A Round-robin Scheduling Algorithm of Relay-nodes in WSN Based on Self-adaptive Weighted Learning for Environment Monitoring

Liu Zhang
School of Design, Hunan University, Changsha, Hunan, China
Email: zhangliu1984@gmail.com

Feihu Chen
School of Architecture, Hunan University, Changsha, Hunan, China
Email: fhchen@hnu.edu.cn

***Abstract*—Wireless Sensor Network(WSN) for environment monitoring, which typically has heavy data transmission load, has critical real-time requirement. Thus the time delay at relay-nodes should be reduced. This paper models queue scheduling as a reinforcement learning process and presents a scheduling algorithm of relay-node based on self-adaptive weighted learning. The presented algorithm schedules queues dynamically. Simulation results under two circumstances (with sufficient bandwidth and limited bandwidth) show that the algorithm can improve the real-time performance and maintain fairness.**

***Index Terms*—wireless sensor network, relay-node, self-adaption weight, round-robin scheduling, environment monitoring**

I. INTRODUCTION

With the development of economy and technology, the demand of environment monitoring is increasing. Environment monitoring is to measure and analyze of environment parameters such as temperature, humidity and oxygen content, etc [1]. Environmental monitoring plays an important role in understanding the environment, and contributes to a better life of human being. Environment monitoring has a wide range of applications, such as coal mine environment monitoring, heritage site environmental pollution monitoring, and so on [2-4].

Wireless Sensor Network (WSN) has been widely used in environment monitoring [5, 6]. WSN consists of spatially distributed autonomous sensors, which monitor environmental conditions and transmit the captured data through network to a main station.

Since environment conditions are time-varying, it's a great challenge to detect the real-time change of environment. In environment monitoring, a large amount of information data needs to be transmitted through WSN. In order to improve the real-time performance, proper flow control needs to be applied in relay-nodes, and queue scheduling is a key part of flow control.

The classical queue scheduling algorithms include First Come First Service (FCFS), Priority queue algorithm and Weighted Round Robin (WRR) [7]. FCFS is simple but it can't meet the different requirements of QoS. The Priority queue can deal with different QoS requirements, but it cause large delay to low priority queues, hence it is unfair for these queues. WRR improves the fairness of queue scheduling, but it can't provide an upper bound of delay. Therefore, it is not suitable for the environment monitoring which has critical real-time requirement [8].

Considering that WRR is a simple algorithm which can be implemented easily, we modified WRR to improve its real-time performance and take advantage of its fairness.

There are many different types of data converge in the relay-nodes of WSN. In order to guarantee the real-time of network and fairness of queue scheduling, we introduce the concept of differentiated service. The round robin algorithm can scheduling different data flow dynamically and adjust network resource based on different priority.

During environment monitoring, a fixed schedule weight can not reflect the change of environment. The schedule weight should be adjusted dynamically according to the change of environment. In this paper, reinforcement learning is adopted to realize dynamic adjustment of the schedule weight.

Reinforcement learning is one of the branches of machine intelligence learning [9, 10]. The Agents choose one strategy to control or affect the system's transforming based on states of the system. Every strategy defines one stochastic process and its objective function value. The basic idea of reinforcement learning is to punish the unexpected outcomes, and gradually form a strategy towards desired outcomes [11]. Reinforcement learning calculates a better or best outcome in dynamic system ultimately by the approach of trial and error or the dynamic interaction with environment [12]. Currently, reinforcement learning has achieved perfect performance and been used widely in many occasions where dynamical change with environment is needed [13-16].

In this paper, we take advantage of the Reinforcement Learning (RL) [17]. The queue scheduling is modeled as a RL process, and the queue scheduling weights is adjusted according to different priority and length of queue by RL algorithm. Consequently, an optimal round robin scheme is achieved.

doi:10.4304/jcp.9.4.830-835

II. SELF-ADAPTIVE LEARNING MODEL OF RELAY-NODES

There are different types of data with different real-time requirements in relay-nodes. All data should be buffered in queues when waiting for transmission.

