Comput Syst Sci & Eng (2019) 1: 5–12
© 2019 CRL Publishing Ltd

International Journal of
Computer Systems
Science & Engineering

A mathematical task dispatching model in wireless sensor actor networks*

Morteza Okhovvat and Mohammad Reza Kangavari†

School of Computer Engineering, Iran University of Science and Technology, Tehran, Iran
E-mail: morteza.okhovvat@gmail.com

In this paper, we propose a mathematical task dispatching model to reduce the total tasks completion time, i.e. make-span, in Wireless Sensor Actor Networks (WSANs). The proposed approach minimizes the completion time of tasks that have been allocated to actors but have not yet been dispatched to actors for execution in the networks. We calculate the best rate of dispatching of tasks by the network sink to allocated actors through a steady state analysis of our proposed model to solve equations and inequalities. It is shown that this dispatching rate improves the network lifetime too. Experimental results with a prototyped simulation of the proposed approach show shorter make-span and longer network lifetime compared to when one of the three famous task allocation algorithms, namely, the min-min, opportunistic load balancing (OLB), and stochastic allocation algorithms, is used.

Keywords: Make-span, task dispatching, wireless sensor actor network, network lifetime

1. INTRODUCTION

A group of wirelessly communicative sensor nodes and actor nodes that respectively collect environmental information and behave in reply to sensory information build up an individual kind of wireless network called Wireless Sensor Actor Networks (WSANs) [1]. The constituent parts of WSANs can be configured differently according to the requirements of applications and existing technologies. In this paper, we consider only WSANs with a semi-automated architecture [1] wherein all sensor nodes transmit their sensory information to a single node in the network named the network sink. This singleton node is more powerful than sensor nodes and actor nodes, and it is made responsible for getting sensory information and determining appropriate tasks (actions) to be done by actors.

One of the main challenges of WSANs is to efficiently use all its capabilities at its disposal to satisfy the quality as well as functional requirements of running applications. In WSANs with semi-automated architecture, this challenge can be partly resolved by the singleton sink node if it can select the most proper set of actors to perform tasks using quality parameters such as reliability, make-span, and completion time of tasks [3-5].

To make efficient use of the capabilities of WSANs, the sink must decide on the most appropriate group of actors to perform the tasks using quality parameters such as make-span, network lifetime, and reliability of services [5, 6]. Therefore, the sink must figure out an efficient dispatching rate to distribute tasks to the related actors considering the limited size queue (buffer) of each actor and the fact that an actor cannot receive any more tasks when its buffer is full. The challenge for the sink is thus to find a dispatching rate that minimizes the completion time of tasks that are waiting in the associated queues of actors to be executed by actors.

Min-min, opportunistic load balancing (OLB), and stochastic allocation are three popular examples of task allocation algorithms that are generally used in distributed systems like WSANs [6]. Load balancing is the main goal of OLB achieved by keeping all actors as busy as possible [6]. This algorithm schedules the tasks based on minimum estimated completion time of tasks in arbitrary order [3]. The min-min algorithm considers the ap-

*The brief version of this paper has been presented in the 105th International Conference on Science, Engineering & Technology (ICSET 2017), Hamburg, Germany, December 22-23, 2017.

†Corresponding author. E-mail: kangavari@iust.ac.ir

proximate execution and completion time of all tasks on each actor and only then repetitively assigns a task with lowest completion time to an actor with minimum execution time [6, 7]. The stochastic allocation algorithm is very simple and allocates tasks to available resources (actors) stochastically. This algorithm does not consider any binding such as execution time of tasks and current situation of resources (idle/busy). The main advantage of this algorithm is its simplicity and can be implemented very fast.

However, existing WSAN scheduling algorithms suppose unbounded queues that are reasonably unrealistic. In this paper, we consider the limitation on the size of queues and yet attempt to reduce the completion time of allocated tasks to each actor that are not performed by the actor in such a way that the make-span is minimized. To attain this goal, it is necessary for the sink to get an approximation of the ability of each actor to calculate an appropriate dispatching rate for that actor and guarantee that the demanded quality parameters of running application are met.

The rest of paper is organized as follows: Notable related works are presented in section 2. Section 3 described our assumptions. In section 4, our proposed approach is presented. Section 5 presents the simulation results, and section 6 concludes the paper and presents future works.

2. RELATED WORKS

M. Sharifi *et al.* [3] have presented an energy and time aware approach to assign tasks to actors in WSANs. They figure out the capability of actors to perform tasks and use this information to assign tasks to actors in such a way to lessen the make-span of tasks. They reported 45% improvement in the network make-span compared to when they use the OLB algorithm. Their approach provides a suitable tradeoff between completion times of all tasks and a balanced load on actors, but it ignores the limitation on the size of actors' buffers.

