Proceedings of ICCTA2009

BIOLOGICAL TASK MAPPING AND SCHEDULING IN WIRELESS SENSOR NETWORKS

Yousef E. M. Hamouda, Chris Phillips

School of Electronic Engineering and Computer Science, Queen Mary University of London,
Mile End Road, London, E1 4NS, United Kingdom
yousef.hamouda@elec.qmul.ac.uk, chris.phillips@elec.qmul.ac.uk

Abstract

The increasingly complex roles for which Wireless Sensor Networks (WSNs) are being employed have driven the desire for autonomic self-organized capabilities for coordinated network operation. A Biological Task Mapping and Scheduling (BTMS) algorithm is presented in this paper to execute an application using a group of nodes. BTMS is inspired from biological behaviors of differentiation in zygote formation. Simulation results show that BTMS leads to improved network lifetime, energy consumption and service time compared with other commonly used algorithms.

Keywords: Wireless Sensor Networks, task mapping and scheduling, biological behaviors.

1 Introduction

Advances in Micro Electro-Mechanical Systems (MEMS), embedded microprocessors and wireless communications technologies have enabled implementation of large-scale Wireless Sensor Networks (WSNs) [1]. WSNs consist of small electronic nodes connected to each other via wireless communication protocols. Each node is equipped with embedded processors, sensor devices, storage, and radio transceivers. The nodes have limited resources in term of battery-supplied energy, processing capability, communication bandwidth, and storage. Some nodes may have more resources than others, which give them ability to enhance their communication and computing activities. WSNs have several attractive civil and military applications including healthcare, target tracking, monitoring, smart homes and surveillance. However, due to the unattended nature of sensor nodes, energy conservation and network lifetime is a crucial issue in WSNs [1].

Many WSNs applications such as target tracking and camera-based applications [2] require real time execution, nodes collaboration and intensive computation. Since an individual sensor node may not have enough processing power and sufficient battery life to perform complex functions to meet an application's requirements, the solution is to exploit cooperative behavior. One approach is to adopt the principle of swarm intelligence [3] where a group of nodes exchange information and update their characteristics based on influence from their peers. This paper considers a general model for high-level application by assuming that the application can be decomposed into smaller tasks with different computation weights and dependencies. Furthermore, node behavior is influenced to varying degrees by the target(s) and particular nodes, once they have been "elected".

Task mapping assigns resources to tasks and task scheduling is the execution sequence of the tasks to achieve or maximize performance objectives. It is well known that optimal task mapping is NP-complete problem [4]. Therefore, heuristic techniques are needed to obtain near-optimal solutions. In high performance computing [5], task mapping and scheduling are considered in depth. However, the design objectives in WSNs are different because of the limited resources in WSNs.

This paper proposed a novel framework, including a task mapping and scheduling algorithm called Biological Task Mapping and Scheduling (BTMS) in which an application is executed by a group of sensor nodes. The motivation behind BTMS is twofold. Firstly, concurrent processing in WSNs using BTMS decreases the prevalence of gaps in the network caused from dying nodes and consequently increases the network lifetime. Moreover, we introduce a decision maker in each node specifically to improve the lifetime of the network. Secondly, BTMS facilitates the timely completion of real time applications by exploiting the speed-up resulting from concurrent processing. The biological [6] aspect of BTMS is inspired from the biological zygote. When a zygote is formed, it comprises a collection of similar stem cells. Over time, the zygote cells start to specialize with different functionalities. This behavior is called differentiation [7]. The same principle is be applied in the proposed system; the network nodes start equally in a default state and then exhibit some kind of differentiation to perform certain tasks according to their resource availability and location

The paper is organized into six sections including this introduction. Section 2 reviews the related work. Section 3 defines the problem. The functional description and algorithms are presented in section 4. Section 5 provides the simulation results. Finally, Section VI summarizes the paper.

2 Related Work

A number of researchers have already considered task mapping and scheduling in WSNs. In [8], a fast online allocation algorithm is proposed to dynamically reconfigure WSN according to the node's activity changes so that the system utility is maximized. However, it does not consider the battery level as a part of sensor resources. BTMS algorithm explicitly involves the energy consumption as a design objective. In [9], six heuristic task mapping and scheduling techniques are compared and evaluated in heterogeneous ad hoc grids. However, unlike WSNs, [9] assumes individual channels for each node and ignores the energy consumption to transfer data. BTMS assumes a more realistic model for the WSN channel and considers the energy cost for all fundamental activities in the network. In [10], a heuristic algorithm has been developed to provide energy-balanced task allocation in a single-hop cluster of homogeneous sensor nodes. Dynamic Voltage Scaling (DVS) is equipped at each node. However, [10] assume the energy consumption to transmit a data item is the same in the sender and receiver, which is not realistic. Additionally, [10] does not exploit the broadcast nature of WSNs. BTMS again uses a more realistic model and differentiates between the energy cost at the sender and receiver. Finally, the authors in [11] propose Multi-hop Task Mapping and Scheduling (MTMS) algorithm for WSNs so that the scheduling length is minimized under the energy consumption constraint. Despite this, MTMS does not allow the mapping of a task to its immediate predecessors and does not order the tasks.