The queues in relay-nodes have different priority. The relay-nodes can realize complex intelligent scheduling algorithm to decide the transmission of data in queue. So each relay-node can be considered as an agent. The queues in relay-nodes can be divided into three priority levels: Queue 0, Queue 1 and Queue 2. Queue 0 has the highest priority to transmit data and Queue 2 adopts the best effort method to transmit data with no specific delay requirement. Accordingly, the priority of Queue 1 is between Queue 0 and Queue 2. There are two assumptions about the queues in relay-nodes:

(1) Incoming data packets are assigned into appropriate queues based on their real-time requirements;

(2) The data packet can't be assigned into other queue again once it has been assigned into one queue;

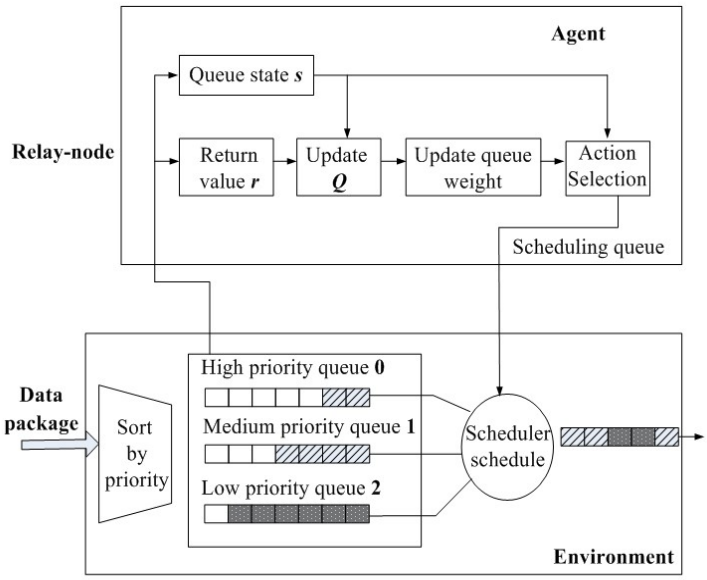

Figure 1. Self-adaptive learning model of relay-nodes

Fig. 1 shows the self-adaptive learning model of relay-nodes. In the model, each queue in a relay-node has a predefined length. The length is decided according to the queue's priority and the real-time demand, which will not change during the running time. The relay-node checks the actual length of each queue, which changes during the data transmission. Then the two lengths will be compared to obtain the state set S in the learning model. S can reflect the time delay of each queue.

After the scheduling of each queue, the queue state set S will change. The return value r of the learning model can be obtained to measure the effect of scheduling. Then the Q value will be updated by relay-node and can be used to update the queue weight. The definitions of r and Q value will be illustrated in detail in Section III. Action selection contains the queue number of next scheduling. The scheduling which is obtained by learning can reduce the time delay significantly.

III. SELF-ADAPTIVE RELAY-NODE SCHEDULING ALGORITHM BASED ON WEIGHTED LEARN ROUND-ROBIN

Definition 1: State set vector S reflects the queue's delay condition.

Given an scheduled length threshold R_i, the higher of the queue's priority, the smaller of R_i. At the sampling site, collect the length of queue i which is denoted as M_i, then compare R_i and M_i. $S=\{s_0, s_1, s_2\}$, Where

$$s_i = \begin{cases} 0 & M_i \le R_i \\ 1 & M_i > R_i \end{cases}.$$

As we all know that, the vector has 8 type values, from the smallest delay {0, 0, 0} to the largest delay {1, 1, 1}.

Definition 2: Action set $A=\{a_1, a_2, a_3\}$, where a_i denotes the next action to schedule queue i.

Definition 3: Individual return value $r_{i.}$ is defined as :

$$r_i = (M_i - R_i)^+ = \begin{cases} 0 & M_i \le R_i \\ M_i - R_i & M_i > R_i \end{cases} \quad (1)$$

$r_{sum} = -\sum_{i=0}^{2} \omega_i r_i$ is the sum of the return value, where $0 < \omega_i \le 1$, $i = 0, 1, 2$. The larger value of w_i the more sensitive of the queue, meanwhile, the value drive the queue scheduler tend to schedule the queue which in worse delay .

Definition 4: Penalty factor r_H. It should be punished when scheduling the lower queue where the band resource is shortage. Penalty factor r_H is defined as:

$$r_{Hi} = \begin{cases} 1 & \text{if } R_i \le \min R_j \ \forall j = 0,1,2 \text{ and } j \ne i \\ K & \text{else} \end{cases} \quad (2)$$

Where K is a constant and $K > 1$, the total penalty factor r_H can be defined as following:

$$r_H = \sum_{i=0}^{2} \lambda_i r_{H\_i} \quad (3)$$

Where $0 < \lambda_i \le 1$, $i = 0, 1, 2$. Also, the reward function is a scalar of state set to action set mapping .We can judge whether the actions' effect is good or bad. This result can in turn affect the choice of action strategy.

$$r(s,a) = r_{sum} \times r_H . \quad (4)$$

Obviously, the largest return value is 0 when all of the queues' delay is met.