Farias *et al.* [7] proposed a task scheduling algorithm for WSANs to improve the energy efficiency and so, increasing the network lifetime. To reach this goal, their algorithm tries to utilize the characteristics of applications with common tasks and avoid repeating tasks unnecessarily. However, their approach can increase the total remaining energies of actuators, but neither make-span nor reliability of services has been considered by their algorithm.

Shu *et al.* [8] presented an energy aware scheduling algorithm to maximize the network lifetime while making strict sensing guarantees in the WSN. In order to verify their algorithm, they performed an in-depth evaluation of its performance via large-scale simulations and reported an average of 39.2% improvement of network lifetime over the baseline method. The main drawback of their algorithm is that neither the reliability of services nor the execution deadline for applications was considered in their work.

Okhovvat *et al.* [9] have proposed a starvation free, time and energy aware scheduling algorithm called Scate. This algorithm allows concurrent executions of any mix of small and large tasks and yet prevents probable starvation of tasks. Reducing the total completion time of tasks and increasing the residual energies of actors simultaneously was the dual objective of Scate. The main drawback of their algorithm is that it does not guarantee the execution deadline for applications.

Momeni *et al.* [11] have proposed a mathematical approach to reduce average number of waiting tasks in WSANs. They calculate the best rate of dispatching of tasks by the network sink to allocated actors through a steady state analysis and showed that their approach reduces the mean number of waiting tasks. In their approach reducing the make-span did not consider explicitly, but they believe that their approach may reduce the total tasks completion time too.

Byun and So [12] have proposed an epidemic-inspired algorithm for data dissemination in WSANs that considers the delay requirements and try to decrease energy consumption. They used a mathematical analysis to predict and support the demanded performance of an application. Their approach controls the infectivity rate that results in an adaptive number of active/sleep nodes. They asserted that their approach can reduce the energy consumption while achieving application delay requirements.

Given this background on task allocation, in this paper we present a mathematical model using queuing theory to reduce the mean number of allocated tasks awaiting execution by actors in WSANs.

3. ASSUMPTIONS

We have considered a semi-automated WSAN with a single network sink and m actors A_j $(j = 1, \ldots, m)$ that should perform n tasks $T_i (i = 1, ..., n)$. In such a network, a schedule for each task is an allocation of one or more time slots to one or more actors [13]. This scheduling problem is known as an *NP-complete* problem [14-17]. In this paper, the aim of our approach is defined to decrease the completion time of tasks allocated to each actor in order to minimize the make-span. This goal is achieved by the calculating of capability of each actor at the time of assignment of tasks such as its current task load, and its speed in executing tasks.

We have further assumed that tasks are independent and sensors transmit their gathered data from physical environment to the sink. The sink figures out the appropriate actions (tasks) and then dispatches them to actors to be performed. Tasks are non-preemptive and their generation process follows a Poisson distribution.

4. PROPOSED APPROACH

We compute the make-span as the sum of the completion time of allocated tasks to actors. Each of the actors is modeled by a $M/M/1/K$ queuing system [18, 19] wherein tasks arrive at actor A_i with λ_i rate and are executed with μ_i rate. Figure 1, depicts the typical model of such a network. To reduce the make-span, we should adjust the dispatching rate of tasks to actor properly. As we assumed that the queue of each actor has limited capacity K, system will reach to a steady state and hence, there is no need to consider the relation $\lambda < \mu$ that has to be considered if the queues had assumed to be boundless.

In our proposed model, tasks are generated by the sink based on the received sensory information and then are assigned to appropriate actors. Appropriate actors are the actors that can finish tasks sooner and hence, minimize the completion time of tasks. These actors are determined by the sink in the proposed approach.

It is assumed that the generation rate of tasks (λ) follows a Poisson process and based on the splitting Poisson distribution [20], allocated actors receive the tasks with λ_i rate. This shown by relation (4.1) for n actors:

$$\lambda = \sum_{i=1}^{n} \lambda_i \tag{4.1}$$

Since the main objective of the presented approach is to minimize the completion time of tasks that should be done by the actors, the dispatching rate of tasks to each actor have to be estimated appropriately. In fact, our approach aims to find the best dispatching rate λ_i $(i = 1ton)$, to minimize the completion time of tasks waiting to be executed by the actors. Thus, each actor is modeled as an $M/M/1/k$ queue wherein the interval time between the allocations of two consecutive tasks and also the service times is an exponential process. Figure 2, shows the continuous time Markov chain (CTMC) model of actor A_i as a $M/M/1/K$ queue. In Figure 2, each ellipse denotes a state of actor A_i, and the number inside of each ellipse shows the number of existing tasks in the queue of actor A_i.

