ij}) = \alpha_1 . l + \alpha_2 . l . d_{ij}^2 \quad (1)$$

where α_1 is the electronic energy required to transmit one bit that depends on factors such as coding, modulation and filtering, and α_2 is related to the radio energy. The energy consumption to receive an l-bit message is:

$$\psi_{RX} = \alpha_3 . l \quad (2)$$

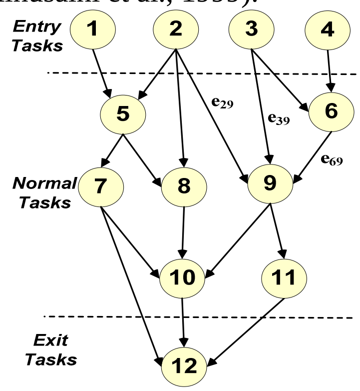
Given a CPU with clock frequency f, the energy consumption to execute N clock cycles (Miller and Vaidya, 2005) is:

$$\psi_{comp}(V_{dd}, f) = NCV_{dd}^2 + V_{dd}\left(I_o e^{\frac{V_{dd}}{nV_T}}\right)\left(\frac{N}{f}\right) \quad (3)$$

where, $f \approx K(V_{dd} - c)$, V_T is the thermal voltage and C, I_o, n, K are CPU dependent parameters.

3.2 Application Model

In parallel system, the application is decomposed into small tasks with dependencies among them (Sinnen, 2007). Direct Acyclic Graphs (DAG) is employed to model the tasks along with dependencies. Fig. 2 shows an example of application DAG (Yu and Prasanna, 2005; Wang and Chandrakasan, 2002; Alhusaini et al., 1999).



Application of DAG

The DAG is model as $A = (\mathbf{V}, \mathbf{E})$, where $\mathbf{V}$ is a set that represents "n" application tasks, $\mathbf{V} = \{v_i | i = 1,2,...n\}$ and $\mathbf{E}$ is a set that represents "q" communication dependencies, $\mathbf{E} = \{e_k | k = 1,2,...q\}$. The edge $e_k \in \mathbf{E}$ between the tasks v_i and v_j is denoted as e_{ij}, where v_j is called the immediate successor of v_i and v_i is called the immediate predecessor of v_i. Therefore, the task cannot be executed until it receives all the results from its immediate predecessors. This dependency between tasks execution is called the communication dependencies constraint. The tasks without immediate predecessors are an entry-task or a source-task while a task without immediate successors is an exit-task or a sink-task. In WSNs, the entry-tasks are used for sensing or gathering the raw data to detect physical phenomena. Therefore, task placement constraints can be defined as only one source task can be assigned to the sensor node. In Fig. 2, v_1 to v_4 are source-tasks, v_{12} is the sink-task, v_1 and v_2 are the immediate predecessors of v_5, and v_{10} is the immediate successor of v_8. The task v_9 cannot be executed until it receives the communication edges (i.e. dependencies), e_{29}, e_{39} and e_{69} from tasks v_2 , v_3 and v_6 respectively.

Each task, $v_i \in V$ of the application of DAG can be modeled as a tuple of the form: $\{N_{v_i}, t_{v_i}, E_{v_i}\}$. N_{v_i} is the number of the task computational cycles, E_{v_i} is the computational energy consumption to execute it which can be calculated using Equation (3) and t_{v_i} is the task execution time given by: $t_{v_i} = N_{v_i}/f_{s_i}$ where f_{s_i} is the CPU clock frequency of sensor node, s_i that executes the task. Each edge, e_{ij} of the application of DAG between the tasks v_i and v_j is also modeled as a tuple of the form: $\{l_{e_{ij}}, t_{e_{ij}}, E_{e_{ij}}\}$. $l_{e_{ij}}$ is the communication data size generated from v_i and required to execute v_j. $E_{e_{ij}}$ and $t_{e_{ij}}$ is the communication energy consumption and communication time required to send e_{ij} from the sensor node that executes the task v_i to the sensor node that executes the task v_j which can be calculated as (Kurose and Ross, 2008):

$$E_{e_{ij}} = E_{e_{ij}}^{TX} + E_{e_{ij}}^{RX}$$

(4)

$$t_{e_{ij}} = t_{e_{ij}}^{t} + t_{e_{ij}}^{p} = \frac{l_{e_{ij}}}{R_b} + \frac{d}{c}$$

(5)

where, $t_{e_{ij}}^{t}$ is the transmission time, $t_{e_{ij}}^{p}$ is the propagation time, d is the distance between the sensor node that exchanges the edge, c is the speed of the light, R_b is the data rate, $E_{e_{ij}}^{TX}$ is transmitted energy consumption dissipated from the source node calculated using Equation (1), and $E_{e_{ij}}^{RX}$ is received energy consumption dissipated from the destination node calculated using Equation (2). $E_{e_{ij}}$ and $t_{e_{ij}}$ is equal to zero if the tasks v_i and v_j are mapped to the same sensor node.

