Adaptivity condition as the extended Reinforcement Learning for MANETs

Saloua Chettibi, Salim Chikhi
MISC Laboratory
Computer Science Departement
University of Mentouri
Algeria
{sa.chettibi, slchikhi}@yahoo.com

ABSTRACT: To design adaptive protocols for MANET, techniques from the field of artificial intelligence have been adopted. *It is evidnet that the efficiency feature is incremental as the bandwidth and energy are scarce resources in MANETs. Moreover, adaptivity is crucial to achieve the routing task correctly in presence of varying network conditions in terms of mobility, links quality and traffic load. In our study we have used the extended application of Reinforcement Learning (RL) technique to achieve adaptive routing in MANETs.*

Keywords: Mobile Ad-hoc Networks, Routing, Reinforcement learning

Received: 19 April 2011, Revised 28 May 2011, Accepted 12 June 2011

1. Introduction

The infrastructure-based and infrastructure-free networks are the two wireless networks which are well known as ad-hoc Networks. In its mobile configuration, the ad-hoc network is called MANET (Mobile Ad-hoc NETwork). In MANET, nodes can randomly join or leave the network and new links appear or disappear accordingly. Furthermore, the wireless medium is rarely stable and can be easily congested due to the limited bandwidth. Besides, mobile nodes are battery-powered and may fail at any time. Hence, network topology changes constantly and unpredictably which complicate the routing task.

To deal with constant changing network conditions in terms of mobility, link quality, available energy-resources and traffic load, arouting protocol for MANETs should be adaptive. To design such adaptive protocols, techniques from the field of artificial intelligencehave been adopted. Particularly, ACO meta-heuristic, which is a subclass of SI (swarm intelligence) algorithms, has made the foundation of the majority and the most significant contributions to adaptive routing problem. Thanks to constant path-probing using ants-like agents, statistical estimates of paths quality are learned and good routing decisions are reinforced. More recently, reinforcement learning has also taken place as an appropriate framework to design adaptive routing policies which have the ability to learn directly by interacting with their operational environment. Our focus,in this paper, is on the modelization of adaptive routing problem in MANETs as a reinforcement learning task.

The remainder of this paper is organized as follows: design issues of routing protocols for MANETs are outlined in section 2. Section 3 introduces the RL-problem definition as well as many elementary RL-algorithms. Next, in section 4, we describe different RL-models for routing problem in MANETs already proposed in the literature.In section 5, we conclude the paper by highlighting theemerging issues and challenges when modeling routing problem in MANETs as a RL-task.

2. Routing Issues in MANETs

Required features of routing protocols for MANETs can be summarized as follows:

1) Adaptivity: A routing protocol for MANETs should be adaptive in face of frequent and unpredictable changes in network topology mainly due to wireless links instability and to nodes mobility and failure after their batteries depletion. Moreover, adaptivity in face of changing traffic loads is important to avoid congestion areas in the network.

2) Robustness: Control and data packets can be lost due to the poor quality of wireless connections and to the interference between simultaneous transmissions. Robustness is a crucial feature to keep the routing protocol operating correctly even when such losses occur.

3) Efficiency: Efficiency is important to deal with bandwidth, processing power, memory and energy limitations in MANETs. A Routing protocol should be efficient by optimizing its exploitation of network resources.

4) Scalability: In many deployment scenarios of MANETs, network size can grow to very large sizes. Hence, scalability should be taken in consideration in routing protocols design.

In MANETs literature, several routing protocols that try to streak a balance between the abovementioned features have been proposed. However, almost all conventional routing protocols for MANETs suffer from their lack of adaptivity leading to their performance degradation under varying network conditions.Indeed, existing routing protocols for MANETs make very simplistic assumptions about the network characteristics such as perfect radio links and random topology[1],[2]. Furthermore, some routing protocols functional parameters are simply prefixed thresholds although the fact of their dependency on many network conditions [3],[4].

3. Reinforcement Learning

The reinforcement learning [5] is a sub-area of machine learning concerned with learning from interaction by trials and errors how to behave in order to achieve a goal.The RL agent interacts with its environment over a sequence of discrete time steps (see Figure 1). In any RL problem, we distinguish four basic elements: 1) actions: are the choices made by the agent; 2) states: are the basis for making the choices; 3) rewards: are the basis for actions' evaluation; and 4) policy: which is a stochastic rule by which the agent selects actions as a function of states. In the RL problem, the agent objective is to maximize the amount of reward it receives over time.