To get a steady state analysis of CTMC shown in Figure 2, we use following relations wherein π_i denotes the steady state probability of existing tasks in state i. In these relations, λ_i denotes the rate of arrival tasks at state i, and μ_i is the service rate of actor A_i. Table 1 shows the notations we have used in defining the relations (4.2)-(4.19).

$$\begin{aligned}
\lambda_i \cdot \pi_0 &= \mu_i \cdot \pi_1 \\
\pi_1(\lambda_i + \mu_i) &= \lambda_i \cdot \pi_0 + \mu_i \cdot \pi_2 \\
\pi_2(\lambda_i + \mu_i) &= \lambda_i \cdot \pi_1 + \mu_i \cdot \pi_3 \\
&\vdots \\
\lambda_i \cdot \pi_{k-1} &= \mu_i \cdot \pi_k
\end{aligned} \tag{4.2}$$

As shown by (4.3), the total probability is always equal to 1 and hence, π_o can be computed by (4.4):

$$\sum_{i=0}^{n} \pi_i = 1$$

$$\pi_0 = \frac{1}{\sum_{n=0}^{k} \left(\frac{\lambda_i}{\mu_i}\right)^n} \tag{4.3}$$

Since each π_n is a function of π_0, every π_n is greater than zero if and only if π_0 is bigger than zero. According to (4.5), π_0 and then all π_n are positive.

$$\left[\sum_{n=0}^{n} \left(\frac{\lambda_i}{\mu_i}\right)^n\right] \prec \alpha \forall \lambda_i, \mu_i \tag{4.4}$$

α: Constant

The probability of steady state π_n for each state of actor A_i is calculated by (4.6) using (4.2) and (4.3).

$$\pi_n = \left(\frac{\lambda_i}{\mu_i}\right)^n \times \pi_0 \tag{4.5}$$

Using (4.4) and (4.6), we deduce (4.7):

$$\pi_n = \frac{\left(\frac{\lambda_i}{\mu_i}\right)^n}{\sum_{n=0}^{k} \left(\frac{\lambda_i}{\mu_i}\right)^n} \tag{4.6}$$

Since it has been assumed that each actor process and carry out the tasks consecutively, if T tasks are in the buffer of an actor, $T - 1$ tasks are waiting. We have considered that the queue of each actor has a limited capacity k and hence, it makes the system comes to a steady state. We can therefore calculate the number of tasks in the queue of actor A_i by (4.8) in which L_i is the number of assigned tasks to the actor A_i, and L_{Qi} is the number of waiting tasks in the queue of that actor.

$$L_{Qi} = L_i + (\pi_0 - 1) \tag{4.7}$$

To compute the spent time of tasks, we used the Little theorem [21] and therefore, we get (4.9) in which W denotes spent time of tasks in an actor, L denotes the queue size of that actor, and λ denotes the arrival rate of tasks to that actor.

$$L = W \cdot \lambda \rightarrow W = L/\lambda \tag{4.8}$$

Relations (4.8) and (4.9) result in relation (4.10):

$$W_{Qi} = W_i + (\pi_0 - 1) \tag{4.9}$$

To figure out W_{Qi}, W_i should be calculated. To do this, both equality and inequality of λ_i and μ_i are studied. In the case of inequality, (4.11) can calculate W_i.

$$W_i = \frac{1}{\lambda_i} \sum_{n=0}^{k} n_i . \pi_{ni} = \frac{1}{\lambda_i} \sum_{n=1}^{k} \left(n . \left(\frac{\lambda_i}{\mu_i}\right)^n\right) \times \left(\frac{1 - \frac{\lambda_i}{\mu_i}}{1 - (\frac{\lambda_i}{\mu_i})^{k+1}}\right) \tag{4.10}$$

We can derive (4.12) and (4.13) from (4.11):

$$W_i = (\frac{\lambda_i}{\mu_i}) . \sum_{n=1}^{k} \left(\frac{d(\frac{\lambda_i}{\mu_i})^n}{d(\frac{\lambda_i}{\mu_i})}\right) . (\frac{1}{\lambda_i}) . \left(\frac{1 - (\frac{\lambda_i}{\mu_i})}{1 - (\frac{\lambda_i}{\mu_i})^{k+1}}\right) \tag{4.11}$$

$$W_i = \left(\frac{1 - (\frac{\lambda_i}{\mu_i})}{1 - (\frac{\lambda_i}{\mu_i})^{k+1}}\right) . (\frac{1}{\mu_i}) . \frac{d\left(\frac{1-(\frac{\lambda_i}{\mu_i})^{k+1}}{1-(\frac{\lambda_i}{\mu_i})}\right)}{d(\frac{\lambda_i}{\mu_i})} \tag{4.12}$$

Simplification of (4.13) results in (4.14).

