Available online at www.sciencedirect.com
ScienceDirect
ELSEVIER
Procedia Computer Science 3 (2011) 484–490
Procedia Computer Science
www.elsevier.com/locate/procedia

WCIT 2010

Task allocation to actors in wireless sensor actor networks: an energy and time aware technique

Morteza Okhovvat[a], Mohsen Sharifi[a *], Hossein Momeni[a]

[a] Iran University of Science and Technology, School of Computer Engineering, Distributed Systems Laboratory,Tehran 16846-13114, Iran

Abstract

Task allocation is a critical issue in proper engineering of cooperative applications in embedded systems with latency and energy constraints, as in wireless sensor and actor networks (WSANs). Existing task allocation algorithms are mostly concerned with energy savings and ignore time constraints and thus increase the makespan of tasks in the network as well as the probability of malfunctioning of the network. In this paper we take both energy awareness and reduction of actor tasks' times to completion in WSANs into account and propose a two-phase task allocation technique based on Queuing theory. In the first phase, tasks are equally assigned to actors just to measure the capability of each actor to perform the assigned tasks. Tasks are then allocated to actors according to their measured capabilities in such a way to reduce the total completion times of all tasks in the network. The results of simulations on typical scenarios shows 45% improvement in the makespan of tasks in a network compared to the well-known opportunistic load balancing (OLB) task allocation algorithm that is generally used in distributed systems. It is shown that our algorithms provide better tradeoffs between load balancing and completion times of all tasks in a WSAN compared to OLB.

Selection and/or peer-review under responsibility of the Guest Editor.

Keywords: Wireless Sensor Actor Network; Task Allocation; Task Completion Time; Load Balancing; Queuing System;

1. Introduction

Recent advances in the technology of micro-electro-mechanical systems have greatly influenced the emergence of wireless sensor actor networks (WSANs) [1, 2]. WSANs consist of sensory nodes and actor nodes that are connected to each other via wireless links. In this paper we assume a semi-automated architecture (Figure 1) for WSANs, wherein sensor nodes gather environmental information and transmit them to the network sink (or base station) that figures out the proper actions to be taken by the actor nodes and assigns these actions (tasks) to appropriate actor nodes (Figure 2); hereafter in this paper we use the words sink and base station (BS) interchangeably.

WSANs are appropriate for quick reactions to environmental events. Since these networks are usually used in critical applications, delays can lead to disasters [2]. To make efficient use of WSANs capabilities, employing appropriate task allocation algorithms is indispensable. The proper mapping of tasks to actors can be guided by quality of service (QoS) parameters of a concerned application that is run on a given WSAN.

* Mohsen Sharifi. Tel.: + 98-21-7391-3307; Fax: + 98-21-7322-3307.
E-mail Address: msharifi@iust.ac.ir

1877-0509 © 2010 Published by Elsevier Ltd. Open access under CC BY-NC-ND license.
doi:10.1016/j.procs.2010.12.082

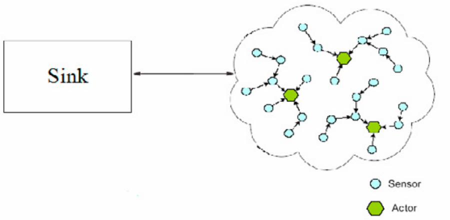

Fig. 1. A typical architecture of WSANs

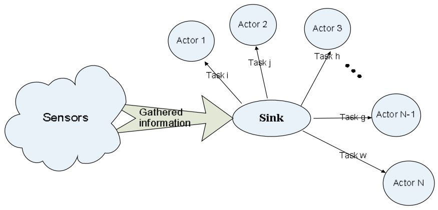

Fig. 2. Assumed WSAN topology

There are a comparatively large number of existing task scheduling algorithms for distributed systems in general that try to reduce the total task completion times of the system [4, 5, 6, 7, 8]. These are however not appropriate for WSAN task scheduling. A popular scheduling algorithm for distributed systems is the opportunistic load balancing (OLB) algorithm that assigns tasks in arbitrary order without considering the execution times of tasks [4]. In case of more than one available resource, it selects one resource arbitrarily. It tries to keep all resources busy as much as possible just to achieve its load balancing objective [5, 6, 7, 8]. The application of OLB to task scheduling in WSANs can lead to poor makespan [5] because it does not consider time constraints explicitly. It only implicitly considers load balancing, which covers energy and time parameters. Although consideration of load balancing alone may lead to reduction in task completion times, but in large scale distributed systems like WSANs, the load balancing objective alone can lead to poor makespan. There is thus a need for better task scheduling algorithms that consider both the load balancing objective and the reduction of task completion times into account in order to lower the makespan of all tasks in the network [10]. We thus present a two-phase technique with the above two objectives in mind for assignment of tasks to actors. The algorithms involved are modelled with queuing networks and simulated for validation.