3.3 Wireless Sensor Netwok Model

The proposed WSN consists of "m" sensor nods deployed in the area of interest to perform a specific application. The sensor node, s_i is modeled as a tuple of several properties and states as: $s_i = \{ID_{s_i}, x_{s_i}, y_{s_i}, E^r_{s_i}(k), f_{s_i}, a_{s_i}\}$ where ID_{s_i} is the sensor node identification, x_{s_i}, y_{s_i} are the xy coordinate of sensor node, $E^r_{s_i}(k)$ is the battery energy remaining of sensor node at time k, a_{s_i} is the time at which the sensor node is available to execute a task and f_{s_i} is the processing speed of sensor node. Therefore, the proposed WSN is model as a set of sensor nodes, $\mathbf{S}_{\mathrm{wsn}} = \{s_i | i = 1,2,..m\}$. Sensor modes s_i and s_j can directly communicate if the distance between them, d_{ij} is less than or equal to the radio range, R_r where:

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$
(6)

Therefore, at time k the sensor node, s_i has a set of $m_{s_i}(k)$ neighbors, $\mathbf{N}_{s_i}(k)$ where: $\mathbf{N}_{s_i}(k) = \{s_k | \forall k \text{ satisfies } d_{ik} \le R_r\}$.

3.4 Problem Formalization and Objective Functions

LAT algorithm adopts a dynamic cluster-based topology. At time step k, a target sensor node (TSN), s_{TSN} in a cluster $\mathbf{S}_{\mathrm{clu}}$ triggers a request to collaboratively execute an application with predefined deadline, D where $s_{TSN} \in \mathbf{S}_{\mathrm{clu}}$ and $\mathbf{S}_{\mathrm{clu}} \subset \mathbf{S}_{\mathrm{wsn}}$. The crucial aim of LAT algorithm is to get the task/node pair, $P^*(v,s)$ used to execute the application where $P^*(v,s)$ shows the "n" mapped tasks of application of DAG (the set $\mathbf{V}$) with its corresponding "n_c" assigning sensor nodes which are the set $\mathbf{S}_c \subset \mathbf{S}_{\mathrm{clu}}$ of the sensor nodes selected by LAT algorithm to execute the application tasks. $P^*(v,s)$ is obtained so that the total energy consumption is minimized and the network lifetime, LT_{wsn} is maximized subject to meet the application DL. According to the following objective functions:

$P^*(v,s) = \arg min_{P(v,s)} \{E_T(P(v,s))\}$

(7)

$P^*(v,s) = \arg max_{P(v,s)} (LT_{wsn})$

(8)

Subject to:

$ms(A) \leq DL$

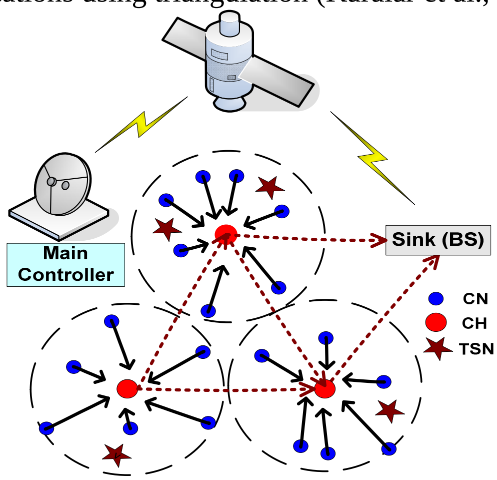
(9)

where, $E_T(P(v,s))$ is the total energy consumption required to execute the application using the task/node pair $P(v,s)$ and $ms(A)$ is the makespan of the application of DAG. The makespan is defined as the time at which the application execution is finished.

4. Network Architecture

As shown in Fig 3, a dynamic cluster-based topology is adopted in this research (Soe, 2008). The cluster head (CH) is placed in the middle of the cluster and equipped with high power and processing units. The other sensor nodes in the cluster is called cluster node (CN) and equipped with normal power and processing units. The radius of the cluster is equal to the radio range of the underlying wireless communication systems. The sink node (SN) relays the messages to the servers, users and headquarters via cellular network, Internet or satellite communication. Therefore, the communication between the CNs and its CH is a single-hop communication, and the communication between the CHs and SN is a multi-hop communication through the CHs hops. The locations of sensor nodes are necessary for the proposed LAT algorithm. Sensor node knows its location using Global

Positioning System(GPS) (Bulusu et al., 2000). Nonetheless, only few sensor nodes use GPS to know its locations and other sensor nodes can calculate their locations using triangulation (Karalar et al., 2004).