BTMS provides an application-independent task mapping and scheduling framework for WSNs with the following contributions different from previous work:

1. A heuristic technique for mapping or allocating and scheduling the application's tasks that have different resource requirements among nodes based on available resources and position so that the total energy consumption is minimized subject to completing an application's execution before its deadline.
2. To make nodes more intelligent and prolong the network lifetime, a decision maker in each node is introduced to decide if it can participate the collaborative activity or not.
3. Unlike other work, this paper introduces an autonomic distributed self-organizing network architecture. An election algorithm is used to select the group of nodes that cooperate and specify their function(s) within the group. The necessary message exchanges are described.
4. The proposed algorithms are simulated using realistic communication protocols, wireless channel and energy consumption models.
5. To our knowledge, this is the first research inspired from zygote biological behavior, where differentiation is related to distances to the target and among the cooperated nodes.

3 Problem Definition and Models

3.1 Application Model

A Direct Acyclic Graph (DAG) [10] is adopted to provide a general model for the application. The DAG $A=(V,E)$ consists of a set of vertices V representing the "n" tasks, $V=\{v_i : i=1,2,\ldots,n\}$, and a set of edges E representing the "e" communication dependencies, $E=\{\xi_k : i=1,2,\ldots,e\}$. The edge $\xi_k \in E$ between $v_i \& v_j \in V$ 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_j. As shown in Figure 1, the task without immediate predecessors is an entry-task while the task without immediate successors is an exit-task. The latency constraint of the application refers to the time to execute the application before its deadline, P.

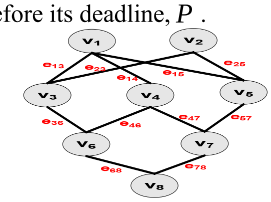

Figure 1 DAG Example

3.2 Energy Consumption Models

The energy consumption [12] required to transmit an l-bit message over a wireless distance d less than a threshold d_o is:

$$E_{TX}(l,d) = E_{elec}.l + \varepsilon_{amp}.l.d^2 \quad (1)$$

where E_{elec} is the electronic energy, and ε_{amp} is the amplifier energy. The energy consumption [12] to receive l-bit message is:

$$E_{RX}(l,d) = E_{elec}.l \quad (2)$$

Given a CPU with clock frequency f, the energy consumption to execute N clock cycles [12] is:

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

where, $f \approx K(\mathrm{V_{dd}} - c)$, V_T is the thermal voltage and C, I_0, n, K are CPU dependent parameters.

3.3 Problem Formalization

Assume, $S_{net} = \{s_i, i=1,2,\ldots,m\}$ are homogenous WSN. "f" is the CPU clock frequency. Assume during the network operation, a sensor node s_T

makes a request to its neighbors nodes asking to share in executing an application and the neighbors' that decide to participate in the execution are $S_n = \{s_i, i = 1,2,....,n_n\}$. The main goal of BTMS is to find a group of nodes " S_g^{opt} " so that the application DAG $A = (V, E)$ is optimally assigned to nodes based on the task requirements and node resources without violating the constraints.

The nodes that may participate the activity are $S_m = \{s_i : i = 1,2,...,n_m\} = S_n \cup \{s_T\}$ where $n_m = n_n + 1$. Define $S_g = \{S_g^Z : Z = 1,2,...\}$ as a set of all possible subsets, $S_g^Z \subset S_m$, where $S_g^Z = \{s_i : i = 1,2,...,n_g^Z\}$ and $n_g^Z \le n_m$. Find $S_g^{opt} \subset S_g$ where $S_g^{opt} = \{s_i : i = 1,2,..., n_g^{opt}\}$ and $n_g^{opt} \le n_g$ so that the total energy consumption, $energy\ (S_g^{opt})$, is minimized where:

$$S_g^{opt} = \arg\min_{S_g} energy\ (S_g) \quad (4)$$

$$energy(S_g) = \sum_{S_g} E_{comm} + \sum_{S_g} E_{comp} \quad (5)$$

and the collaborative execution time does not violate the time required to fulfill the activity, P:

$$CET \le P \quad (6)$$

4 Framework and Algorithms Used

In our previous paper [6], the system architecture and functional description is described in detail.