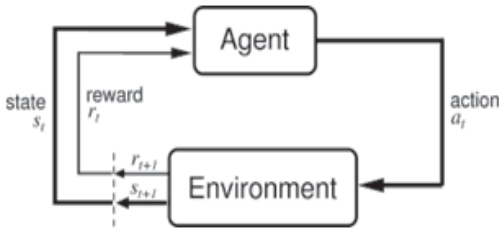

Figure 1.The Agent-Environment Interaction in Reinforcement Learning

Important notions in RL problem formulation as a Markov Decision Process and its resolution can be summarized as follows [5]:

1) Markov property: An environment satisfies the Markov property if the state signal compactly summarizes the past without degrading the ability to predict the future. If the Markov property holds, then the RL environment is called a Markov Decision Process (MDP).

2) Markov Decision Process (MDP): Formally, a finite MDP is a tuple < S,A,T,R > where S is a finite set of environment sates, A is a set of actions available at the agent, $T:S\times A \rightarrow \Pi(S)$ is the state transition function giving for each state and action a probability distribution over states, R: $S\times A\times S \rightarrow R$ is the reinforcement function that indicates the real-value obtained when transiting from a state to another taking a particular action.

3) Return: The return R_t is function of future rewards that the agent seeks to maximize. It can be defined as a simple finite sum of rewards when the agent-task breaks to finite-episodes. Instead, for continuing tasks, R_t is formulated as the infinite sum of discounted rewards.

4) Partially Observable MDP (POMDP): The POMDP is a variant of the MDP in which the state of the environment is only partially visible to the learning agent. What are available are indirect, potentially stochastic observations of the environment state.

5) Value-functions: Almost all reinforcement learning algorithms are based on estimating either state-value or action-value functions. State-value function, $V^{\pi}(s)$, estimates the expected future reward to the agent when starting in state s and following the policy p thereafter. Action-value function, $Q^{\pi}(s,a)$, estimates the expected future reward to the agent when it performs a given action in a given state and following the policy p thereafter.

3.1Q-Learning

Q-learning [6] is a model-free Off-policy RL-method that belongs to the class of TD (Temporal Difference) methods. TD methods combine sampling and bootstrapping, where the learning agent takes a sample of just one step, and then bootstraps information. Let us define $< s,a,r,s' >$ to be an experience tuple summarizing a single transition in the environment. Here, *s* is the agent state before the transition, a is its choice of action, *r* the immediate reward it receives and *s´* the resulting state. The one-step Q-learning version of Q-learning algorithm is depicted on the Figure 2.

InitializeQ(s,a) arbitrarily
Repeat (foreachepisode)
Initializes
Repeat(repeat for each step of episode)
ChooseafromsusingpolicyderivedfromQ
Takeactiona,observer,s'
$Q(s,a):= Q(s,a)+\alpha[r+\gamma \max Q(s',a')-Q(s,a)]$
$s \leftarrow s'$
Untilsisterminal

Where: $Q(s,a)$: expected discounted reinforcement of taking action a in state s; α: learning rate; γ: discount factor $(0 \leq \gamma \leq 1)$

Figure 2. The Q-learning Algorithm

3.2 Monte Carlo Methods (MC)

MC methods [5] are model-free RL resolution methods based on averaging sample returns. To ensure that well-defined returns are available, Monte Carlo methods are defined only for episodic tasks. That is, experience is divided into episodes, and all episodes eventually terminate no matter what actions are selected. It is only upon the completion of an episode that action-value functions, Q(s,a), and policies are changed. MC methods are thus incremental in an episode-by-episode sense, but not in a step-by-step sense like Q-learning.

We distinguish two families of MC methods namely: the every-visit and the first-visit MC methods. The former estimates the value of a state-action pair as the average of all returns that have followed visits to the state in which the action was selected, whereas the latter averages only returns following the first time in each episode that the state was visited and the action was selected. In addition, we can find two incarnations of MC methods, namely, on-policy and off-policy MC. The first visit on-policy version with the ε-greedy selection ruleis depicted in Figure 3. Note that, an ε-greedy rule selects the best action most of the time, and selects uniformly with a small probability, ε, an action at random.