After one scheduling process, the agent can calculate the instant income, and updates the value of Q. $Q(s, a)$ denotes that execute action a at state s. In the schedule model of this paper, the Q is a forecast of queue state and return value after scheduling former queue. The larger the Q is, the higher probability of the queue is scheduled.

$$Q^{\pi}(s,a) = E[r \mid s_0 = s, a, s'] = r(s,a) + \gamma \sum P(s' \mid s,a) V(s', \pi) \quad (5)$$

Where s' represents the next state following the finished state *s*.

$P(s' \mid s,a)$ is the probability of transfer the state s' after action *a*. $V(s', \pi)$ is the estimate function.

$$V(s', \pi) = \max_{\pi} Q^{\pi}(s,a) \quad (6)$$

It represents the estimate of the retribution after taking the strategy π. $V(s', \pi) = \max_{\pi} Q^{\pi}(s,a)$.

The learning strategy action is approaching to the queue whose length is large than the schedule queue length R_i. The expression of strategy π is $\pi^*(s) = \arg\max_a Q^*(s,a)$.

Taking the single step Q learning, the update function of Q is in (7).

$$Q(s,a) \leftarrow (1-\alpha) Q(s,a) + \alpha[r(s,a) + \gamma \max Q(s', a)] \quad (7)$$

where α is the learning rate, it's value is $0 \le \alpha \le 1$, γ is the discount factor, it represents the effect of the next Q on the present Q. it's value is $0 \le \gamma \le 1$.Every learning step is greedy, and the goal is $r(s,a) + \gamma \max Q(s', a)$.

Comparing M_i and R_i, we get the state set *S* of the model, *S* is changing along with the schedule action. Combine the *r* and *S*, the relay-node calculates and updates Q, then updates the queue weight according to the value of Q, thus we obtain the queue label *A*, which is passed on to the relay-node scheduler for the queue scheduling.

IV. VEHILCE RELAY-NODE SCHEDULE

A. Single Relay-node Schedule

During the single relay-node scheduling, only the current state is taken into consideration.

Let Q be the weight of queue schedule, the relay-node adaptive weight round-robin schedule algorithm is shown in algorithm 1.

Algorithm1:Single relay-node adaptive weight Round-robin schedule algorithm

input: R_i, M_i

output: The queue label *i* which is ready to be scheduled

BEGIN

Initialize Q value table $Q(s,a)$ arbitrarily;

Initialize action set $A = \{a_1, a_2, a_3\}$;

Initialize state set $S = \{s_0, s_1, s_2\}$;

Repeat (for each episode)

Calculate s_i according to R_i and M_i;

Choose a_i from A using policy derived from Q;

Take action a_i, according to $\varepsilon - greedy$;

Make operating system scheduler schedules queue i;

Observe revenue r_i, state s_i';

Update Q value by (8)

$$Q(s,a) \leftarrow (1-\alpha) Q(s,a) + \alpha[r(s,a) + \gamma \max Q(s', a)] \quad (8)$$

Update state set S;

Store Q value in value table;

Until S is terminal.

END

B. Multiple Relay-node Queue Schedule

Wireless sensor network for the environment monitoring is typically composed of ten or more relay-nodes. The relay-node can be treated as an agent. Multiple agents enforcement learning is expanded from single agent enforcement learning. Each agent can choose the best strategy based on its own transform state and return value. In order to improve the learning efficiency, we assume that each agent only considers its own action strategy; others are just considered as a part of environment [18].

This paper adopts Coordination graphs, where the global function Q is divided into local function Q^i [19]. Every Q^i is based on the subset *a* of global action set.

$$a \in A^i \times A_{j \in \Gamma(i)}{}^{j}. \quad (9)$$

a is the union set of the relay-node *i*'s action set and the other relay-node's action set which can affect *i*.

This divide process can be donated as a digraph $G = (V, E)$, where *V* is the set of nodes (ie., relay-node agents); *E* is the set of side (i, j), representing the direct relationship between relay-node *i* and *j*.

