A Task Scheduling Algorithm Based on Q-Learning for WSNs

Benhong Zhang[1], Wensheng Wu[2], Xiang Bi[1(✉)], and Yiming Wang[1]

[1] School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230009, Anhui, China
bixiang@hfut.edu.cn

[2] Intelligent Manufacturing Institute, Hefei University of Technology, Hefei 230009, Anhui, China

Abstract. In industrial Wireless Sensor Networks (WSNs), the transmission of packets usually have strict deadline limitation and the problem of task scheduling has always been an important issue. The problem of task scheduling in WSNs has been proved to be an NP-hard problem, which is usually scheduled using a heuristic algorithm. In this paper, we propose a task scheduling algorithm based on Q-Learning for WSNs called Q-Learning Scheduling on Time Division Multiple Access (QS-TDMA). The algorithm considers the packet priority in combination with the total number of hops and the initial deadline. Moreover, according to the change of the transmission state of packets, QS-TDMA designs the packet transmission constraint and considers the real-time change of packets in WSNs to improve the performance of the scheduling algorithm. Simulation results demonstrate that QS-TDMA is an approximate optimal task scheduling algorithm and can improve the reliability and real-time performance of WSNs.

Keywords: Wireless sensor networks · Q-Learning · Task scheduling

1 Introduction

In recent years, the flexibility and cost efficiency of wireless networks have become the main motivations for adopting wireless communications in industrial environments. Various network specifications such as WIA-PA [16], WirelessHART and ISA 100.11a [13] have been used to meet strict industrial requirements like real-time and reliability.

One of the major approaches to improve network performance is to use TDMA-based scheduling algorithm. The classic scheduling algorithms of WSNs based on TDMA are mainly the Earliest Deadline First (EDF) [15] and the

This paper is supported by the National Key Research Development Program of China (2016YFC0801800), Anhui Postdoctoral Research Funding Project (2017B144) and the National Natural Science Foundation of China (61501161).

X. Liu et al. (Eds.): ChinaCom 2018, LNICST 262, pp. 521–530, 2019.
https://doi.org/10.1007/978-3-030-06161-6_51

improvement of EDF algorithm [9]. But the problem of packet transmission scheduling in WSNs has been proved to be an NP-hard problem [4,10]. It is difficult to achieve an optimal or approximate optimal scheduling scheme by using the traditional methods.

In this context, many researchers have turned their attention to the field of machine learning [1]. Many novel methods based on machine learning have been applied to many aspects of scheduling tasks. The role-free clustering with Q-Learning for Wireless Sensor Networks (CLIQUE) is introduced in [6]. CLIQUE allows each node to investigate its capabilities as a cluster head node by combining the Q-Learning algorithm with some dynamic network parameters. A task scheduling algorithm for wireless sensor networks based on Q-Learning and sharing value function(QS) to solve the problem of frequent exchange of cooperative information in WSNs is introduced in [14]. QS can ensure that the nodes complete the application functional requirements while performing good cooperative learning. Considering that the real-time scheduling problem in multi-hop wireless network, a markov decision process of the packet transmission is proposed in [8]. All the above papers proved superiority of solving problems related to WSNs based on machine learning. However, few of them research on scheduling algorithms that improve the real-time and reliability of the network.

In this paper, We propose QS-TDMA algorithm for real-time scheduling of WSNs with strict deadline limitation in the form of flow. The algorithm uses Q-Learning [5] to achieve an approximate optimal scheduling scheme and improves the real-time and reliability of WSNs under the packet transmission constraint.

2 System Model

2.1 Data Flow Model

The transmission process of the generated packet sent from the source node to the destination node defined as a data flow. We consider that a set of M flows $F = \{f_1, f_2, \cdots, f_M\}$ are arranged in a single frequency band, The flow f_i is defined as follows:

$$f_i = (T_i, D_i, \phi_i, H_i) \tag{1}$$

Where $1 \leq i \leq M$, T_i represents the packet generation period of flow f_i; positive discrete variable D_i represents the deadline for each packet generated from the source, and $D_i \subset \{h_i^*, h_i^* + 1, \cdots, D_i^*\}$, h_i^* denotes the total number of hops required for the flow f_i to be transmitted from the source node to the destination node, and the value of D_i^* is large enough; ϕ_i represents the route of the flow f_i; H_i represents the length of f_i.