3.3 Collaborative Reinforcement Learning (CRL)

CRL [7] extends the conventional RL framework with feedback models for decentralized multi-agent systems. The feedback models include a negative feedback and a collaborative feedback models. The former model decays an agent's local view of its neighborhood either by constraints in the system or by a decay model. The latter model allows agents to exchange the effectiveness of actions they have learned with one another.

CRL system optimization problems are decomposed into a set of discrete optimization problems (DOPs) that are solved by collaborating RL agents. In CRL, the set of actions that an agent can execute include DOP actions that try to solve the DOP locally, delegation actions that delegate the solution of the DOP to a neighbor and a discovery action that allows agents to find new neighbors. In fact, CRL is a model-based RL technique with the following update rule:

$$Q_i(s,a) = R(s,a) + \sum_{\acute{s} \in s_i} Q_i(\acute{s}|s,a).D_i((\acute{s}|s,a) + Decay(V_j(\acute{s}))) \quad (1)$$

Initialize,foralls $\in$ *S, a* $\in$ *A(s):Q(s,a)* $\leftarrow$ *arbitrary;*
Returns(s,a) $\leftarrow$ *emptylist;*π $\leftarrow$ *anarbitrary*ε *-greedypolicy*
Repeatforever
(a)Generateanepisodeusingπ
(b)Foreachpairs,aappearingintheepisode:
R $\leftarrow$ *returnfollowingthefirstoccurenceofs,a*
appendRtoReturns(s,a)
Q(s,a) $\leftarrow$ *average(Returns(s,a))*
(c) Foreachsintheepisode:
$a^* \leftarrow argmax_a\ Q(s,a)$

$$For\ all\ a \in A(s),\ \pi(s,a) \leftarrow \begin{cases} 1-\varepsilon + \frac{\varepsilon}{|A(s)|} & if\ a = a^* \\ \frac{\varepsilon}{|A(s)|} & if\ a \neq a^* \end{cases}$$

Where:*A(s) is the set of available actions at state s.*

Figure 3.The first visit e-greedy on-policy MC method

Where a is a delegation action, R(*s*,*a*) is the MDP termination cost; $P_i(s' + |s,a|)$ is the transition model; $V_j(s')$ is the estimated optimal value function for the next state at agent ni and $D_i(s' + |s,a|)$ is the estimated connection cost to the next state.

4. Routing Problem in Mobile Ad-hoc Networks as a Reinforcement Learning Task

The application of reinforcement learning to routing problem in MANETs is relatively a young research field. Actually, only few RL-based routing protocols for MANETs can be found in the literature. Moreover, only a subset of them deals directly with routing as a reinforcement learning task. In fact, works like [3] and [8]exploit the reinforcement learning framework to learn some routing-protocol parameters rather than to fix them experimentally. In this section, we focalize on routing protocols which learn to make routing decisions (choosing next-hop or path for routing) via reinforcement learning. We have selected works that give the explicit formalization of the routing problem as a MDP or a POMDP.

Indeed, the first demonstration that network packet-routing can be modelized as a reinforcement learning task is the Q-routing protocol [9] proposed for fixed networks. The authors claim, in the paper [9], that a packet-routing policy answers the question: to which adjacent node should the current node send its packets to get as quickly as possible to its eventual destination? Hence, the routing problem can bemodelized as a RL-task where the environment is the network, the states are the nodes and the learning agent's possible actions are the next-hops it can take to reach the destination.

4.1 Mobility Aware Q-routing

In reference [10], straightforward adaptation of Q-routing to the context of MANETs was proposed. Indeed, the same RL-model of Q-routing is maintained always in the perspective of optimizing packets delivery time:

1) States: In a very intuitive way the set of states is the set of mobile nodes in the network.

2) Actions: At a node *x*,available actions are reachable neighbors. A node learns how to choose a next-hop to forward its traffic.

3) Reward: Local information is used to update the routing policy of a source node *x*. This includes the min Q-values of its neighbors and the time the current packet spent on the queue, b_t^x ,at node *x* before being sent off at period time t as shown in equation (2), where $0 < \alpha < 1$ is the learning rate:

$$Q_t^x(d,y) = (1 - \alpha)\, Q_{t-1}^x(d,y) + \alpha\,(b_t^x + \min_z Q_{t-1}^y(d,Z)) \qquad (2)$$

4. Action selection rule- a greedy policy is adopted. When a node *x* receives a packet for destinationd, it sends the packet to the neighbor *y* with the lowest estimated delivery time $Q_t^x(d,y)$.