The rest of paper is organized as follows. Section 2 presents notable related works. Section 3 describes our assumptions and the choice of a queuing model. Section 4 presents the queuing networks model of the proposed algorithms. Section 5 presents simulation results and Section 6 concludes the paper.

2. Related Work

Due to the challenging features and constraints of WSANs, such as resource and energy constraints and their dynamicity, most existing general-purpose scheduling algorithms are inapplicable to WSANs. There is however a rich set of scheduling algorithms for wireless sensor networks (WSNs) with the purpose of reducing task completion time and energy consumption. The applicability of both types of notable scheduling algorithms are discussed here.

M. Sharifi et al. [11] have presented a graph transformation-based approach to allocate tasks to sensor and actor nodes in support of real-time applications in WSANs. Each task is associated with proper graph transformation rules in such a way to guarantee that all existing tasks in the network complete before their deadlines expire. They have proved the correctness of their approach and reported 65% improvement in deadline hit ratio compared to the FIFO approach of task assignment. Although their approach tried to miss fewer deadlines but did not consider load balancing which can lead to reduction of network lifetime.

H. Park et al. [12] have proposed an energy-efficient task allocation framework for WSNs. They use graph descriptions to decompose tasks and then to assign them to appropriate sensor nodes. Their objective is to minimize energy consumption but time constraints are not considered in their approach.

A task allocation algorithm for gateways (cluster heads) within sensor networks have been proposed by M. Younis et al. [13]. Task scheduling on cluster heads is simplified by considering the processing time of collected data by at least one cycle. They try to maximize the network lifetime but do not consider the execution times of tasks.

Y. Yu et al. [14] have proposed an energy-balanced allocation for collective processing of tasks in WSNs. Tasks are allocated to sensor nodes, the voltage settings of tasks that allocate communication activities to channels is done, and eventually the scheduling of calculation and communication actions is performed. The broadcast nature of wireless communication was not however considered in their proposed communication scheduling model, which can adversely increase the time to completion of tasks.

Shivle et al. [15] have presented new task mapping and scheduling heuristics for mobile ad hoc networks. They assume that each node has a dedicated communication channel for simultaneous transmission and reception of data. This approach is inappropriate for WSANs wherein sensor nodes generally lack such capabilities.

An energy-constrained task mapping and scheduling (EcoMapS) has also been proposed by Y. Tian et al. [16]. Minimizing the schedule length of an application with energy consumption restriction has been the goal of this approach. Communications over multiple channels are modelled by extra linear restrictions of an Integer Linear Programming (ILP) problem through several single hop wireless channels and the problem is then solved accordingly [15,16]. Although energy spending is optimized in EcoMaps, but execution deadline of an application is not guaranteed.

Given this background on task allocation in sensor networks, in this paper we propose an energy and time aware technique for allocating tasks to actors in order to provide a suitable tradeoff between load balancing and completion times of all tasks in a WSAN.

3. Assumptions and Modeling Choice

We consider a WSAN wherein *m* actors A_j *(j = 1, . . ., m)* should perform *n* tasks T_i *(i = 1, . . . , n)*. Task scheduling amounts to the allocation of one or more time slots to one or more actors [17]. The total expected time taken by actor A_j with no load at the time of assignment to execute T_i is known as the execution time of task T_i by actor A_j (E_{ij}).

The expected completion time of a task *Ti* by an actor A_j (c_{ij}) is the time interval in which A_j completes *Ti* after finishing any remaining earlier assigned tasks. Makespan is known as a measure of throughput in heterogeneous systems as in WSANs [9, 11]. So, the scheduling problem in WSANs amounts to the allocation of a set of tasks to a set of actors based on declared QoS parameters such as makespan minimization. Consideration of all QoSs to get an optimized scheduling scheme is however impractical because this scheduling problem is an NP-complete problem [6, 8, 17, 18].

We further consider the following assumptions in our work:

- A semi-automated architecture for WSAN.
- Tasks are independent and sensors send their gathered information from environment to BS. BS figures out the proper actions to take and assigns each action (task) to an actor to be performed. This process follows a Poisson distribution.
- Considering there are *n* actors that each can perform its assigned tasks with a μ rate, and also assuming that the service rate of BS to figure out tasks is λ, relation Δ shows the condition which should hold for each actor *i*:

$$\frac{\lambda}{n} \pi \mu_i \qquad (\Delta)$$

- Tasks are non-preemptive and the formation rate of task*s* by BS is exponential.