The packet transmission state on the flow f_n is defined as (t_n, h_n), t_n denotes the remaining deadline of packet transmission to the destination node, and h_n denotes the remaining hops of packet transmission from the current node to the destination node. If $t_n > h_n$, it means that the packet can be transmitted to the destination theoretically. If f_n is assigned to a time slot, the current transmission state of f_n is changed to $(t_n - 1, h_n - 1)$, and the transmission state on the other flows changes to $(t_n - 1, h_n)$. If $t_n = 0$, a new packet will be generated and entered into the flow to wait for transmission scheduling, and the packet state will be changed to the initial state (D_n, h_n).

2.2 Q-Learning Model

Reinforcement learning is a branch of machine learning. By identifying optimal strategies, each state is mapped to actions that the system should take in these states in order to maximize the numerical target reward over time [2]. Figure 1 illustrates the reinforcement learning model for task scheduling. It performs actions in WSNs and uses the reward feedback of a specific environment as a new learning process for the next experience.

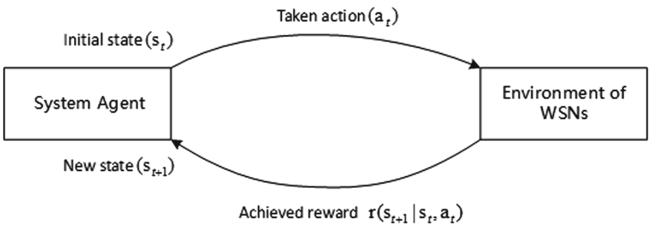

Fig. 1. Reinforcement learning model

As a popular method of reinforcement learning, Q-Learning can learn the usefulness of each task in the system and the benefit value of task execution in a certain environment over time to achieve the best adaptation to the current environment. Before the packet transmission, the system will give an immediate reward for the current action and make an evaluation. Before the end of the current transmission, the cumulative reward in the finite-state space is calculated by the value function and the Q-value evaluation is given. In the process of system learning, considering the flow transmission and constraints in the actual application scenario, we make the following assumptions:

- Only one flow per slot performs packet transmission;
- The packet generated on the flow has a strict deadline, assuming that the packet generation period is equal to the deadline;
- The probability of successful transmission from node i to node j through wireless communication will be affected by many physical factors such as transmission power, coding method, and modulation scheme. In this paper, we only consider the effect of the scheduling of node time slots, the probability that the node successfully transmitted to the next node $p = 1$.

Q-Learning algorithm can be seen as a random expression of the value iterative algorithm. The value iteration can be expressed by the action value function [12], and $V^{\pi}(S)$ represents that under the strategy π, the system performs the action f with the probability $P(S'|S,f)$ from the state S to the next state S'. The action value function of the state S is defined as follows:

$$V^{\pi}(S) = \max_{f \in F}[R(S'|S,f) + \gamma \sum_{S' \in S} P(S'|S,f)V^{\pi}(S')] \tag{2}$$

Where $P(S'|S,f)$ represents transition probability of the system when the agent select the flow f to perform from state S to state S'; $R(S'|S,f)$ represents the average reward for state transitions and γ is the discount factor, $\gamma \in (0,1)$. The optimal strategy is to obtain the execution action that maximizes the value function. The optimal strategy $\pi^*(S)$ in the state S defined as follows:

$$\pi^*(S) = \arg V^{*(\pi)}(S) = \arg\max_{f \in F}[R(S'|S,f) + \gamma \sum_{S' \in S} P(S'|S,f)V^{\pi}(S')] \tag{3}$$

In our Q-Learning model, the Q-value function of time slot t is defined as follows:

$$Q_t(S_t,f) = R(S'|S,f) + \gamma \sum_{S' \in S} P(S'|S,f) \max_{f \in F} Q_t(S_t,f) \tag{4}$$

Where $Q_t(S_t,f)$ represents the Q-value corresponding to the flow f selected by the state S in the two-dimensional table of state actions; $\max Q_t(S_t,f)$ represents that at the t time slot, the system moves from state S to the next state S' of all flows that may perform packet transmission tasks and selects one of the actions that maximizes its Q-value. The Q-value update of the system is defined as follows:

$$Q_t(S_t,f) \leftarrow (1-\alpha)Q_t(S_t,f) + \alpha[R(S'|S,f) + \gamma \sum_{S' \in S} P(S'|S,f) \max_{f \in F} Q_t(S'|S,f)] \tag{5}$$

Where $\alpha \in (0,1)$ represents the learning rate factor. The larger α is, the more the system learning process relies on reward function and value function, the smaller α is, the more the system relies on accumulated learning experience and the slower the learning rate is.