Network Architecture for LAT Agorithm

The main purpose of the proposed LAT algorithm is to parallelize the application tasks among the cluster nodes so that the total energy consumption is minimized and the network lifetime is improved subject to satisfy the deadline of the application. The proposed scheme operates in four steps. Step (1) the target sensor node (TSN) triggers a request to execute an application. Therefore, TSN sends an execution_request (exe_req) message to its CH. The exe_req message includes the application tasks needed to be collaboratively executed. Step (2) CH performs the LAT algorithm based on the current state of the cluster. Step (3) CH sends an LAT_request (LAT_req) message to the CNs that are selected to execute the application. The LAT_req message includes the tasks with their assigned node (i.e. task/node pairs) along with information about the dependencies required to execute the tasks. Step (4) the CH sends the final results of the application execution to the TSN via execution_response (exe_res) message.

The LAT messages are denoted as $m_{s,d}$ where "s" denotes the source sensor node of the message and "d" is the destination sensor node. The message can contain either actual data or control information. The LAT messages can be defined as a tuple of several fields as: $m_{s,d} = \{s_s, s_d, T, D, STAMP\}$ where s_s, s_d are the source and destination message, T is the message type, D is the data and $STAMP$ is the time stamp.

5. The Proposed LAT Algorithm

The total processing energy consumption (called serial energy consumption) and the total processing execution time (called serial execution time) required to computationally execute the application tasks are determined as follows:

$$E_T^p = \sum_{k=1}^{n} E_{v_k}$$
(10)
$$t_T^p = \sum_{k=1}^{n} t_{v_k}$$
(11)

The total communication energy consumption and the total communication execution time required to exchange the dependences of the application tasks are determined as follows:

$$E_T^c = \sum_{k=1}^{q} E_{e_{ij}}$$
(12)
$$t_T^c = \sum_{k=1}^{q} t_{e_{ij}}$$
(13)

The overall energy consumption and time required to execute the application tasks using node/task pair, $P(v, s)$ are calculated as:

$$E_T[P(v,s)] = E_T^p + E_T^c$$
(14)

$t_T[P(v,s)] = t_T^p + t_T^c$

(15)

Each task, v_i mapped to sensor node, s_j is starting to be executed at the starting executing time of the task $t_s(v_i, s_j)$. The task is executed when the sensor node is available after it receives all the task dependencies. Thus, $t_s(v_i, s_j)$ is calculated as:

$t_s(v_i, s_j) = \max\{a_{s_i},\ t_{max}[pred(v_k)]\}$

(16)

where, $t_{max}[pred(v_k)]$ is the maximum time at which the last predecessor of task is received by the node, s_j and a_{s_i} is the time at which the sensor node, s_i is available. When the sensor node s_j starts to execute the tasks, it finishes after the task execution time. The time at which the task is completely executed is called the task finish which is given by:-

$t_f(v_i, s_j) = t_s(v_i, s_j) + t_{v_i}$

(17)

$t_T[P(v,s)]$ defined in Equation (15) is total communication and computation time required to execute the application using the cluster nodes. Due to the parallelism, the makespan could be less than $t_T[P(v,s)]$. The makespan of the application of DAG can be calculated as follows:

$ms(A) = max_{\forall s_i \in S_c}\{\ t_f(v_i, s_j)\ \}$

(18)

As described in Equation (7) and Equation (8), the task mapping and scheduling solution, $P^*(v,s)$ is obtained so that the total energy consumption is minimized and the network lifetime, LT_{wsn} is maximized. Therefore, the fitness function includes the energy remaining of the sensor node, $E_{s_i}^r(k)$ at time step, k normalized by the maximum (i.e. initial) energy level E_m. In fact, the death of sensor nodes causes empty gaps or holes in the network and in turn, the connectivity of the network is degraded. Therefore, the network lifetime contributed by the sensor node, s_i at time step, k is modeled as the number of neighbors, $m_{s_i}(k)$ of this sensor node at time step, k. The fitness function of sensor node, s_i at time step, k is formalized as:

$$f_{obj}(s_i, k) = \left[\beta * \frac{E_{s_i}^r(k)}{E_m}\right] + \left[(1-\beta) * \frac{m_{s_i}(k)}{m}\right]$$

(19)

where $0 \leq \alpha \leq 1$ is a weighted controlled parameter. The complete LAT algorithm is shown in Fig. 4. The application of DAG is firstly converted to level-based DAG where the task immediate predecessors in each level are only placed at the lower levels (Shivle et al., 2004). Then, the weighting parameter β is selected according the current system requirements. After that, the sensor node belongs to the cluster with the highest fitness value is selected to map all the application tasks. Next, the makespan of the assigned tasks is calculated. As shown in Line 11 of the LAT algorithm, the parallelism process continues until the maskespan is less than or equals a predefined application deadline. Since the tasks in the same level are independent, they can be executed at the same time. Therefore, for each parallelism step, the level with the maximum tasks that can be parallelized is obtained. This can be performed by computing the difference, $P(L_v)$ between the number of tasks in the level, C_v and the number of sensor nodes by which the tasks of the level are assigned C_s. The level with the maximum difference L^* is chosen to map its task. After that, For more parallelism efficiency and energy saving, the task with the largest computation load is selected to be mapped to the sensor node with minimum total energy consumption.