Let *S* denote all of state; A^i denotes the action set of agent *i*; P^i: $S \times A^i \rightarrow \Delta(S)$ denotes the state transition function, and $\Delta(S)$ is the probability distribution; r^i: $S \times A^i \times S \rightarrow R$ denotes the return value of agent *i* and $r^i(s, a^i, s')$ is the return value after agent *i* finished action a^i from state *s* to *s*'; π^i denotes the local strategy of one agent's schedule, so we have π^i : S $\rightarrow A^i$.

$$V^i(s, \pi^i) = E(\pi^i)[r^i \mid s_0 = s] = E(\pi^i)[\sum_{t=0} \gamma^t r_{t+1}^i \mid s_0 = s] \quad (10)$$

Equation(10) is the discount estimation function, r_{t+1}^i is agent *i*'s discount return at time simple t, γ is discount factor, we have $0 \le \alpha \le 1$.

The expression of function Q^i is as follows:

$$Q^i(s, a, s') = \sum r^i(s, a^i, s') + \gamma \sum P(s' \mid s, a^i) V^i(s, \pi^i) \quad (11)$$

For simplicity, we suppose that every agent's action influences its neighbour only. Every agent has one local function

$Q^i(s,a)$ is obtained based on the relay-node individual action. However, its neighbour relay-nodes must be taken into account.

Q is changeable along with the variation of its queue state and its neighbour agent. Considering the neighbour relay-node agent, we introduce weigh function as following:

$f(i,j)$, it represents the contribution of agent j due to agent *i* updates its Q. We can formulate process as follows.

$$Q^i(s,a^i) \leftarrow (1-\alpha)Q^i(s,a^i) + \alpha[R+\gamma \sum_{j\in\{i\cup\Gamma(j)\}} f(i,j)\max Q^j(s',a^j)] \quad (12)$$

Under the single relay-node schedule algorithm, we propose the multiple relay-node adaptive weight learning round-robin schedule algorithm.

Algorithm2:Multiple relay-node adaptive weight learning round-robin schedule.

input: $R_i, M_i, f(i,j)$

output: The queue label *i* which is ready to be scheduled.

BEGIN

Initialize each relay-node's Q value table $Q^i(s,a)$ arbitrarily;

Initialize action set $A=\{a_1,a_2,a_3\}$;

Initialize state set $S=\{s_0,s_1,s_2,...,s_n\}$;

Repeat (for each episode)

Calculate s_i according to R_i and M_i;

Choose a_i from A using policy derived from Q^i in each relay-node;

Take action a_i, according to $\varepsilon-greedy$;

Make operating system scheduler schedules queue *i*;

Observe revenue each relay-node's revenue r_i and state s_i';

Update each relay-node's *Q* value by (13)

$$Q^i(s,a^i) \leftarrow (1-\alpha)Q^i(s,a^i) + \alpha[R+\gamma \sum_{j\in\{i\cup\Gamma(j)\}} f(i,j)\max Q^j(s',a^j)] \quad (13)$$

Update state set *S*;

Store *Q* value in value table;

Until *S* is terminal.

END

V. SIMULATION

In order to illustrate the effectiveness of the proposed algorithm, a simulation on a coal mine environment monitoring is performed. In this monitoring case, temperature, humidity and oxygen content of air are measured to decide whether the conflagration will be occur. The simulation topological graph is depicted in Fig. 2.

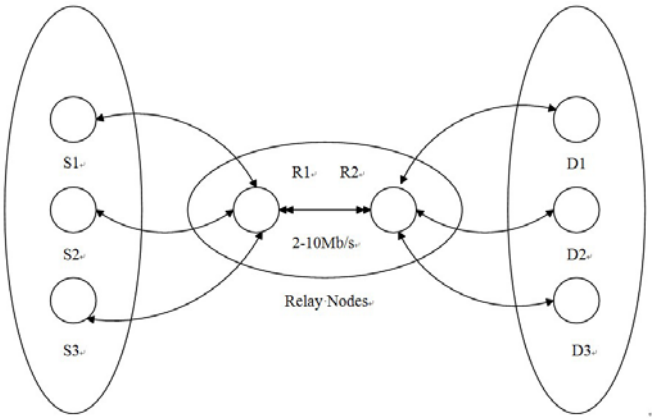

Figure 2. The topological graph of simulation

Assuming there are two relay-nodes in the simulation, relay-node 1 and relay-node 2. The bandwidth between relay-nodes and sensors is 2Mb/s. S1, S2 and S3 are transmission sensors. S1 detects the oxygen content of air; S2 detects the temperature information; S3 detects the humidity information. D1, D2 and D3 are receiving sensors. Simulation time is 60 seconds. S1 is with the highest priority as shown in Table I. The priority of relay-node queue 1, queue 2and queue 3 is decreasing by degrees. The R_i, which is the length of scheduled queue 1, queue 2 and queue 3 is 200 bytes, 400 bytes and 600 bytes respectively, as shown in Table II. Assuming that one unit is 50 bytes in R_i, the learning rate is 0.7, and the discount factor is 0.98.