3 QS-TDMA Scheduling Algorithm

3.1 System Space

In order to implement the task scheduling problem for M flows, the system needs to select a flow to perform the task in a hyper-period scheduling table. Hyper-period is usually defined in the industrial environment as the least common multiple of the packet generation periods of the field devices [15]. In this paper,we define the hyper-period as the least common multiple of the time slots to the deadlines of all flows. Therefore, the action space of the system is to determine which flow is assigned at each time slot, that is, the action space is $A = \{f_1, f_2, \cdots, f_M\}$. Each time slot of the hyper-period is mapped to the state space of the system, the state space is $S = \{1, 2, \cdots, T\}$.

3.2 Reward Function

The reward function reflects the value of rewards and punishments for the execution of the task, including two ways proposed in [11,14], respectively. Literature [11] adopts the mutative reward mechanism to achieve the applicability prediction to control the task executions during the learning process, while Literature [14] defines different fixed reward value for each task according to the priority of task execution in the application. In this paper, we consider the immediate reward of real-time flows allocated to the time slots, represented by r, and the influence of other flows not assigned to the time slots, represented by R_L. The combination of two factors reward function is defined as follows:

$$R(S, f) = r + R_L \tag{6}$$

The immediate reward r is composed of the total number of hops of the flow and the initial deadline. The smaller the total number of hops, the longer the initial deadline, and the smaller the value, the lower the priority of the current flow. The immediate reward r defined as follows:

$$r = k_1 \frac{h}{t} + k_2 \frac{1}{t - h + 1} \tag{7}$$

Where $\frac{h}{t}$ reflects the urgency of the packet; $t-h$ reflects the effect of the actual remaining time, which is not reflected in $\frac{h}{t}$; $t \geq h$, k_1, k_2 satisfy $0 < k_1, k_2 < 1$ and $k_1 + k_2 = 1$.

R_L denotes the feedback of the action, which reflects the negative reward. When the system in the state S_i and select f^i to perform in slot i, assume that there are L_{i0} flows in all flows satisfies $t_i - h_i = -1$, L_{i1} flows satisfies $t_i - h_i = 0$, and L_{i2} flows satisfies $t_i - h_i = 1$ before entering the next state S_{i+1}. Then we can define R_L as follows:

$$R_L = -(\rho_1 L_{i0} + \rho_2 L_{i1} + \rho_2 L_{i2}) \tag{8}$$

Where ρ_1, ρ_2, ρ_3 is the relevant discount parameters, $0 < \rho_1, \rho_2, \rho_3 < 1$,$\rho_1 > \rho_2 > \rho_3$, and $\rho_1 + \rho_2 + \rho_3 = 1$. Combining Eqs. (6)–(8) to get final reward function as follows:

$$R(s, f) = k_1 \frac{h}{t} + k_2 \frac{1}{t - h + 1} - (\rho_1 L_{i0} + \rho_2 L_{i1} + \rho_2 L_{i2}) \tag{9}$$

Local separation and combination of reward parameters and reward factors allow the reward function to be adjusted for external weights. And hence the behavior of the overall system is up to the initial states and the reward feedback of all flows.