LAT Algorithm

input: DAG, $G(V,E)$ and cluster nodes S_{clu}
output: Mapped task/node pairs, $F(v,s)$
begin
DAG conversion to Leveled-based DAG;
select $\beta \in [0 \quad 1]$;
calculate $f(s_j)$ for each node;
get the node $S^* \in S_{clu}$ that has the maximum, $f(s_j)$;
allocate all tasks into S^*;
calculate the makespan of the DAG;
update $F(v,s)$;
while (makespan < DL) **do**:
remove S^* from S_{clu};
for each level $L_k \in G(V,E)$ **do**:
get the number of mapped nodes in level, C_s;
get the number of tasks in level, C_v;
calculate $P(L_k) = C_v \quad C_s$;
end for;
get the level, L^* with the maximum, $P(L_k)$;
get the largest unparalleled task, v^* in the level L^*;
allocate the task, v^* to the node, $S^* \in S_{clu}$ that minimize the total energy consumption;
compute makespan;
end while;
Finish;

LAT Algorithm

6. Simulation Results

In this section, the proposed LAT algorithm is evaluated and compared with well-known task allocation algorithms. In the beginning of this section, the simulation assumption is introduced. After that, the simulation results of the LAT algorithm are explained and critically assessed.

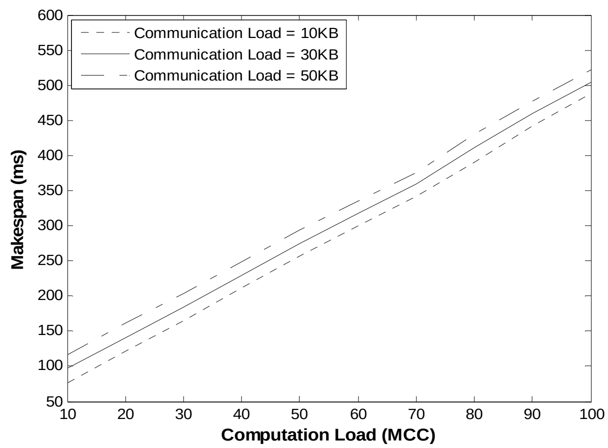
6.1 Simulation Assumption

In this section, the proposed LAT algorithm is evaluated and compared with other known algorithms using event-driven simulation. C++ is used to build the simulation environment using an Intel Core i5 2.2 GHz processor and 4GB memory. Sensor nodes of m=350 are randomly deployed in an area of 400m × 400m. Two nodes are in the same coverage area if the distance between them is equal to or less than the radio range, which is set to 50m. Line of Sight (LOS) communication is assumed between the nodes within the same coverage area. The energy model parameters are set as follows: α_1=100 nJ/b, α_2=1 pJ/bm2, α_3=100 nJ/b, V_T=26mV, C=0.67nF, I_o=1.196mA, n=21.26, K=239.28, c=0.5 and f=100MHz.The energy level, E_m of sensor nodes is set to 10J. The Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA) protocol (Bianchi, 2000) is modeled in the simulator as the MAC layer protocol with transmission speed of 1 Mb/s. Unless specifically mentioned, the weighting parameter, β in Equation (19) is set to 0.5, the simulation is terminated after 400 seconds and TSN is selected randomly during the simulation. The simulation is repeated and the results are averaged for 250 different application DAGs. For each DAG, the number of entry tasks, normal tasks and exit tasks are set to 5, 10 and 1 respectively. The immediate successors of entry and normal tasks are selected randomly from one to three successors. The entry tasks should not have any immediate predecessors and the exit tasks should not have any immediate successors. The normal tasks must have at least one immediate predecessor.