To take care of nodes mobility, two additional rules are proposed for Q -values updates of neighboring nodes: $Q^x(d,y) = \infty$ when

y moves out of *x* range; and $Q^x(d,y) = 0$ when *y* moves into *x* range.Note that the second update rule is made optimistic to encourage exploration of new coming neighbors.

Finally, we note that the same model described above was used in LQ-routing protocol [11]. This latter have introduced the path-lifetime notion to deal with mobility.

4.2 SAMPLE Routing Protocol

In SAMPLE[1], the optimization goal is to maximize network throughput. The routing problem was mapped onto a MDP problem as follows:

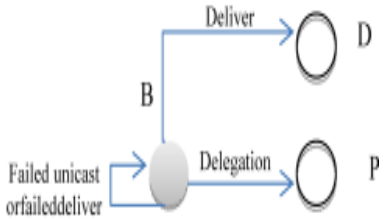

Figure 4. Sample MDP

1) States: Each node n_i has a set of states S_i={B, P, D}where B indicates that a packet is in a buffer waiting to be forwarded (start state), P indicates that a packet has been successfully unicast to a neighbor, and D indicates that a packet has been delivered at node n_i (terminal state).

2) Actions: The actions available at different states in the MDP are a packet delivery action, a broadcast action to discover new neighbors, links, and routes; and for each neighboring node, a delegation action (unicast).

3) Transition model-a statistical transition model that favors stable links is considered. It acquires information about the estimated number of packets required for a successful unicast as an indication of links quality. In order to build this model, a number of different system events are sampled within a small time window into the past. The monitored events are: attempted unicast transmissions; successful unicast transmissions; received unicast transmissions; received broadcast transmissions; and promiscuously received unicast transmissions.

4) Reward model:a simple static reward model was considered. The rewards are set at values -7 and 1 to model the reward when transmission succeeds under a delegation action and fails, respectively. In effect, these values reflect connection costs in IEEE.802.11 protocol.

5) Action selection rule: Concerning delegation actions, the decision of which next hop to take is chosen probabilistically using the Boltzmann-action selection rule. Variation of temperature value controls the amount of exploration. Furthermore, SAMPLE also uses a simple greedy heuristic in order to restrict exploration to useful areas of the network by only allowing a node to forward to those neighboring nodes with a value that is less than its function value. The discovery action is also allowed with a certain probability in order to explore new routes.

To deal with nodes mobility, authors in [1] have considered Q-values to decay. They suggest configuring the decay-rate to match any estimated mobility model. Note that,the CRL algorithm was used to solve the RL problem of SAMPLE.

One critic that can be made about SAMPLE MDP is its static reward model. This latter limitsSAMPLE applicability to MANETs deploying IEEE.802.11 at the MAC layer.

Indeed the same RL-model of SAMPLE was used in SNL-Q routing protocol [2]. However, the authors claim that is a POMDP model which is not correct. We believe that authors confused the probabilistic transition model with the observations setwhich we found in POMDPs.

4.3 RL-based Secure Routing

Finding secure routes has made the main focus of the RL-based routing protocol proposed in [4]. Learning a policy for selecting neighbors based on their reputation values was mapped to a MDP as follows:

1) States: The state of any source node encompasses the reputation values of its neighbors.

2) Actions: The decision maker should select one among its neighbors to for routing.

3) Reward: The destination node can be reached via several paths. Hence, a reward of +1 is assigned to every node on all discovered paths. Otherwise no reward is assigned.

What is notable is the simplicity of the proposed model. However, authors have fixed the number of neighboring nodes to four which is an unrealistic hypothesis. Concerning RL-algorithm, the ONMC method was adopted. The authors justify their choice by the episodic nature of the routes discovery process which starts when a source node initiates a route discovery and terminates when the destination node is found or when a maximum number of hops is reached.

4.4 Selfishness and Energy aware RL-based Routing Protocol

In [12], the routing problem is mapped into a partially observable MDP as follows:

1) State and observation spaces: A node state is a vector of its one-hop-neighboring nodes parameters. Those parameters can be about congestion level, selfishness, remaining energy, etc. The overall state space is the *m* x (*n*-1) dimensional unit cube, where n is the number of nodes in the network and m is the number of considered parameters. However, those parameters are usually unknown to the decision-maker node. To deal with this partial observability, a source node derives estimates about the values from past experiences with its neighboring nodes. Note that, only energy and selfishness parameters were considered in the experiments.