3.3 Exploration-Exploitation Policy

In the trial and error process of the system, the relationship between exploration and exploitation needs to be balanced. The general $\varepsilon - greedy$ strategy is prone to converge rapidly. Developing in a situation where exploration is not adequate can result in a short learning process and serious learning biases. In this paper, we introduce the Metropolis Criterion (MC) in the Simulated Annealing [3,7] method into the flow selection in our exploration and exploitation, which can better solve the problem of excessive convergence and the balance between exploration and exploitation. The exploration probability ε_t is defined as follows:

$$\begin{cases} \varepsilon_p = \exp[-\left|(Q(S,a_r) - Q(S,a_o))\right|/KT_k] \\ \varepsilon_t = \max\{\varepsilon_{min}, \varepsilon_p\} \end{cases} \tag{10}$$

Where $Q(S,a_r)$ represents the Q-value that randomly selects an action in the state S; $Q(S,a_o)$ represents the Q-value that selects an optimal action in the state S; T_k is a fixed value, K is a coefficient, and $K = \lambda^e$, decline factor $\lambda \in (0,1)$, e is the number of learning, as e increases, the value of ε_p will become smaller and smaller, and the entire exploration process will become stable; ε_t is the exploration probability based on MC, which is maximum value of ε_p and ε_{min}; ε_{min} is the minimum exploration probability given, which is the lower bound of exploration.

3.4 Q-Value Function Update

The definition of Q-value function update can be known from formula (5). In the formula, $P(S'|S,f)$ denotes the probability that the system selects flow f from the state S to the next state S'. $P(S'|S,f)$ is usually unknown, but the Q-Learning algorithm is obtained by replacing $R(S,f)+\sum_{S'\in S} P(S'|S,f) \max_{f\in F} Q_t(S'|S,f)$ by its simplest unbiased estimator built from the current transition $R_{t+1} + \max_{f\in F} Q_t(S'|S,f)$, In this way, the final Q-Learning algorithm Q-value function update formula is obtained:

$$Q_t(S_t,f) \leftarrow (1-\alpha)Q_t(S_t,f) + \alpha[R(S'|S,f) + \gamma \max_{f\in F} Q_t(S'|S,f)] \tag{11}$$

3.5 Algorithm Description

The process description of our Q-Learning task scheduling algorithm can be obtained from the above two parts, the system model and the QS-TDMA scheduling algorithm. But in industrial environment, we have to consider the packet transmission scheduling process of WSNs is subject to a strict deadline limitation. For packets whose remaining time is less than the remaining transmission hops, it is impossible to be sent to the destination node theoretically and the packet will be lost. If we continue to allocate time slots for these flows, it is undoubtedly a waste of resources, and it will lengthen the entire process of learning and exploring. Therefore, we add selection constraints in the process of trial and error, and we do not allocate time slots for the flows that lose the meaning of theoretical transmission. By this way, the efficiency of the exploration-exploitation policy and the accuracy of system convergence can be improved.

In the process of learning, the system always has MDP feature in the face of external environment, and gradually rewards the flow with a small number of packet losses to perform packet transmission task, and finally obtains an approximate optimal scheduling algorithm. Specific algorithm description is shown in Table 1.

Table 1. QS-TDMA algorithm description

```
Algorithm 1 QS-TDMA algorithm description
1.  Initialize $Qmatrix$, $T$, $\alpha$ etc;
2.  Episode start;
    Initialize state $S_i = 1$, $i = 1$;
3.  If the number of episode reaches the requirement, repeat 7, otherwise
    select the task $f^t$ to be performed according to
    the exploration-exploitation strategy in time slot $i$;
    3.1: Produce a random number $a (a \in (0, 1))$;
    3.2: Select the optimal task $f_p$;
    3.3: Produce a random task $f_r$, calculate $\varepsilon_t$;
    3.4: If $a < \varepsilon_t$, $f^t = f_r$, otherwise $f^t = f_p$;
    3.5: If the state of the packet $f^t$ is $t_{f^t} < h_{f^t}$, repeat 3.1;
4.  Execute task $f^t$, obtain reward $R_t$;
5.  According to (9) and (12), update the Q-value;
6.  If $S_t < T$, the system goes to the next state $S_{t+1}$, $i = i + 1$, repeat 3,
    otherwise repeat 2;
7.  End.
```

4 Simulation Analysis

In this paper, we simulate the performance of the algorithm and give the experimental results. The basic parameters are set as follows: the learning rate and discount factor $\alpha = \gamma = 0.9$; the weight of the two influencing factors of the immediate reward function $k_1 = k_2 = 0.5$; the influence parameters of flows that are not assigned to time slots $\rho_1 = 0.5, \rho_2 = 0.4, \rho_3 = 0.1$; The constant term in the exploration-exploitation policy $T_k = 1000$, $\lambda = 0.9$ and $\varepsilon_{min} = 0.01$.