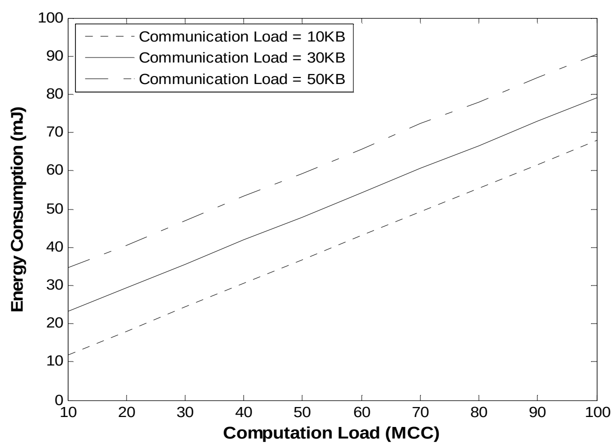
6.2 LAT Algorithm Evaluation

In this subsection, the simulation results for the LAT evaluation are introduced, discussed and compared with other well-known approaches. The makespan, energy consumption, speedup, dead nodes count, cluster size and average neighbor count are plotted and discussed for different operation conditions. The simulator uses events in its operation. The event stores in a list which is updated when the events are scheduled and triggered. There are some events related to the CSMA/CD such as events for CSMA/CD timers, back-off, transmitting, receiving, acknowledgment, retransmission and collision. Other events are related to LAT algorithm such events for TSN request, sensor selection and LAT execution. At the beginning of the simulation, initialization routine is executed to initialize the simulation time and generate the sensors, targets, DAG and WSN models. After that, the event of TSN request is scheduled. Consequently, CSMA/CD events are scheduled to select the cluster that is responsible to execute the application. Finally, LAT algorithm events are scheduled to perform the LAT algorithm. Statistical counters are updated and printed during the simulation to collect the simulation results.

Fig. 5 and Fig. 6 show the variation of makespan and energy consumption with the computing load for different communication load values. The makespan and energy consumption increase with increasing the computation load because more processing cycles required to execute bigger tasks. In addition, for more communication load, the makespan also increases because the total transmission time increases with increasing the communication load. Similarly, for more communication load, the energy consumption increases because the sensor nodes dissipate more communication energy for bigger communication load. When the communication load increases from 10KB to 30KB, the average makespan and energy consumption increase to 7% and 28%, respectively. Therefore, the energy consumption is more sensitive to the communication load variations compared with execution time. In fact, the energy consumption due to the communication load is the cost of the parallelism in WSNs.

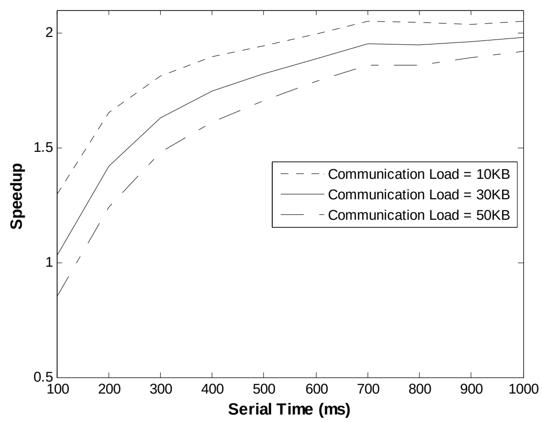


Makespan versus computation load for different communication load



Energy consumption versus computation load for different communication load

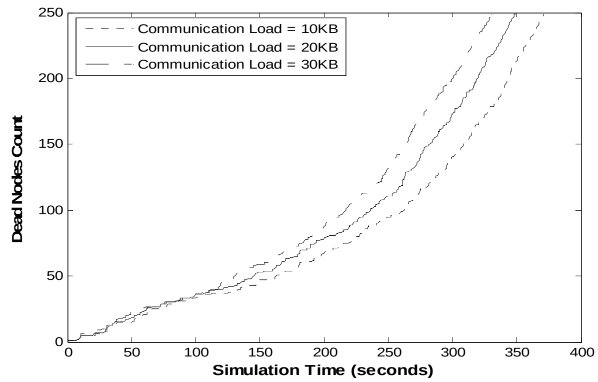
The speedup is defined as the ration between the serial execution time (i.e. sequential execution time) to the parallel execution time (i.e. makespan). Fig. 7 shows the speedup versus the serial time for different communication load. With increasing the serial time (i.e. the computation load), the speedup increases because the parallelism is more beneficial for higher computational load. The communication load influences negatively the speedup because of communication time overhead.



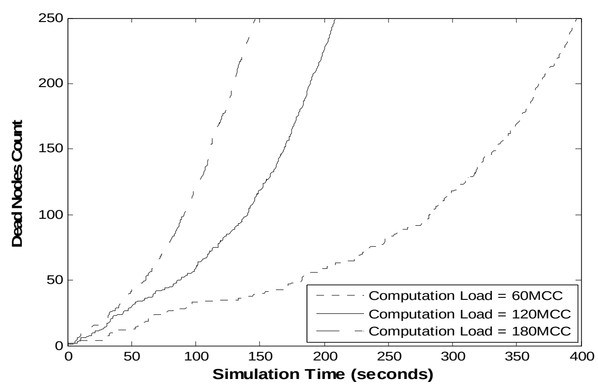
Speedup versus serial time for different communication loads

Fig. 8 and Fig. 9 show the number of dead nodes with the simulation time for different communication and computational loads. Since the sensor nodes consume more energy with time for tasks execution and communication, the dead node count increases with simulation time. The dead node count increases with increasing communication and computational loads because more communication and computational energy are consumed to execute the application of tasks. As shown in Fig. 8 and Fig. 9, when the communication and computational loads increase to 50% of their values, the rate of dead node count increases in case of communication load is less than the one in case of computational load.