$$\begin{aligned}
W_i = &\left(\frac{1 - (\frac{\lambda_i}{\mu_i})}{1 - (\frac{\lambda_i}{\mu_i})^{k+1}}\right) . (\frac{1}{\mu_i}) \\
&\left(\begin{array}{c} \left(\frac{-(k+1).(\frac{\lambda_i}{\mu_i})^k}{1-(\frac{\lambda_i}{\mu_i})}\right) + \\ \left(1 - (\frac{\lambda_i}{\mu_i})^{k+1}\right) . \left(1 - (\frac{\lambda_i}{\mu_i})^{-2}\right) \end{array} \right)
\end{aligned} \tag{4.13}$$

Finally, (4.15) can be derived from (4.11), (4.12), (4.13), and (4.14). We use (4.15) to determine the spent of allocated tasks to actorA_i.

$$W_i = \frac{1}{\lambda_i} \times \left[\left(\frac{(\frac{\lambda_i}{\mu_i})}{1 - (\frac{\lambda_i}{\mu_i})}\right) - \left(\frac{(k+1).(\frac{\lambda_i}{\mu_i})^{k+1}}{1 - (\frac{\lambda_i}{\mu_i})^{k+1}}\right)\right] \tag{4.14}$$

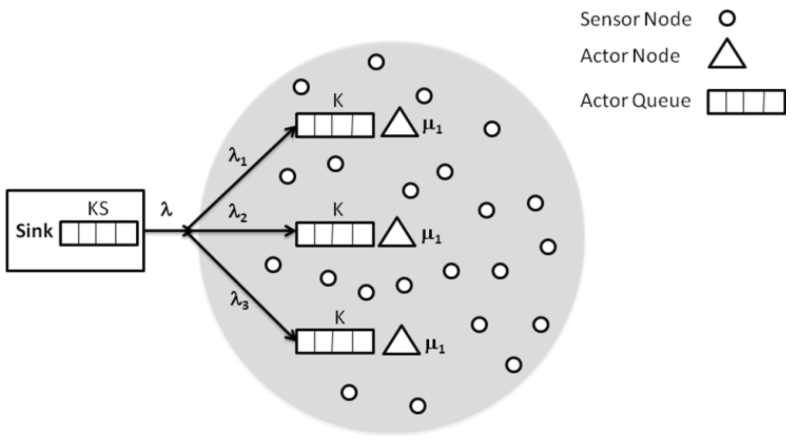

Figure 1 A queuing network model of WSAN.

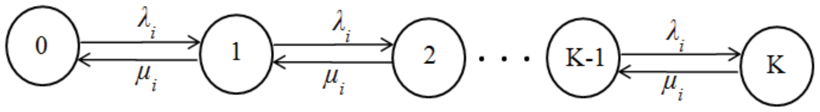

Figure 2 CTMC for an actor A_i.

Table 1 Notations used in the relations.

Term	Definition	Term	Definition
N	Number of tasks	L_i	Number of tasks in the actor A_i
K	Size of queue of each actor	L_{Qi}	Number of waiting tasks in the queue of actor i
λ_i	Arrival rate of tasks to actor i	W_i	The time that actor i finishes its assigned tasks
π_i	Steady state probability of existing tasks in state i	W_{Qi}	waiting time of tasks in actor i.
μ_i	Service rate of actor i	W_{QTotal}	Total waiting time of all tasks
A	A constant number that is greater than zero	m	Number of actors

In the case that λ_i and μ_i are equal, (4.16) gives the number of tasks allocated to A_i. In the other words, if the arrival rate of tasks to an actor is the same as the service rate by that actor the relation (4.16) can be used.

$$W_i = (\frac{1}{\lambda_i}).\sum_{n=0}^{k} n.(\frac{\lambda_n}{\mu_n})_i = \frac{k}{2\lambda_i} \tag{4.15}$$

After solving and simplifying (4.11), (4.12), (4.13) (4.14), (4.15) and (4.16), we derive (4.17) that calculates the mean waiting time of tasks in actor A_i.

$$W_{Qi} = \begin{cases} \frac{1}{\lambda_i} \times \left[\begin{array}{l} \left(\frac{(\frac{\lambda_i}{\mu_i})}{1-(\frac{\lambda_i}{\mu_i})} \right) - \left(\frac{(k+1).(\frac{\lambda_i}{\mu_i})^{k+1}}{1-(\frac{\lambda_i}{\mu_i})^{k+1}} \right) \\ -\left(1 - \frac{1-(\frac{\lambda_i}{\mu_i})}{1-(\frac{\lambda_i}{\mu_i})^{k+1}} \right) \end{array} \right] \\ \quad \text{if } (\lambda \neq \mu) \\ \\ \frac{1}{\lambda_i} \times \left(\frac{k}{2} - \left(1 - \frac{1}{k+1} \right) \right) \\ \quad \text{if } (\lambda = \mu) \end{cases} \tag{4.16}$$

To calculate the total completion time of tasks that should be performed by actors in the WSAN, we apply (4.17) to compute the completion time of tasks in each actor. The total completion time of tasks that should be accomplished by actors is thus given by (4.18):

$$W_{QTotal} = \sum_{i=1}^{m} W_{Qi} \tag{4.17}$$

We can finally use (4.17), (4.18) and (4.19) to formulate the main goal of this paper, which is to minimize the overall completion times of all tasks in the network, i.e. make-span, presuming that the capacity of the all actors are the same and equal to k.