Given these assumptions, we need to choose a proper queuing model to model our proposed task allocation technique. Simplicity and high capability of queuing models in performance and evaluation of reliability of computers and communication systems have lead to the wide deployment of queuing models in computers and communication systems. A single station queuing system comprises a finite or infinite number of queuing buffers and one or more identical servers [19]. We use the Kendall's notation to specify the distribution of the interarrival times, the number of servers, the distribution of service times, and system capacity. For example, M/M/1 notation means that the arrival process is Poisson, service times are exponentially distributed, and there is a single server. We use M/M/1 queues to model performance measures such as steady states, the probability of the number of jobs in the system, the average waiting time in the queue, the average spent time for each request, and the performance of the server.

Formula 1 calculates the average spent time of each request (w), wherein λ denotes the arrival rate of tasks to the queue and μ denotes the service rate; proof of formula can be found in [19].

$$w = \frac{1}{(\mu - \lambda)} \quad , \quad \lambda \pi \mu \qquad (1)$$

Formula 2 calculates the server utilization assuming there are no limitations on the number of tasks in the single server queue; utilization of a server denotes the server's busy times.

$$\lambda = (\rho * \mu) \Rightarrow \rho = \frac{\lambda}{\mu} \quad , \quad \lambda \; \pi \; \mu \qquad (2)$$

Queuing networks that include more than one service station are more appropriate for representing the structure of systems with a large number of resources, than models with single service station [19]. A queuing network in which tasks can come in or go out of the network is called *open* [19]. A queuing network is called *closed* if *task*s can neither come in nor go out of the network. An open queuing network is shown in Figure 3.

4. Task Allocation

Our proposed task allocation technique comprises of two algorithms. The main goal in both proposed algorithms is to reduce task completion time, while load balancing is also considered in the second algorithm. The inherent unpredictability of WSANs is considered in our proposed task allocation formulation using the queuing theory. BS and each actor have their own independent queue. Tasks are initially inserted in the queue of BS and later put in the queue of the actor chosen by BS to perform the task.

Given our assumptions in Section 3 and using queuing networks, Figure 4 shows a model of our assumed WSAN.

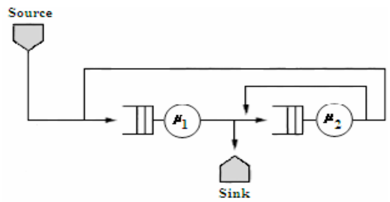

Fig. 3. A simple open queuing network [19]

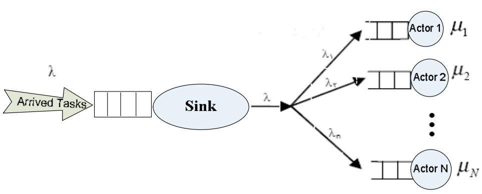

Fig. 4. A queuing networks model of WSAN

Following a Poisson process, BS inserts its configured tasks into its queue with λ rate. As BS is usually more rapid than the actors with fewer faults (or ideally with no faults at all), it works like a gate whose output rate is the same as its input rate λ. Based on splitting Poisson process [20], tasks are inserted into selected actors by BS with λ_i rate as is shown by Formula 3 for *n* actors.

$$\lambda = \sum_{i=1}^{n} \lambda_i \qquad (3)$$

Algorithm 1. As mentioned before, our task allocation technique comprises of two steps (levels). In the first phase, the arrival rate of all actors is assumed equal to $\frac{\lambda}{n}$. Calculating the spent time by the Little theorem, we get:

$$L = \lambda W \Rightarrow w = \frac{L}{\lambda} \qquad (I)$$

Relations 1 and I yield relation II :

$$L = \frac{\rho}{1 - \rho} \qquad (II)$$

Substituting L in Relation I with Relation II yields:

$$w = \frac{\rho / 1 - \rho}{\lambda} \qquad (III)$$

Using Relation II, Relation III, Relation IV and Relation V for calculation of the spent time (w), we get:

$$w = \frac{1}{\mu(1 - \rho)} = \frac{1}{\mu - \lambda} \qquad (IV)$$

$$w = \frac{1}{\mu - \lambda} \qquad (V)$$

The average tasks completion time can thus be calculated using Relation VI:

$$w_i = \sum_{i=1}^{n} w_i \qquad (VI)$$

As in the initial level, the arrival rate of all actors are equal but based on the capability of each actor, the rate of performing tasks (μ_i) are different. To achieve minimum tasks completion time (w_t), the arrival rate of tasks for each actor should be based on the capability of that actor. Hence, in the second level of algorithm, all λ_i and w_i pairs are sorted and based on Relation VII, a proper pair of λ_i and w_i is selected and therefore, *n-1* equations is generated. These equations can be solved using Formula 3, resulting in λ_i that represents the minimum task completion time. Because in this case further and bigger tasks are transmitted to the actors in which the capability of performing tasks is more than others.