Therefore, the dead node count is more sensitive to the computing load compared with communication load.

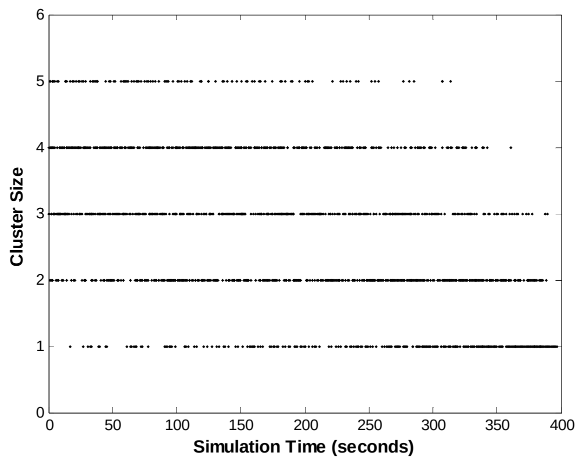


Dead nodes count versus simulation time for different communication loads



Dead nodes count versus simulation time for different computation loads

The cluster size, n_c with the simulation time is plotted in Fig. 10 where the application of DAG is randomly generated for each run. The sensor nodes consume energy for communication and processing. Thus, the death of sensor nodes increases with the time. Therefore, the larger cluster sizes such as $n_c = 5$ and $n_c = 4$ deceases with the simulation time as shown in Fig 10. On the other hand, the smaller cluster sizes such as $n_c = 1$ and $n_c = 2$ increases with the simulation time. The cluster size of 3 almost appears equally during the simulation time. As a result, the cluster size should be selected according to the dead nodes and network states.

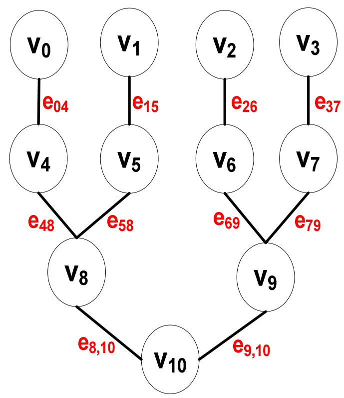


Cluster size versus simulation time

6.3 Comparison with other Known Task Allocation Algorithms

In this section, the proposed LAT algorithm is compared with the BTMS (Hamouda and Phillips, 2009) and MTMS algorithms (Tian and Ekici, 2007). In this comparison, a specific DAG for visual surveillance application shown in Fig. 11 (Tian and Ekici, 2007) is used.

The computation load of tasks $v_0, v_1, v_2, v_3, v_4, v_5, v_6$ and v_7 is 4 Mega Clock Cycle (MCC) and the computation load of other tasks is 2 MCC. The communication load for edges e_{04}, e_{15}, e_{26} and e_{37} is 20B while the communication load for other edges is 40B. In fact LAT algorithm adopts two techniques for minimize the makespan. The first technique is the choice of the DAG level that has the maximum tasks that can be parallelized. The second technique is the choice of the biggest task to be parallelized first. Therefore, as shown in Table 1, the proposed LAT algorithm has the lowest makespan.

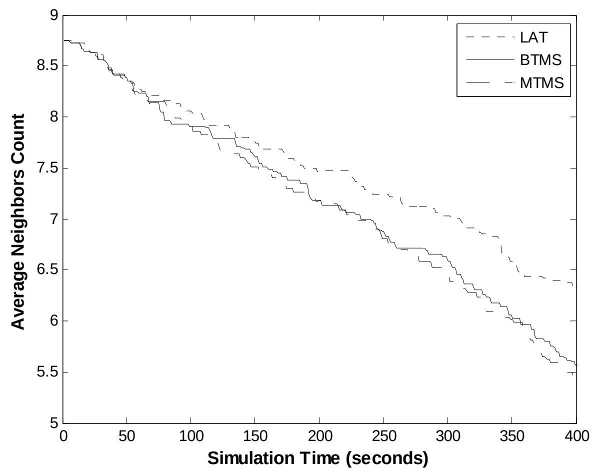


Specific DAG for visual surveillance application

Algorithm	makespan (ms)	Energy Consumption (mJ)
LAT	1280.382	248.458
BTMS	1330.986	248.448
MTMS	1636.418	248.559

Table 1 Makespan and energy consumption for different task allocation algorithms

The average number of sensor node neighbors $m_{av}(k)$ (i.e. average neighbors count) at time step k is defined as $m_{av}(k) = \sum_{i=1}^{m} m_{s_i}(k)$. In Fig. 12, the average neighbors count versus the simulation time is plotted for LAT, BTMS and MTMS algorithms. In Fig. 12, the average neighbors count reduces with time for all algorithms because the death of sensor nodes increases with time due to energy consumption for communication and processing. As shown in Equation (19), the proposed LAT algorithm considers the neighbors count in task allocation to reduce the network gaps and hence improves the network lifetime and coverage. Furthermore, the energy remaining of sensor nodes is also considered for load balancing among sensor nodes. Consequently, the proposed LAT algorithm has the best performance compared with BTMS and MTMS algorithms. In addition, the BTMS algorithm has better average neighbors count compared with MTMS algorithm because it considers adopts the decision rules to map the tasks (Hamouda and Phillips, 2009).