TABLE I.

THE NETWORK CONFIGURATION

Sending sensors	Receiving sensors	priority
S1	D1	high
S2	D2	middle
S3	D3	low

TABLE II.

THE SIMULATION PARAMETERS FOR THE SCHEDULING

Queue number	Defined path length R_i
1	200 bytes
2	400 bytes
3	600 bytes

1. Comparison of queue length when bandwidth is sufficient.

The link bandwidth is 10Mb/s. The simulation result of queue length with the proposed algorithm and WRR algorithm is shown in Fig. 3.

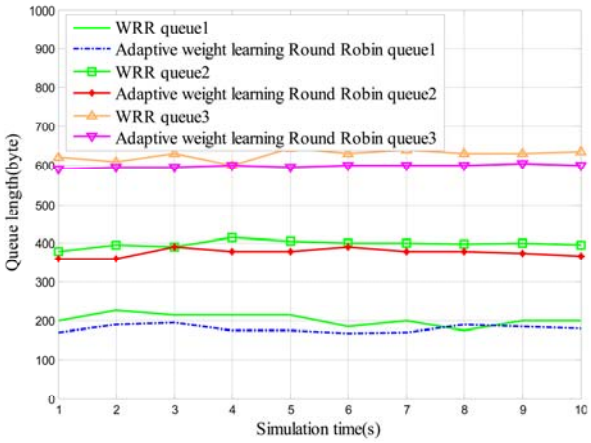

Figure 3.The length of queue when the band resource is sufficient

It could be seen clearly that at the very beginning, the proposed algorithm needs to search the optimal solution. After 2 seconds, the length of queue stays nearby the R_i gradually.

2. Comparison of queue length when bandwidth is limited.

The link bandwidth is 2Mb/s. The simulation result of queue length with the proposed algorithm and WRR algorithm is shown in Fig.4:

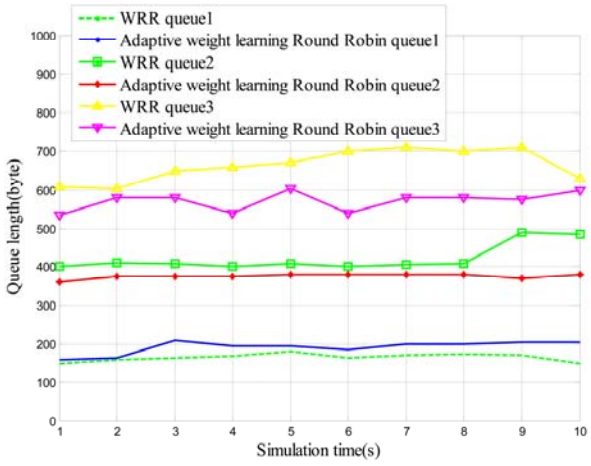

Figure 4.The length of queue when the band resource is insufficient

From Fig. 3 and Fig. 4, it can be found that when the bandwidth resources are sufficient and limited, the lengths of three priority queues are different by using two different scheduling algorithms. Both of the two scheduling algorithms can meet the delay requirement of the highest priority queue, and the queue length can stabilize at a preset value of 200 bytes. However, as the weighted round uses the fixed weight, the scheduling time for the low priority queue is shorter than that for the high priority queue. After a time, the deviation between queue length and the preset value increases. Since the queue 3 has the shortest scheduling time in each round, the queue length is far away from the preset value of 600 bytes, and the delay increases significantly over time.

By learning, the proposed algorithm can adjust the queue weight to accommodate the change of the queue, so the delay requirement of queue 3 with lowest priority can also be meet. That is to say, the proposed adaptive weight learning scheduling algorithm can guarantee real-time transmission of high priority queues. Meanwhile, low priority queues are served properly as well.

VI. CONCLUSION

In this paper, an adaptive weight learning round-robin scheduling algorithm of relay-nodes in WSN is presented for environment monitoring. By applying enforcement learning method, an adaptive weight learning model is developed. Single and multiple relay-node queue scheduling algorithms are investigated. The simulation results show that, the proposed algorithm can achieve better real-time performance while maintaining fairness..

ACKNOWLEDGEMENT

The author wishes to thank Yu Zhiheng and Yao Pingping for their help and technical expertise in this contribution.