The goal will be as (4.19) where k, λ_i, μ_i are constants, $0 < \mu_i$, $0 \leq \lambda_i$ and $\forall i \,:\, 0 < i \leq m$ where m is the total number of actors. It should be noted that if λ_i comes to zero, the actor A_i will be unavailable and the relation (4.19) is be applied to available actors.

$$Minimizing \sum_{i=1}^{m} \begin{cases} \frac{1}{\lambda_i} \times \left[\left(\frac{(\frac{\lambda_i}{\mu_i})}{1-(\frac{\lambda_i}{\mu_i})} \right) + \left(\frac{(1-k).(\frac{\lambda_i}{\mu_i})^{k+1}}{1-(\frac{\lambda_i}{\mu_i})^{k+1}} \right) - (\frac{\lambda_i}{\mu_i}) \right] \\ \quad if\ (\lambda \neq \mu) \\ \frac{1}{\lambda_i} \times \left(\frac{k}{2} - (\frac{\lambda_i}{\mu_i}) \right) \\ \quad \text{if } (\lambda = \mu) \end{cases} \tag{4.18}$$

5. EXPERIMENTAL RESULTS

To show the efficiency of our approach we conducted our simulations using MATLAB [22] in a typical scenario. We evaluated the proposed approach in compare with three well-known task allocation algorithms, namely, the min-min, OLB, and stochastic allocation algorithms in terms of total completion time of tasks and lifetime of actors. In addition, to study the effect of scale on the efficiency of our approach, we performed simulations in both large and small scales in two different settings. In the small scale, we assumed a 2D space, square field, 10m×10m, containing 100 sensor nodes with 1 meter transmission range, and 7 actor nodes. We have assumed that the tasks to be executed by actors were independent and that actors could browse the whole network with no restrictions on routing hops. The primary energy of each actor is assumed to be the same as others and equal to 25 Joules. The bandwidth of nodes is assumed to be 250 *Kb/s*.

In the large scale, we assumed a 2D space, square field, 100m×100m containing 10000 sensors with 1 meter transmission range, and 20 actor nodes. The primary energy of each actor was assumed to be the same as others and equal to 25 Joules. The bandwidth of nodes is assumed to be 250 *Kb/s*.

As the network sink is usually faster than actors and it has fewer faults (or ideally has no faults at all), we have assumed that the queue of the sink never overloads. The size of queue of each actor was assumed to be 10 and to simplify we assumed that the sizes of all tasks were the same. To have a better evaluation, actors are chosen from three different categories with fast, medium and slow service rates. We further assumed that each actor runs only a single task at any time and tasks were independent and sensors transmitted their collected information from environment to the sink and the sink allocated tasks to each actor with proper rate.

It is important to note that using the proposed approach in larger scales with more sensor/actor nodes will lead to similar results. In fact, our choice of semi-automated architecture for WSANs does not confine the applicability of the proposed approach to real large-scale WSANs. As stated by Liu and others [23], some large-scale WSANs may be single-hop in terms of wireless communication for transmitting information. A sink can be mobile and get close to sensors so that transmission of data could be done in a single hop. In other examples, embedded sensors may move toward a stationary sink. For example, sensors can be embedded into actors to trace their locations over time. When the actor approaches a fixed sink, collected information can be transmitted.

Figures 3 and 4 show the make-span of the network under four task allocation approach in both small scale and large scale settings, respectively. As Figure 3 shows, in the small scale wherein the required time to transmit data between sink and actors is not much compared to execution time of tasks on the actors, the min-min algorithm results in less make-span compared to OLB and stochastic allocation while the proposed approach results in the best make-span.

As Figure 4 depicts, in the large scale wherein the required time for communication between sink and actors is considerable in compare with the execution time of tasks on the actors, the proposed approach results in the shortest make-span compared with other approaches; thereafter, the OLB results in shorter make-span than min-min and stochastic allocation algorithms. However, the proposed approach results in the best make-span in both small scale and large scale while stochastic allocation algorithm results in the worst make-span in the both scales.

Since WSANs are mostly deployed in harsh environments, it is very difficult and actually impossible to recharge the actor nodes. Therefore, the lack of energy of an actor means the death of that actor. Hence, we also compared our approach with the three mentioned approaches in terms of network lifetime. The network lifetime is evaluated in terms of all of the actor nodes alive (ANA) and half of the actor nodes alive (HNA) for the sake of clarity.