Average neighbours of sensor nodesforfor different algorithms

7. Conclusion

This research considers the task mapping and scheduling in WSNs. The complex application is divided into dependent tasks using DAG. The LAT algorithm is introduced to allocate tasks to the sensor nodes so that the energy efficiency, network lifetime and application execution time are enhanced subject to meet the application deadline. To improve the network lifetime, the fitness value of sensor node is modeled as the weighted sum of its remaining energy and number of neighbors. The LAT algorithm reduces the makespan by mapping the bigger tasks first and the tasks in the DAG level with maximum tasks that can be parallelized.

The impact of communication and computing loads in makespan and energy consumption are evaluated. The dead nodes count is more sensitive to the computation load compared with communication load. Furthermore, the energy consumption is more sensitive to the communication load variations compared with execution time. The LAT algorithm is compared with the BTMS and MTMS algorithms and shows better performance in terms of makespan and the average neighbors count that is indicated to the network lifetime.

8. References

AbdelhakS., Gurram C.S., Ghosh S. and BayoumiM. 2010:Energy-balancing task allocation on wireless sensor networks for extending the lifetime, Proceedings in 53rd IEEE Int. MWSCAS, 781-784.

Akyildiz I.F., Su W., Sankarasubramaniam Y. and Cayirci E. 2002:Wireless sensor networks: A survey, Computer Networks: The International Journal of Computer and Telecommunications Networking, 38(4), 393-422.

AlhusainiA.H., PrasannaV.K. and RaghavendraC.S. 1999: A unified resource scheduling framework for heterogeneous computing environments, Proceedings in IEEE 8th Heterogeneous Computing Workshop 1999. (HCW 1999), 156-165.

Ali M., Bohm A. and JonssonM. 2008: Wireless Sensor Networks for Surveillance Applications: A Comparative Survey of MAC Protocols, Proceedings of the fourth IEEE international conference on wireless and mobile communications (ICWMC 2008), 399-403.

Ferjani A.A., LiouaneN. and KacemI. 2016: Task allocation for wireless sensor network using logic gate-based evolutionary algorithm, International Conference on Control, Decision and Information Technologies (CoDIT), 654-658.

Arampatzis T., Lygeros J. and Manesis S. 2005: A Survey of Applications of Wireless Sensors and Wireless Sensor Networks, Proceedings of the IEEE 13th Mediterranean Conference on Control and Automation, 719–724

BianchiG. 2000: Performance Analysis of the IEEE 802.11 Distributed Coordination Function, IEEE J. Sel. Areas Commun., 18(3).

Bolla D. R., Jijesh J. J., Pramod M. S. and Shivashankar. 2017: Real-Time Data Fusion Applications in Embedded Sensor Network using TATAS, Indian Journal of Science and Technology, 10 (13), 1-7.

Braun T.D., Siegel H.J., Beck N., BoloniL., Freund R.F., HensgenD., MaheswaranM., Reuther A.I., Robertson J.P., Theys M.D. and Yao B. 2001: A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems, Journal of Parallel and Distributed Computing, 61(6), 810-837.

Bulusu N., Heidemann J. and Estrin D. 2000: GPS-less low cost outdoor localization for very small devices, Computer science department, University of Southern California.

Demirkol I., ErsoyC., and AlagozF. 2006: MAC Protocols for Wireless Sensor Networks: A Survey, IEEE Commun. Mag., 44(4), 115-121.

Devi K. N. and Muthuselvi R. 2016: Parallel processing of IoT health care applications, 10th IEEE International Conference on Intelligent Systems and Control (ISCO), 1-6

Estrin D., Govindan R., Heidemann J. and Kumar S. 1999: Next Century Challenges: Scalable Coordination in Sensor Networks, Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking, USA, 263-270.

Fernandez-Baca D. 1989: Allocating modules to processors in a distributed system, IEEE Trans. Softw. Eng. , 15(11), 1427-1436.

GiannecchiniS., CaccamoM. and Shih C.S. 2004: Collaborative resource allocation in wireless sensor networks, Proceedings in IEEE 16th Euromicro Conference on Real-Time Systems, 35-44.

Grama A, Gupta A., Karypis G and Kumar V. 2003: Introduction to Parallel Computing, 2nd edn, England: Pearson Education Limited.