$$\lambda_i * w_i = \lambda_j * w_j \qquad (VII)$$

Algorithm 2. The sole objective of reducing tasks completion time and ignoring in Algorithm 1, may lead to overloading of some actors and idling of some other actors, resulting in the partitioning of the WSAN. To avoid this problem, a second algorithm is proposed that considers load balancing too. Every λ_i is calculated by Algorithm 1, but the second level of Algorithm 1 uses Formula 2 and substitutes ρ_i instead of w_i. Then by using the average amount of λ_i that was derived in the first and second levels, the total tasks completion time is reduced and load balancing is taken care of too. In the other words, this algorithm makes a proper tradeoff between load balancing and reducing total tasks completion time and tries to optimize both load balancing and reducing total tasks completion time.

5. Experimental Results

The two proposed algorithms are compared with OLB algorithm through an example scenario. To have better evaluation, actors are chosen from three different categories with fast, medium and slow service rates. In the performed simulations, we assumed tasks are independent and sensors transmit their collected information from environment to BS and BS allocates tasks to appropriate actors.

We consider a WSAN with 5 actors with an arrival rate of tasks to BS equal to 50 units per second ($\lambda = 50$). The rates of execution of tasks by actors are $\mu_1 = 14$, $\mu_2 = 12$, $\mu_3 = 20$, $\mu_4 = 57$, $\mu_5 = 45$. The average tasks completion time in each actor, the workload of actors and the total tasks completion time are shown in Figures 5, 6 and 7, respectively. In these figures, A is the case of not using any of the algorithms, B is the case of using Algorithm 1, C is the case of using OLB and D shows the results of using Algorithm 2. Results show that Algorithm 1 yields the least total tasks completion time. They also show that the Algorithm 2 yields a poorer result than Algorithm 1 but works much better than OLB in reducing tasks completion time.

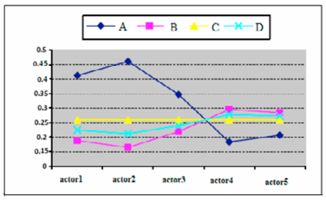

Fig. 5. Average tasks completion time in each actor

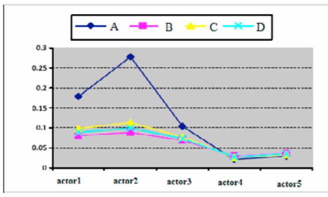

Fig. 6. Workload of each actor

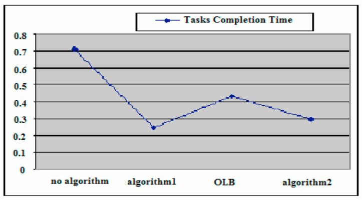

Fig. 7. Total tasks completion time under different algorithms

OLB gives the best load balancing compared to others while Algorithm 2 gives a better result than Algorithm 1. All in all, Algorithm 1 yields minimum total tasks completion time and OLB yields the best load balancing but Algorithm 2 yields reasonable results as a trade-off between both load balancing and minimizing tasks completion time objectives.

6. Conclusion

This paper proposed a technique comprising of two algorithms for allocating tasks to actors in WSANs, modelling the network with the queuing networks. The first algorithm only considered the reduction of tasks completion time as its objective. Balancing energy dissipation among all actor nodes and reducing total tasks completion time simultaneously was the dual objective of the second proposed algorithm. Experimental results showed that the first algorithm did minimize tasks completion time but was bad on load balancing. The well-known opportunistic load balancing (OLB) task allocation algorithm did best on load balancing but performed badly on tasks completion time. The second proposed algorithm yielded a reasonable tradeoff between both load balancing and minimizing total tasks completion time.