As shown in Figure 5, in small scale and in terms of ANA, our approach and the min-min algorithm have nearly result in the same result, which is better than the lifetime resulting from other algorithms. But in large scale, our approach improves the lifetime (ANA) considerably better than three other algorithms; thereafter, the OLB results in better lifetime than the min-min and stochastic allocation algorithms.

Figure 6 depicts the network lifetime in terms of HNA in both small and large scales. As shown in Figure 6, in small scale, our proposed approach results in bigger HNA compared with the three mentioned algorithms; thereafter, the min-min results in better HNA than OLB and stochastic allocation algorithms while stochastic allocation results in the worst HNA. In large scale, the min-min algorithm results in less HNA than OLB, but it still operate better than stochastic allocation algorithms. However, the results demonstrate that our approach results in the best HNA among the four mentioned approaches.

In the stochastic allocation approach, tasks are assigned to actors arbitrary and hence, several tasks may be assigned to some actors while other actors are idle. This will unbalance the energy consumption of the actors and will lead to network partitioning. Therefore, as we expected, the stochastic allocation algorithm results in the worst ANA and HNA in compare with the three

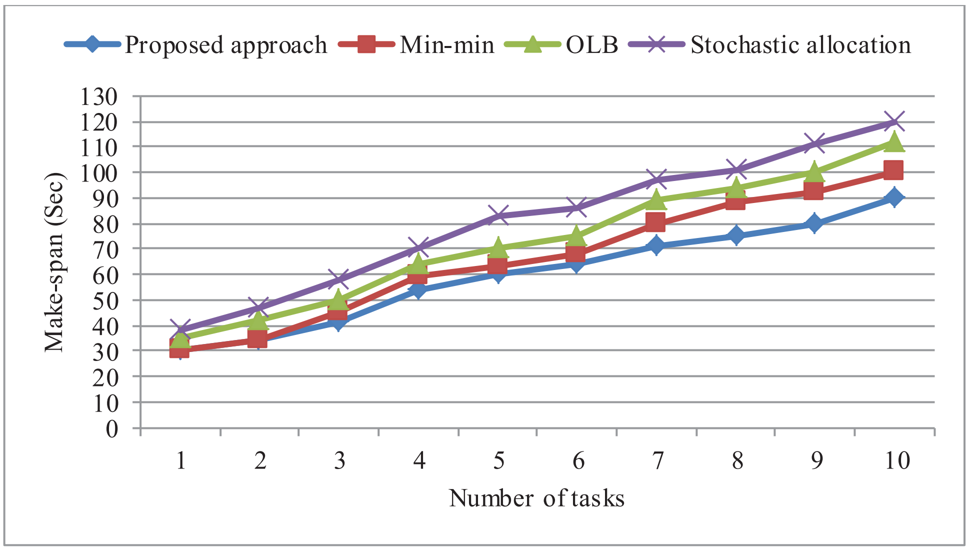

Figure 3 Make-span in small scale network.

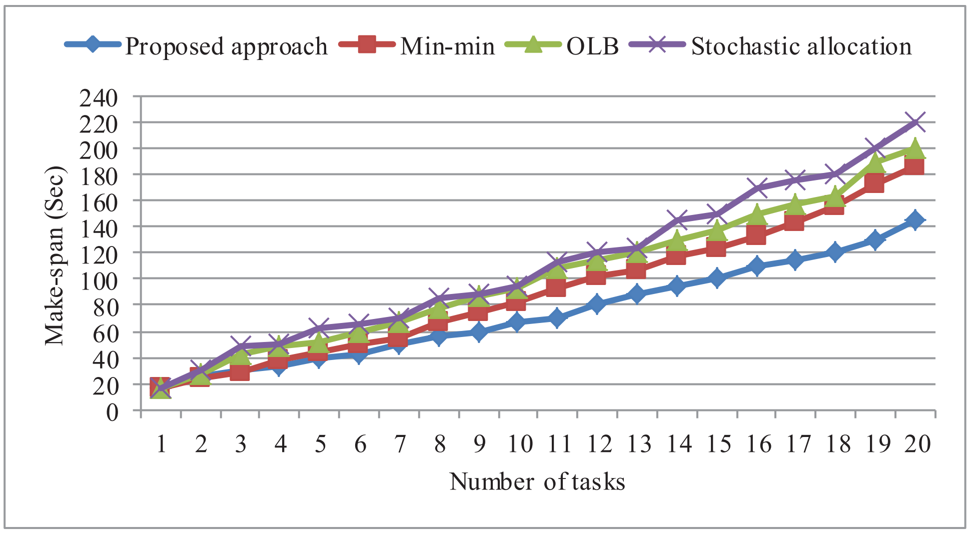

Figure 4 Make-span in large scale network.