We use the number of lost packets in a hyper-period as the criterion to measure the performance of the scheduling algorithms. We assume that the 'Optimal' algorithm (OP) in literature [8] is achievable and we compare QS-TDMA with other three strategies, respectively, as OP, EADF and RB. Analyze the network performance of QS-TDMA and other three algorithms in the number of different flows and different deadlines.

In Table 2, we consider two flows like [8], the total number of hops for the two flows with $H_1 = H_2 = 2$, the deadlines with $D_1 = 2, D_2 = 6$ and the hyper-period with $T = 1000$. The hyper-period takes a long enough value to offset the effect of randomness. In the following experiment, the hyper-period take the same value. Although the result of the QS-TDMA is good enough, but the result of learning has minimal fluctuations. The simulation data here take the average of 10 results, later comparisons will also be compared in the same way. As shown in the Table 2, the results of the QS-TDMA are close to the OP and the performance is much better than EADF and the RB. The scheduling of the RB is the worst in the feasible algorithms.

Table 2. The number of lost packets in the scheduling algorithm

Algorithm	QS-TDMA	OP	EADF	RB
Number of lost packets	176	166	251	500
Number of successful packets	490	500	415	166
Loss rate (%)	26.4	24.9	37.7	75.1

In Fig. 2, we consider the change in the number of lost packets for QS-TDMA, EADF, RB and OP as the flows increase. The number of selected flows increase from 2 to 6 in sequence. We consider three sets of symmetric flows. The deadline of the flow is randomly generated, and the flow deadline is one to three times the total hop count. The total number of hops for the flow is 2, 2, 3, 3, 5, 5, and the corresponding deadline is 2, 6, 6, 9, 10, and 15. As we can see in the figure, with the number of flows increase, the average number of lost packets for the four methods is increasing. Considering that only one flow can be sent for each time slot in a single frequency band, this result is in line with the actual situation. At the same time, it can be found that the OP is the optimal scheduling with the least theoretical packets loss. The QS-TDMA is close to OP, EADF is the second, and RB loses most packets. Moreover, with the increase in the number of flows, the result of RB is worse.

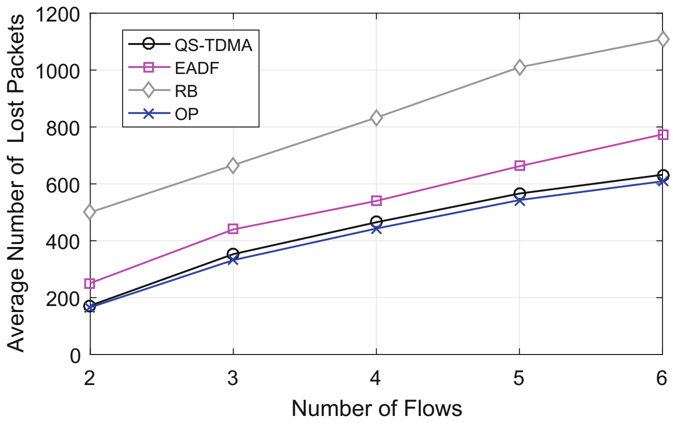

Fig. 2. The total number of lost packets with the increase of flows

In Fig. 3, we consider three flows with a total number of hops of $H_1 = 1, H_2 = H_3 = 3$, and the deadline for the three flows is the same. We analyze the number of lost packets for the four algorithms when the deadline increases from 3 to 8. It can be seen from Fig. 3 that with the increase of the deadline, the OP has always been the optimal result. When the deadline increases from 3 to 6, EADF and RB have the same number of lost packets, and performance of the EADF is worse than that of the RB when the deadline reaches 7. This is because the EADF is scheduled based on the principle of minimum average deadline first. The performance of EADF decreases with the same deadline for each flow. The overall performance of QS-TDMA is stable with the increase of deadline, and the performance is better than EADF and RB.