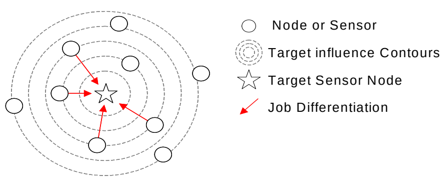

Figure 1 System Architecture

As shown in Figure 2, initially, all the wireless nodes are assumed to be equal. We also assume a potential target (e.g. target sensor node) can be regarded as a virtual chemical emitter that influences nodes in the affected area to a degree that is determined by their proximity to the target. When the target appears, nodes within its region of influence organize themselves into a group. This group cooperatively provides the required application and the network management functions according to the strength of the chemical and node available resources. There are three system operational phases, namely: group discovery, service provisioning and group management.

4.1 Group Discovery & Election Algorithm

The target sensor node broadcasts a request (REQ) message to its neighbors. The REQ takes the form:

$$REQ\{NID_{s_T}, E_{s_T}, x_{s_T}, y_{s_T}\}$$

where NID_{s_T} is the target node ID, E_{s_T} is its remaining energy and (x_{s_T}, y_{s_T}) is its position. By using the decision maker rules, each neighbor that receives the REQ message, decides whether it can participate this activity or not as follows:

If ($(E_{s_i} \succ E_{th})$ && (NotOnlyRelayNode)):
participate the activity;
else:
ignore the REQ message;

E_{th} is a predefined threshold energy. Therefore, if the energy level of a node under a threshold value, the node prefers to remain a relay node to prolong the network lifetime. In addition, if a node is located in scarce area, to improve the network connectivity and lifetime, the node will prefer to stay as relaying node. Conversely, if a node is identified as NotOnlyRelayNode it means that it has enough neighbors to relay the data in the network. Nodes that decide to participate the activity send to the target sensor a reply (REP) unicast message. REP message takes the form:

$$REP\{NID_{si}, E_{s_i}, x_{s_i}, y_{s_i}\}.$$

The nodes that received REQ and decide to participate in the activity will accept the REP message. After that, the target sensor performs a local election algorithm to determine the Main Node (MN) and Helper Node (HN) roles. The MN is the node that has the maximum fitness function that is defined as:

$$f_{s_i} = \alpha * \frac{E_{s_i}}{E_{\max}} + (1 - \alpha) * C_{s_i} \quad (7)$$

where $\alpha \in [0 \quad 1]$, E_{s_i} is the node remaining energy, $E_{\max}$ is its maximum energy and C_{s_i} is the its centrality which indicates how much it is in the group centre and is defined as follows:

$$C_{s_i} = \frac{radioRange}{\sum_{\forall j \in S_n} d_{ij}} \quad (8)$$

where S_n is the node neighbors and d_{ij} is the distance to the neighbor j. The target sensor sends the results to the MN through an election (ELEC) unicast message.

4.2 Service Provisioning & BTMS Algorithm

The MN performs the BTMS algorithm to distribute task functionalities among the group nodes based on election results. In $A = (V, E)$ each task $v_i \in V$ is a tuple of the form: $\{N_{v_i}, t_{v_i}, E_{v_i}\}$. N_i is the number of its computational cycles and $t_{v_i} = N_{v_i} / f$ is its execution time. E_{v_i} is the computational energy consumption to execute it and is calculated using equation (3). Each edge $\xi_k \in E$ referred as e_{ij} is a tuple of the form: $\{b_{e_{ij}}, t_{e_{ij}}, E_{e_{ij}}\}$. $b_{e_{ij}}$ is the data size to be transmitted between node (i) and (j). The time to transmit the $b_{e_{ij}}$ bits is the transition and propagation times, $t_{e_{ij}} = b_{e_{ij}} / R_s + d_{ij} / c$, where c is the light speed, and d_{ij} is the distance between sender and receiver. $E_{e_{ij}}$ is the communication energy consumption to transmit and receive e_{ij} and is calculated using equations (1) and (2). Each sensor node $s_i \in S_{net}$ is a tuple of the form: $\{ NID_{s_i}, E_{s_i}, x_{s_i}, y_{s_i}, E_{th} \}$. Figure 3 shows the BTMS algorithm.