2) Actions : *the actions space* A is a n-1dimensional simplex spanned by the vectors $g_{max}e_j$. where n is the number of nodes in the network, g_{max} an upper bound on the number of packets a node can process during a single time step and e_j is the unit vector along the j^{th} coordinate axis.

3) Reward: A non-linear reward function was used, defined by :

$$r(a(t),s(t)) = \sum f^t(\hat{\Theta}_j(t))\,(\alpha, a_{ij}(t) - exp(a_{ij}(t).C)) \qquad (3)$$

Where: f^t is the learned controller at time step t;α is a predefined constant; C the energy needed to send a packet;α_{ij} (t) the number of packets send by node *i* through node *j* at time *t* and $\hat{\Theta}_j$ (t) is the current estimate of node *j* parameters values.

4) State transitions: To update energy and selfishness estimates, two learning rate were used: α_w when wining and α_l when losing. When the ratio between the number of packets forwarded by a node *j* and the number of packets sent to node *j* is greater than the corresponding estimated value than the node is winning otherwise it is losing.

5) Action selection rule: When a source node needs to make decision it calculates the value of the controller for all nodes in the set of one hop neighboring toward a destination d, given the current nodes parameters estimates. Then, it selects the greedy action i.e. the node that is most likely to forward the packets with probability $1-\varepsilon_t$ and a randomly selected node, different from the greedy choice, with probability ε_t where $\varepsilon_t = 1/t$.

In this work, a stochastic gradient descent based algorithm that allows nodes to learn a near optimal controller was exploited.This controller estimates the forwarding probability of neighboring nodes. Roughly speaking, the idea behind policy search by gradient is to start with some policy, evaluate it and make an adjustment in the direction of the empirically estimated gradient of the aggregate reward, in order to obtain a local optimum policy [13].

4.5 RL-based Energy-Efficient Routing

The RL -based routing protocol presented in [14] has two contrasting optimization objectives, namely: maximizing network lifetime and minimizing energy consumption in MANETs. The routing problem was mapped into a MDP as follows:

1) States: the state space of a source node is given by $S=\{s|s=[P_{l_e}(i), B_{l_b}(j)], 1 \le i \le n, 1 \le j \le m\}$ where $P_{l_e}(i)$ and $B_{l_b}(j)$ denotes the quantized energy and battery network levels, respectively.

2) Actions: the decision maker should choose a path. The action space includes three actions, namely: minimum-energy routing path "*a*(1)",the max-min routing path "*a*(2)" and the minimum cost routing path"*a*(3)". Hence $A=\{a \mid a=[a(1),a(2),a(3)], a(j) \in \{0,1\} \Sigma a(j)=1\}$ whereselecting a path is indicated by attributing 1 to the corresponding component.

3. Cost structure[1] : Once the source node selects an action at a given state, the following cost incurs: $c(s,a) = (P_l)^{x1} (B_l)^{-x2}$

[1]Since the optimization goals are minimization problems, we talk about cost rather than reward.

$(B_{init})^{x3}$, where:B_{init} is the initial level of battery assumed to be constant for all nodes; x1,x2,x3 are weight factors empirically fixed to 1, B_l and P_l are ,respectively, the battery bottleneck and the energy consumption along the path l.

	MDP	POMD	Model Based	Model Free	TD algorithm	MC algorithm	Stochastic Gradient descent
MQ-routing[10]	√			√	√		
LQ-routing[11]	√			√	√		
SAMPLE[1]	√		√		√		
SNL-Q routing[2]	√		√		√		
RL-based Secure routing[4]	√			√		√	
Selfishness and based energy aware RL routing[12]		√	√				√
RL based Energy-efficient routing[14]	√			√		√	

Table1. Comparisonof RL-based routing protocols described in Section 4

The authors have applied the ONMC method. They justify this choice by the episodic nature of messages-exchange that starts when sending a message and terminates when it reaches its destination.Note that, the e-greedyactions-selection rule was applied.