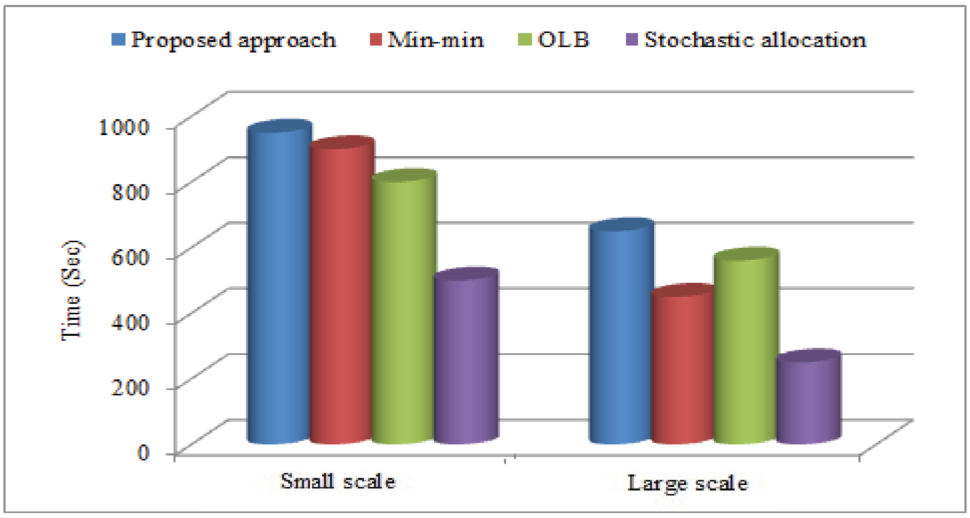

Figure 5 Experimental results in terms of ANA.

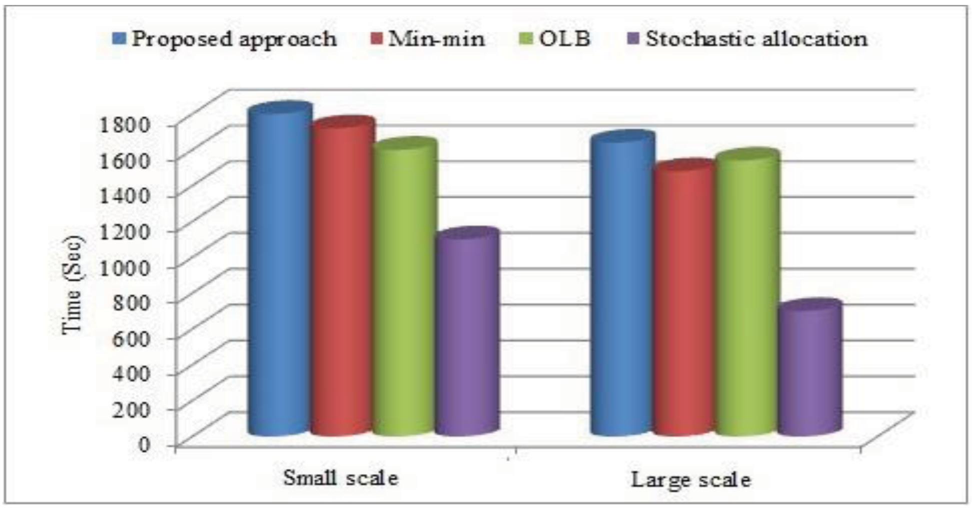

Figure 6 Experimental results in terms of HNA.

other approaches. But the proposed approach assigns tasks to actors based on their capability to minimize the make-span and implicitly improves the balanced loads on the actors. Hence, as we expected, experimental results illustrated that our proposed approach can improve the network life time in both small and large scales better than the min-min, OLB and stochastic allocation algorithms.

All in all, our proposed approach results in the best network lifetime in both small scale and large scale in compare with the min-min, OLB and stochastic allocation algorithms, while stochastic allocation results in the worst network lifetime.

6. CONCLUSION AND FUTURE WORKS

We proposed a mathematical technique to reduce the make-span of WSANs in which queuing theory was used to model the network. Solving the equations and inequalities, the best dispatching rate of tasks are calculated by the sink to allocated actors through a steady state analysis of the proposed model. Hence, using the arrival rate of tasks to each actor, the make-span is minimized. To evaluate the proposed approach, it compared to three well-known task allocation algorithms, namely, the min-min, OLB, and stochastic allocation algorithms in terms of make-span and lifetime of actors. Results of experiments on typical scenarios in small and large scales showed that our approach is the best in compare with the three mentioned algorithms in terms of reducing make-span. Our proposed approach also enhanced the network lifetime, which was evaluated in terms of ANA and HNA, considerably better than the min-min, OLB, and stochastic allocation algorithms. Consideration of task deadlines in support of real-time applications, reliability of services, fault-tolerance of actors and other QoS parameters are the open issues of the proposed approach.