REFERENCES

[1] K. Clifford, R. Alex, R. David, et al., “Overview of Sensors and Needs for Environmental Monitoring.” *Sensors*, vol. 5, 2005, pp. 4-37.

[2] M. Li, Y. Liu, “Underground Coal Mine Monitoring with Wireless Sensor Networks.” *ACM Transactions on Sensor Networks* , vol. 5, no. 2, 2009, pp. 10-39.

[3] J. Holder, “Pattern and Impact of Tourism on the Environment of the Caribbean,” *Tourism Management*, vol.9, 1988, pp. 119-127

[4] D. Camuffo, *Microclimate for Cultural Heritage*, Netherlands: Elsevier Science Publishers, 1998.

[5] L. Zhang, Q. Zhu, A. Chen, “Fast Message Dissemination Tree and Balanced Data Collection Tree for Wireless Sensor Network.” *Journal of Software*, vol. 8, 2013, pp. 1346-1352

[6] Y. Yong, X. Xiong, Z. Yong, “Energy Efficient Hierarchical Collaboration Coverage Model in Wireless Sensor Network.” *Journal of Software*, vol. 23, 2012, pp. 2783-2794

[7] T. Velmurugan, H. Chandra, S. Balaji, “Comparison of Queuing Disciplines for Differentiated Services Using OPNET”, *Proc. International Conference on Advances in Recent Technologies in Communication and Computing (ARTCom 09)*, IEEE Press, Oct. 2009, pp.744-746

[8] Y. Zhang, G. Peter, “Performance of a Priority-Weighted Round Robin Mechanism for Differentiated Service Networks,” *Proc. International Conference on Computer Communications and Networks (ICCCN 2007)*, IEEE Press, Aug. 2007, pp.1198-1203

[9] Y. Niv, J. A. Edlund, P. Dayan, et al., “Neural Prediction Errors Reveal a Risk-sensitive Reinforcement-learning Process in the Human Brain.” *The Journal of Neuroscience*, vol. 32, 2012, pp. 551-562.

[10] Q. Zhang, M. Li, X. Wang, Y. Zhang, “Reinforcement Learning in Robot Path Planning.” *Journal of Software* , vol. 7, 2012, pp. 161-166

[11] M. M. Botvinick, “Hierarchical Reinforcement Learning and Decision Making.” *Current opinion in neurobiology*, vol.22, 2012, pp. 956-962.

[12] C. Claus, C. Boutilier, “The Dynamics of Reinforcement Learning in Cooperative Multiagent systems.” *AAAI/IAAI*, 1998, pp. 746-752.

[13] Džeroski S, De Raedt L, Driessens K. "Relational Reinforcement Learning." *Machine learning*, vol.43, 2001, pp. 7-52.
[14] L. Prashanth, "Reinforcement Learning with Function Approximation for Traffic Signal Control." *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, 2011, pp. 412-421.
[15] T. Kim, A. Arora, "RL-based Queue Management for QoS Support in Multi-channel Multi-radio Mesh Networks." *IEEE International Symposium on Network Computing and Application*, 2009, pp. 306-309.
[16] M. Bourenane, "Inductive QoS Packet Scheduling for Adaptive Dynamic Networks." *IEEE International Conference on Communications*, 2008: 3090-3094
[17] R. Sutton, A. Barto, *Reinforcement Learning: an Introduction*. USA: MIT Press, 1998.
[18] M. Bourenane, "Inductive QoS Packet Scheduling for Adaptive Dynamic Networks", *Proc. IEEE International Conference on Communications (ICC 08)*, IEEE Press, May. 2008, pp. 3090-3094.
[19] C. Young-Cheol, "A Survey on Multi-agent Reinforcement Learning: Coordination Problems", *Proc. International Conference on Mechatronics and Embedded Systems and Applications (MESA 10)*. IEEE Press, July. 2010, pp. 81-86.

Liu Zhang received the B.A. degrees from Tongji University in 2006 and M.A. degree from Hunan University in 2009. Ms. Zhang is now a Ph.D candidate at the School of Design, Hunan University, China. Her recent research and design practice are focus on (i) architecture and landscape design in cultural park; (ii) Tourism planning in minority areas etc.

Feihu Chen is a professor in School of Architecture & School of Design, Hunan University. Prof. Chen is the Deputy Dean of School of Architecture, Hunan University, China. His recent research interested in (i) cultural and historic landscape; (ii) Tourism planning of heritage etc.