5. Conclusions

In this paper, we have selected some representative works from the literature of RL-based routing protocols for MANETs. We have described various MDP/POMDP models that formalize the routing problem with different optimization goals including quality of service (delay and/or throughput), security and energy constrained routing. We can summarize thefundamental issues that arise when dealing with routing problem as a RL task in MANETs as follows:

1) Model-free Vs Model-based: All described works in the previous section are model-free apart from protocols [10] and [12] which are model-based. The authors have indicated, in [12], that model-based RL is more appropriate in environments where acquiring real-word experiences is expensive. However, model-based methods are characterized by slower execution times. This is, in fact, problematic in the case of real-time applications with strict response-time constraints. Besides, cost of constant probing incurred by model-free RL methods in terms of routing overhead may degrade the routing protocol efficiency. Therefore, a deep analysis of the convergence time and efficiency compromise is needed.

2) MDP vs POMDP: In fact, uncertainty is more pronounced in MANETs due to their very dynamic nature. However, all works already presented in the previous section, apart from [12], neglect this fact and consider the environment state as completely observable. We believe that an accurate model that really reflect MANETs features should deal explicitly with partial observability.

3) Exploitation versus exploration: The tradeoff between exploration and exploitation is well known as a challenging issue in any reinforcement learning algorithm. However, we believe that in presence of mobility the network can be considered somewhat auto-explorative. We think that the relationship between mobility, exploitation and exploration must be investigated.

4) RL algorithms: Except, the policy search by gradient applied in [12],RL algorithms used to solve the RL problems described in the previous section are either TD or MC methods. However, we do not know which approach is more efficient for MANETs and under which circumstances. We believe that comparative studies in this sense will be of significant interest.

References

[1] Dowling, J., Curran, E., Cunningham, R., Cahill, V. (2005). Using feedback in collaborative reinforcement learning to adaptively optimize MANET routing, *IEEE Trans. Syst. Man, Cybern.* 35, p. 360–372.
[2] Binbin, Z., Quan, L., Shouling, Z. (2010). Using statistical network link model for routing in ad hoc networks with multi-agent reinforcement learning. International Conference on Advanced Computer Control, p. 462 – 466
[3] Usaha, W., Barria, J.A. (2004) A reinforcement learning Ticket-Based Probing path discovery scheme for MANETs, *Ad Hoc Networks Journal*, 2, p.319-334.
[4] Maneenil, K., Usaha, W. (2005). Preventing malicious nodes in ad hoc networks using reinforcement learning, *In*: The 2nd International Symposium on Wireless Communication Systems, p. 289-292 Italy.
[5] Sutton , R., Barto, A. (1998). Reinforcement learning. MIT Press.
[6] Watkins, C. J. (1989). Learning with Delayed Rewards. PhD thesis, Psychology Department, University of Cambridge, UK.
[7] Dowling, J., Cunningham, R., Harrington, A., Curran, E. , Cahill, V. (2005). Emergent consensus in decentralised systems using collaborative reinforcement learning. *In*: Self-Star Properties in Complex Information Systems, LNCS, V. 3460, p.63-80. Springer,Verlag.
[8] Usaha, W., Barria, J A. (2007). QoS Routing in MANET with Imprecise Information Using Actor-Critic Reinforcement Learning IEEE Wireless Communications and Networking Conference, p.3382-3387 Hong Kong.
[9] Boyan, J. A., Littman, M.L. (1994). Packet routing in dynamically changing networks: A reinforcement learning approach. Advances *In*: Neural Information Processing Systems 6, p. 671-678.
[10] Chang, Y.-H. , Ho, T. (2004). Mobilized ad-hoc networks: A reinforcement learning approach. *In*: ICAC '04: Proceedings of the First International Conference on Autonomic Computing, IEEE Computer Society, p. 240–247, USA.
[11] Tao T., Tagashira, S., Fujita S. (2005). LQ-Routing Protocol for Mobile Ad-Hoc Networks, *In*: Proceedings of the Fourth Annual ACIS International Conference on Computer and Information Science.
[12] Nurmi, P. (2007). Reinforcement Learning for Routing in Ad Hoc Networks, *In*: Proc. 5th Intl. Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks , IEEE Computer Society.
[13] Peshkin, L. (2001). Reinforcement Learning by Policy Search. PhD thesis, Brown University.
[14] Naruephiphat, W., Usaha, W. (2008). Balancing tradeoffs for energy-efficient routing in MANETs based on reinforcement learning. The IEEE 67th Vehicular Technology Conference Singapore.