References

1. F. Xia, Y. C. Tian, Y. Li and Y. Sun, Wireless Sensor Actuator Network Design for Mobile Control Applications, Sensors 7 (2007), 2157-2173.
2. S. Hong, D. Kim and J. Kim, Battery-Aware Real-Time Task Scheduling in Wireless Sensor Networks, IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA'05), Hong Kong (2005).
3. H. Momeni, M. Sharifi and S. Sedighian, A New Approach to Task Allocation in Wireless Sensor Actor Networks, First International Conference on Computational Intelligence, Communication Systems and Networks (CICSyN), Indore, India (2009).
4. M. Maheswaran, S. Ali, H. J. Siegel, D. Hensgen and R. F. Freund, Dynamic Mapping of a Class of Independent Tasks onto Heterogeneous Computing Systems, Journal of Parallel and Distributed Computing 59 (1999), 107-121.
5. T. D. Braun, H. Jay Siegel, N. Beck, L. L. Boloni, M. Maheswaran, A. I. Reuther, J. P. Robertson, M. D. Theys and B. Yao, 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 (2001), 810-837.
6. R. Armstrong, D. Hensgen and T. Kidd, The Relative Performance of Various Mapping Algorithms is Independent of Sizable Variances in Run-time Predictions, Proceeding of 7th IEEE Heterogeneous Computing Workshop (1998), 79-87.
7. R. F. Freund, M. Gherrity, S. Ambrosius, M. Campbell, M. Halderman, D. Hensgen, E. Keith, T. Kidd, M. Kussow, J. D. Lima, F. Mirabile, L. Moore, B. Rust and H. J. Siegel, Scheduling Actors in Multi-user Heterogeneous Computing Environments with SmartNet, Proceeding of 7th IEEE Heterogeneous Computing Workshop (1998), 184-199.
8. R. F. Freund and H. J. Siegel, Heterogeneous Processing, IEEE Computing, (1993), 13-17.
9. M. Maheswaran, S. Ali, H. J. Siegel, D. Hensgen and R. F. Freund, Dynamic Mapping of a Class of Independent Tasks onto Heterogeneous Computing Systems, Journal of Parallel and Distributed Computing 59(2) (1999), 107-121.
10. T. Yamaguchi and Y. Takahashi, A Queue Management Algorithm for Fair Bandwidths Allocation, Journal of Computer Communications 30 (2007), 2048-2059.
11. H. Momeni, V. Rafe, M. Sharifi and A. T. Rahmani, A Graph Transformation-Based Approach to Task Allocation in Wireless Sensor Actor Networks, 21th IEEE International Conference on Software Engineering and Knowledge Engineering (SEKE09), USA(2009).
12. H. Park and M. B. Srivastava, Energy-Efficient Task Assignment Framework for Wireless Sensor Networks, Center for Embedded Networking Sensing (CENS), Technical Report (2003).

13. M. Younis, K. Akkaya and A. Kunjithapatham, Optimization of Task Allocation in a Cluster-based Sensor Network, Proceeding of 8th IEEE International Symposium on Computers and Communication (2003), 329-334.
14. Y. Yu and V. K. Prasanna, Energy-Balanced Task Allocation for Collaborative Processing in Wireless Sensor Networks, Mobile Networks and Applications 10 (2005), 115-131.
15. S. Shivle, R. Castain, H. J. Siegel, A. A. Maciejewski, T. Banka, K. Chindam, S. Dussinger, P. Pichumani, P. Satyasekaan, W. Saylor, D.Sendek, J. Sousa, J. Sridharan, P. Sugavanam and J. Velazco, Static Mapping of Subtasks in a Heterogeneous Ad-hoc Grid Environment, 18th Int'l Parallel and Distributed Processing Symposium (IPDPS 2004), USA (2004).
16. Y. Tian, E. Ekici and F. Ozguner, Energy-Constrained Task Mapping and Scheduling in Wireless Sensor Networks, Proceedings of IEEE International Conference on Mobile Ad-hoc and Sensor Systems, (2005), 8-16.
17. P. Brucker, Scheduling Algorithms, Fifth Edition, Springer, 2007.
18. O. H. Ibarra and C. E. Kim, Heuristic Algorithms for Scheduling Independent Tasks on Nonidentical Processors, Journal of the ACM 24(2) (1997), 280_289.
19. G. Bolch, S. Greiner, H. De Meer and K. Trivedi, Queuing Networks and Markov Chains, John Wiley &Sons Publication, Second Edition, 2006.
20. P. L. Burke, The Output of a Queuing System. Operation Research 4(6) (1956), 699-704.
21. M. Draief, J. Mairesse, and N. O'Connell, Joint Burke's Theorem and RSK Representation for a Queue and a Store, Discrete RandomWalks (DRW'03) (C. Banderier and C. Krattenthaler, eds.), vol. AC, Institut Henri Poincar´e, Paris, (2003), 69–82.