1. Convert the DAG into level-based DAG;
2. Sort the task in each level in decreasing order;
3. Select $\beta \in [0 \quad 1]$;
4. For each task $v_i \in V$ from the lowest level do:
5. For each node $s_j \in S_m$ do:
6. Calculate $E_T(s_j, v_i)$;
7. Calculate $t_f(s_j, v_i)$;
8. Calculate $f(s_j, v_i)$;
9. End of inner for loop;
10. Map and schedule the task v_i to the node s_j that has minimum $f(s_j, v_i)$;
11. Update the nodes energy remaining;
12. Do not assign any more tasks to the nodes that has energy less than E_{th};
13. End of outer for loop;
14. Calculate the execution time (CET);
15. If ($CET > P$): choose different β;
16. End if;

Figure 3 BTMS Algorithm

In line (1), the level-based DAG is built so that the lowest level contains the entry tasks and the highest level contains the exit tasks. The immediate predecessors of the tasks in each level only belong to the upper levels [9]. In line (6), the total energy required to execute the task v_i is defined as:

$$E_T(s_j, v_i) = E_{v_i} + \sum_{\forall k} \left(E_{e_{ik}} + \varepsilon_s \right) \quad (9)$$

where, the summation term is the communication energy consumption to transmit and receive the edges of all the immediate predecessors (k) of v_i. In line (7), the time at which the task execution is finished is defined as:

$$t_f(s_j, v_i) = t_s(s_j, v_i) + t_{v_i} \quad (10)$$

where, $t_s(s_j, v_i)$ is the start execution time,

$$t_s(s_j, v_i) = \max\left\{ ava(s_j),\ \max_{\forall k} \left(ava(s_k) + t_{e_{ik}} + \tau_s \right) \right\} \quad (11)$$

and "ava" stands to the availability of the node (i.e. the time at which the node can execute the next event). τ_s and ε_s are the average startup time and energy consumption for the communication. They include the cost of collisions. In line (8), the fitness function is specified as:

$$f(s_j, v_i) = \left[\beta * \frac{t_f(s_j, v_i)}{P} \right] + \left[(1 - \beta) * \frac{E_T(s_j, v_i)}{E_{\max}} \right] \quad (12)$$

$\beta \in [0 \quad 1]$ is a design parameter which controls the weights of minimizing the total energy consumption and Collaborative Execution Time (CET) is the time taken for a group of nodes to complete a given activity. After the MN runs the BTMS algorithm, it sends the results to the group nodes through functionality (FUN) unicast message which contains the tasks assigned to the nodes, their types and immediate predecessors. The final result from the exit task is sent back to the target node through a result (RES) unicast message.

4.3 Group Management Phase

The MN sends management messages to control and communicate with the HNs. Management messages are used for such things as mobility management, target information handover, changing the MN and target movement. However, the management phase is outside the scope of this paper.

5 Simulation Results

The C++ programming language has been used to build a simulation model. In the model, 350 wireless sensor nodes are randomly deployed across an area of 1Km*1Km. We assume line of sight (LOS) communication between the nodes within the same coverage area. 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 150m. The Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA) protocol is used as the MAC layer protocol and the transmission speed is set to 1 Mb/s. As in [12], the parameters in equations (1) to (3) are set as follows: ε_{amp}=10pJ/b/m^2, V_T=26mV, C=0.67nF, I_o=1.196mA, n=21.26, K=239.28MHz/V c=o.5 and f=100MHz. The parameters in equations (7) to (11) are set as follows: α=β=0.5, τ_s=0.6528ms and ε_s=0.5uJ.

Algorithm	CET (ms)	Energy (mJ)
BTMS	25.496	2.304713
MTMS	44.2731	2.468315

Table 1 Results for Visual Surveillance DAG

The performance in terms of Collaborative Execution Time (CET) and energy consumption is evaluated for visual surveillance application DAG presented in [11] using BTMS and MTMS algorithms. The simulation is repeated twenty times for different visual image sizes and the average CET and energy consumption are tabulated in Table 1. As shown, BTMS can perform better than MTMS. In BTMS, tasks in each level are arranged in decreasing order. Therefore, the large tasks are mapped first. As shown in Figure 4, this feature leads to a decrease the CET because instead of executing large task after finishing the small ones, the large tasks will be executed concurrently with the execution of the small tasks.

In Figures 4 and 5, the simulations have been repeated 250 times using 250 different DAGs. Each DAG is created using the parameters: maximum successors = 3, number of entry tasks = 5, number of other tasks = 10 and number of exit tasks = 1. The deadlines are chosen so that they increase with increasing the computational load. Deadlines are selected to be less than the serial execution time, which is the time needed to execute the application in one node. Therefore, the deadlines increase with increasing the computational load. As shown in Figures 4 and 5, the CET and energy consumption increase with increasing the deadline and communication load. Deadline has to be large enough so that CET can meet it. This indicates that the computational load has to be large enough compared with communication load so that CET does not violate the deadline.