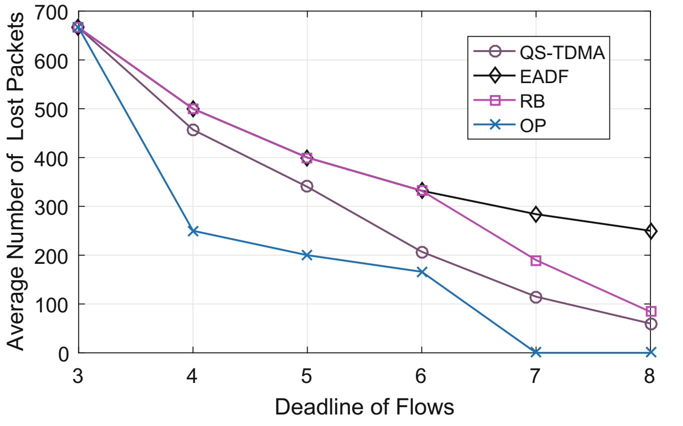

Fig. 3. The total number of lost packets with the increase of deadlines

5 Conclusion

In this paper, we proposed a TDMA-based task scheduling algorithm for wireless sensor networks. The algorithm combined the two factors of priority and real-time change of the packets. Under the given packet transmission constraints, an approximate optimal scheduling scheme is achieved by rewarding feedback and value iteration of system scheduling. The next step of the paper, we will consider how to improve the real-time and reliability of WSNs in the case of allowing concurrent data.

References

1. Abu Alsheikh, M., Lin, S., Niyato, D., Tan, H.P.: Machine learning in wireless sensor networks: algorithms, strategies, and applications. Commun. Surv. Tutor. IEEE **16**(4), 1996–2018 (2015)

2. Arnold, B.: Reinforcement learning: an introduction (adaptive computation and machine learning). IEEE Trans. Neural Netw. **9**(5), 1054 (1998)
3. Chen, S.L., Wu, H.Z., Xiao, L., Zhu, Y.Q.: Metropolis policy-based multi-step Q learning algorithm and performance simulation. J. Syst. Simul. **19**(6), 1284–1287 (2007)
4. Choi, H., Wang, J., Hughes, E.A.: Scheduling on sensor hybrid network. In: International Conference on Computer Communications and Networks, 2005. ICCCN 2005. Proceedings, pp. 503–508 (2013)
5. Watkins, C.J.C.H.: Q-learning. Mach. Learn. **8**, 279–292 (1992)
6. Forster, A., Murphy, A.L.: Clique: role-free clustering with Q-learning for wireless sensor networks. In: IEEE International Conference on Distributed Computing Systems, pp. 441–449 (2009)
7. Guo, M., Liu, Y., Malec, J.: A new Q-learning algorithm based on the metropolis criterion. IEEE Trans. Syst. Man Cybern. Part B Cybern. A Publ. IEEE Syst. Man Cybern. Soc. **34**(5), 2140 (2004)
8. Kashef, M., Moayeri, N.: Real-time scheduling for wireless networks with random deadlines. In: IEEE International Workshop on Factory Communication Systems, pp. 1–9 (2017)
9. Li, Q., Ba, W.: Two improved EDF dynamic scheduling algorithms in soft real-time systems. Chin. J. Comput. **34**(5), 943–950 (2011)
10. Saifullah, A., Xu, Y., Lu, C., Chen, Y.: Real-time scheduling for WirelessHART networks, pp. 150–159 (2010)
11. Shah, K., Kumar, M.: Distributed independent reinforcement learning (DIRL) approach to resource management in wireless sensor networks. In: IEEE International Conference on Mobile Adhoc and Sensor Systems, pp. 1–9 (2008)
12. Sigaud, O., Buffet, O.: Markov Decision Processes in Artificial Intelligence. ISTE, New York (2010)
13. Stig, P.: A comparison of WirelessHART and ISA100.11a for wireless instrumentation (2014)
14. Wei, Z., Zhang, Y., Xu, X., Shi, L., Feng, L.: A task scheduling algorithm based on Q-learning and shared value function for WSNs. Comput. Netw. **126**, 141–149 (2017)
15. Wu, C., Sha, M., Gunatilaka, D., Saifullah, A.: Analysis of EDF scheduling for wireless sensor-actuator networks. In: Quality of Service, pp. 31–40 (2014)
16. Yu, P.: Analysis on features of industrial wireless standard WIA-PA and the application in prospect. Process Autom. Instrum. **31**(1), 1–4 (2010)