Hagras T. and JanecekJ. 2003: A high performance, low complexity algorithm for compile-time job scheduling in homogeneous computing environments, Proceedings of IEEE International Conference on Parallel Processing Workshops (ICPPW’03), 149–155.

Hamouda Y.E.M. and Phillips C. 2009: Biological Task Mapping and Scheduling in Wireless Sensor Network, IEEE International Conference on Communications Technology and Applications, 914 – 919.

HeinzelmanW.R., ChandrakasanA. and BalakrishnanH. 2000:Energy-Efficient Communication Protocol for Wireless Microsensor Networks, Proceedings of the IEEE 33rd Annual Hawaii International Conference on System Sciences (HICSS '00), 1-10.

Heinzelman W.B., ChandrakasanA. and BalakrishnanH. 2002:An application-specific protocol architecture for wireless microsensor networks, IEEE Trans. Wireless Commun., 1(4), 660-670.

Iverson M.A., OzgunerF.and FollenG. J. 1995:Parallelizing existing applications in a distributed heterogeneous environment, 4th IEEE Heterogeneous Computing Workshop (HCW 1995), 93-100.

Jiang J, Han G and Zhu C. 2016: A Complicated Task Solution Scheme Based on Node Cooperation for Wireless Sensor Networks, . 22nd IEEE

International Conference on Parallel and Distributed Systems (ICPADS), 264-269

JinY., JinJ., GluhakA., MoessnerK., PalaniswamiM. 2012:An Intelligent Task Allocation Scheme for Multihop Wireless Networks, IEEE Transactions on Parallel and Distributed Systems, 23(3), 444-451.

Karalar T.C., Yamashita S., Sheets M. andRabaey J.2004:A low power localization architecture and system for wireless sensor networks, IEEE Workshop on Signal Processing Systems, USA: Signal Processing Society,89-94.

Miller M.J. and Vaidya N.H. 2005:AMAC protocol to reduce sensor network energy consumption using a wakeup radio, IEEE Trans. Mobile Comput., 4(3), 228-242.

Papataxiarhis V. 2016:Optimal Task Assignment in Sensor Networks, 17th IEEE International Conference on Mobile Data Management (MDM), 26-31.

Potdar V., Sharif A. and Chang E. 2009:Wireless Sensor Networks: A Survey, IEEE International Conference on Advanced Information Networking and Applications Workshops. USA: IEEE Computer Society, 636-641.

RinnerB. and Wolf W. 2008:An Introduction to Distributed Smart Cameras, Proceedings of the IEEE , 96(10), 1565-1575.

Sinnen O. 2007: Task Scheduling For Parallel Systems. New Jersey: John Wiley & Sons, Inc., Hoboken.

SHAMS R. andKHAN F. 2012:Solving Wireless Network Scheduling Problem by Genetic Algorithm, IAMURE International Journal of Mathematics Engineering & Technology, 4(), 63-70.

ShivleS., CastainR., Siegel H.J., MaciejewskiA.A., Banka T., ChindamK., DussingerS., PichumaniP., SatyasekaranP., Saylor W., Sendek D., Sousa

J., SridharanJ., SugavanamP. and VelazcoJ. 2004:Static mapping of subtasks in a heterogeneous ad hoc grid environment, In Proceedings of IEEE eighteenth International Parallel and Distributed Processing Symposium, 110.

Soe K.T. 2008: Increasing Lifetime of Target Tracking Wireless Sensor Networks, Proceedings of World Academy of Science, Engineering and Technology, 410-415.

Tian Y., JarupanB., EkiciE., and OzgunerF. 2006:Real Time Task Mapping and Scheduling for Collaborative In Network Processing in DVS-Enabled Wireless Sensor Networks, Proceedings in IEEE International Parallel and Distributed Processing Symposium 2006 (IPDPS 2006), 1-10.

Tian Y. and EkiciE. 2007:Cross-Layer Collaborative In Network Processing in Multihop Wireless Sensor Networks, IEEE Trans. Mobile Comput., 6(3), 297-310.

Tian Y., Ekici E. and Ozguner F. 2005:Energy-Constrained Task Mapping and Scheduling in Wireless Sensor Networks, Proceedings of IEEE International Workshop on Resource Provisioning and Management in Sensor Networks, 211-218.

Wang A. and ChandrakasanA. 2002:Energy-efficient DSPs for wireless sensor networks, IEEE Trans. Signal Process. Mag., 68-78.

Yang J., Zhang H., Ling Y., Pan C. andSun W. 2014:Task Allocation for Wireless Sensor Network Using Modified Binary Particle Swarm Optimization, IEEE Sensors Journal, 14(13), 882-892.

Yu Y. and PrasannaV.K. 2005:Energy-balanced task allocation for collaborative processing in wireless sensor networks, ACM Mobile Networks and Applications, 10(1-2), 115-131.