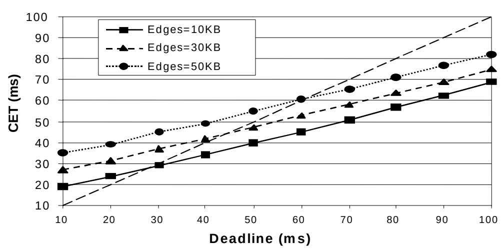

Figure 4 CET versus Deadline

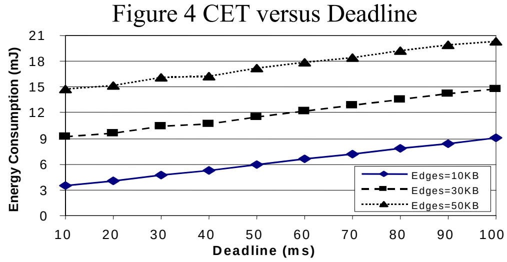

Figure 5 Consumed Energy versus Deadline

In Figure 6, the lifetime is the time at which the first node death occurs. The simulation is repeated twenty times for different network topologies and DAGs and the average lifetime performance ratio (LT_{BTMS} / LT_{MTMS}) is plotted with a 95% confidence interval. BTMS improves the lifetime relative to MTMS because it adopts the decision rules presented in Section 4. BTMS allows the mapping tasks onto nodes on which the predecessors are mapped. Therefore, the consumed energy is reduced.

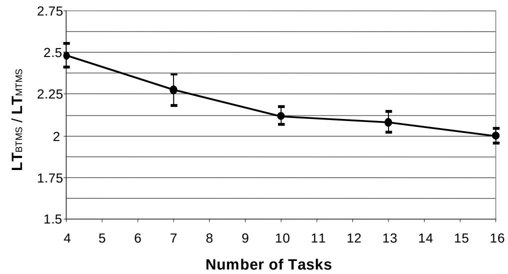

Figure 6 Lifetime Ratio versus Number of Tasks

6 Conclusion & Future Work

This paper introduces the BTMS algorithm in an autonomic self-organized WSN framework. The motivation is to reduce the energy consumption and meet application deadlines. Simulation results show that compared with MTMS, BTMS can improve the energy consumption, network lifetime and CET.

References

[1] D. E. D. Culler, M. Srivastava. Overview of sensor networks. *IEEE Computer,* 2004

[2] Rinner B., Wolf W. An Introduction to Distributed Smart Cameras. *Proceedings of the IEEE* , vol.96, no.10, pp.1565-1575, 2008

[3] Bicocchi N., Zambonelli F. Autonomic communication learns from nature. *Potentials IEEE,* vol.26, no.6, pp.42-46, 2007

[4] D. Fernandez-Baca. Allocating modules to processors in a distributed system. *IEEE Transaction on Software Engineering*, Vol. SE-15, No. 11, Nov. 1989, pp. 1427–1436

[5] T. Hagras, J. Janecek. A high performance, low complexity algorithm for compile-time job scheduling in homogeneous computing environments. *In Proc. Of International Conference on Parallel Processing Workshops* (ICPPW'03), pp. 149–155, 2003

[6] Y. E. M. Hamouda, C. Phillips. Biologically Inspired, Self Organizing Communication Networks. *PGNet2007*, 2008

[7] MM Student Projects, Site "Embryology Homepage", http://sprojects.mmi.mcgill.ca/ embryology/ (access at 27/04/2008)

[8] S. Giannecchini, M. Caccamo, C.-S. Shih, Collaborative resource allocation in wireless sensor networks. *In Proc. of Euromicro Conference on Real-Time Systems* (ECRTS'04), June/July 2004, pp. 35–44

[9] S. Shivle, R. Castain, et al, "Static mapping of subtasks in a heterogeneous ad hoc grid environment", *In Proc. of Parallel and Distributed Processing Symposium*, 2004

[10] Y. Yu, V. K. Prasanna. Energy-balanced task allocation for collaborative processing in wireless sensor networks. *Mobile Networks and Applications,* vol. 10, pp. 115–131, 2005

[11] Y. Tian, E. Ekici. Cross-Layer Collaborative In Network Processing in Multihop Wireless Sensor Networks. *IEEE TRANSACTIONS ON MOBILE COMPUTING*, VOL. 6, 2007

[12] A. Wang, A. Chandrakasan. Energy-efficient DSPs for wireless sensor networks. *IEEE Signal Processing Magazine*, July 2002.