REFERENCES

1. Zhou J, Jiang H, Wu J, Wu L, Zhu C, Li W. SDN-Based Application Framework for Wireless Sensor and Actor Networks. IEEE Access, 2016; 4: 1583-1594.
2. Liu Q, Yang S, Zhang L, Ji L. Game-theory-based coordination in wireless sensor and actor networks. IET Wireless Sensor Systems, 2015; 6: 166-172.
3. Okhovvat M, Sharifi M, Momeni H. Task Allocation to Actors in Wireless Sensor Actor Networks: An Energy and Time Aware Technique. J Procedia Computer Science, 2011; 3: 484-490.
4. Braun TD, Jay Siegel J, Beck N, Boloni LL, Maheswaran M, Reuther AI, Robertson JP, Theys MD, Yao B. A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems. J Parallel and Distributed Computing, 2001; 61: 810-837.
5. Lai ZH, Yang J, Shan H. (2003) A Resource Allocation Method in the Neural Computation Platform. In: Grid and Cooperative Computing; 7-10 December 2003; Shanghai, China: pp. 166-169.
6. Okhovvat M, Kangavari MR. TSLBS: A time-sensitive and load balanced scheduling approach to wireless sensor actor networks. Comput Syst Sci & Eng, 2019; 34:1–9.
7. Farias CM. de, Pirmez L, Delicato FC, Li W, Zomaya A.Y, Souza JN de. A Scheduling Algorithm for Shared Sensor and Actuator Networks. In: IEEE 2013 Information Networking Conference (ICOIN); 28-30 January 2013; Bangkok, Thailand: IEEE. pp. 648-653.
8. Shu Y, Shin KG, Chen J, Sun Y. Joint Energy Replenishment and Operation Scheduling in Wireless Rechargeable Sensor Networks. IEEE Trans. Industry Informatics 2017; 13: 125-133.
9. Sharifi M, Okhovvat M. Scate: A Scalable Time and Energy Aware Actor Task Allocation Algorithm in Wireless Sensor and Actor Networks. ETRI J 2012; 34: 181-192.
10. Shivle S, Castain R, Siegel H.J, Maciejewski AA, Banka T, Chindam K, Dussinger S, Pichumani P, Satyasekaran P, Saylor W, Sendek D, Sousa J, Sridharan J, Sugavanam P, Velazco J. Static Mapping of Subtasks in a Heterogeneous Ad-hoc Grid Environment. In: IEEE. Parallel and Distributed Processing Symposium; 26-30 April 2004; Santa Fe, NM, USA.
11. Momeni H, Sharifi M, Okhovvat M. A Mathematical Approach to Reduce the Mean Number of Waiting Tasks in Wireless Sensor Actor Networks. Information J 2012; 15: 181-192.
12. Byun H, So J. Node Scheduling Control Inspired by Epidemic Theory for Data Dissemination in Wireless Sensor-Actuator Networks With Delay Constraints. IEEE Trans. on Wireless Communications 2016; 15: 1794-1807.
13. Park H, Srivastava MB. Energy-Efficient Task Assignment Framework for Wireless Sensor Networks. In: Center for Embedded

Networking Sensing (CENS), Technical Report; 01 January 2003; California, USA: pp. 2-11.

14. Armstrong R, Hensgen D, Kidd T. The Relative Performance of Various Mapping Algorithms is Independent of Sizable Variances in Run-time Predictions. In: IEEE. Heterogeneous Computing Workshop; 30-30 March 1998; Orlando, FL, USA: pp.79-87.
15. Freund RF, Siegel H.J. Heterogeneous Processing. IEEE Computing 1993; 26: 13-17.
16. Brucker P. Scheduling Algorithms. 5th ed. Berlin, Germany: Springer, 2007.
17. Ibarra OH, Kim C.E. Heuristic Algorithms for Scheduling Independent Tasks on Non-identical Processors. J ACM 1997; 24: 280-289.
18. Bolch G, Greiner S, De Meer H, Trivedi KS. Queuing Networks and Markov Chains. 2nd ed. USA: Wiley, 2006.
19. Cooper RB. Introduction to Queuing Theory. 2nd ed. Amsterdam, Netherlands: Elsevier, 1981.
20. Burke PL. The Output of a Queuing System. Operation Research 1956; 4: 699-704.
21. Trivedi KS. Probability and Statistics with Reliability, Queueing, and Computer Science Applications. 2nd ed, USA: Wiley, 2001.
22. MATLAB - MathWorks - MATLAB & Simulink. [Online]. Available: https://www.mathworks.com/products/matlab.html.
23. Liu H, Leung Y.W, Chu X, Ad-hoc and Sensor Wireless Networks: Architectures, Algorithms and Protocols, Hong Kong: Bentham Science Publishers, 2009.