Throughput Maximization for Energy Efficient Multi-Node Communications using Actor-Critic Approach

Charles Pandana and K. J. Ray Liu
Department of Electrical and Computer Engineering University of Maryland, College Park, MD 20742
Email: cpandana, kjrliu@glue.umd.edu

***Abstract*— We investigate the problem of average throughput maximization per total expended energy in packetized multi-node wireless sensor communications. We consider a CDMA like multiple access scenario where multi transmitters are communicating with one receiving node. Since the transmitting nodes are typically distributed in different location for wireless sensor applications, a good transmission strategy should be employed in each node to maximize throughput per unit energy in the node. We propose to use a distributed Actor-Critic algorithm to *independently* learn a good transmission strategy for each transmitting node. The resulting strategy adapts to the incoming traffic rate, buffer condition, and channel condition, which is influenced by both the location of the transmitting node and the other nodes' transmission strategy. Our proposed method achieves 1.5 to 6 times throughput per energy compared to the simple constant signal to interference ratio (CSIR) policy, particularly in high packet arrival. Moreover, the proposed algorithm is robust in tracking packet arrival rate variation.**

I. Introduction

Advances in sensor and wireless communication technologies have made possible the large scale deployment of wireless sensor networking. These wireless sensor networks have found themselves a lot of important applications, such as environment habitat monitoring, health care monitoring, battlefield surveillance and maintenance of modern highway, manufacturing and complex system. One crucial characteristic of these networks is to have a very long network lifespan, since human intervention for energy supply replenishment may not be possible. In order to achieve long network lifespan, a highly energy efficient resource management is a must. In these applications, the traditional low power design, focusing mainly on circuits and systems has been shown inadequate [1]. The stringent energy requirement calls for the realization of energy aware communication system, which reconfigures transmission parameters from different communication layers to maximize the energy efficiency [2]. Such a cross-layer optimization can be realized by employing an optimal control agent that interacts with different communication layers and dynamically reconfigures each layer's parameters.

There exist several literatures that focus on the wireless resource management. In [3] [4], power control scheme for packet networks is formulated using dynamic programming (DP). In these schemes, the power control follows the threshold policy that balances the buffer content and the channel interference. In [5], the DP formulation for power control with imperfect channel estimation is addressed. They show that the dynamic programming solution is better than the constant signal to interference ratio (CSIR) approach. Joint optimized bit-rate and delay control for packet wireless networks has also been studied within DP framework [6]. Most of the literatures based on DP framework consider point-to-point communication and assume the knowledge of the exact probability model, hence the optimal solution is obtained using dynamic programming computational methods [7]. In practical system, the probability model is very hard to obtain, if not impossible, especially in a complex system involving the interaction of multi nodes. In this complex situation, the strategy employed by one transmitter will be influenced by the strategies employed by the others. And the optimal solution is very hard to obtain. This motivates us to develop and investigate a distributed stochastic optimization scheme that learns a good policy from sample path realization. The proposed algorithm is based on the *independent* learning in each of the nodes.

In this paper, we focus on the average throughput maximization per total consumed energy in multi-node wireless sensor communications. In particular, we formulate the optimization problem as a multi-agent Markov Decision Process (MDP) [8]. In this framework, an intelligent control agent resides in each of the node and the objective of the agent is to obtain the modulation and transmit power to maximize its own average throughput per total consumed energy. We propose to extend the single agent Reinforcement Learning (RL) [9] algorithm called Actor-Critic (AC) algorithm to solve the posed multi-node optimization problem. The resulting algorithm *independently* learns the control policy to maximize the throughput in each node. Compared to the simple CSIR policy, where the transmitter chooses the highest modulation possible while maintaining the predefined link signal-to-interference ratio (SIR) given one particular modulation, our proposed method achieves more than two times throughput, especially at high packet arrival rate and for nodes that far away from the receiver.

The rest of this paper is organized as follows. In the

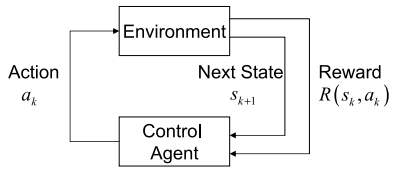


Fig. 1. Interaction between agent and environment in MDP

next section, we briefly review the definition of single-agent MDP and the Actor-Critic (AC) algorithm. In section III, We propose a simple extension of AC algorithm for multi-agent MDP. The formulation of throughput maximization per total consumed energy in multi-node communications is presented in section IV. Simulation results are given in section V. Finally, the conclusions are drawn in section VI.

II. Single-agent Markov Decision Process and Actor Critic Algorithm

A single-agent MDP [8] is defined as a $(\mathbf{S}, \mathbf{A}, \mathbf{P}, \mathbf{R})$ tuple where $\mathbf{S}$ is the state space that contains all possible states of the system, $\mathbf{A}$ is the set of all possible control actions from each state, $\mathbf{P}$ is a transition function $\mathbf{S} \times \mathbf{A} \times \mathbf{S} \rightarrow [0, 1]$, and $\mathbf{R}$ is a reward function $\mathbf{S} \times \mathbf{A} \rightarrow \mathbf{R}$. The transition function defines the probability distribution over the next state as a function of the current state and the agent's action, i.e. $[\mathbf{P}]_{s_k, s_{k+1}}(a_k) = P_{s_k, s_{k+1}}(a_k)$ specifies the probability of transition from state $s_k \in \mathbf{S}$ to $s_{k+1} \in \mathbf{S}$ under control action $a_k \in \mathbf{A}$. Here, the notation $[\mathbf{A}]_{i,j}$ denotes the element on the i^{th} row and the j^{th} column of matrix $\mathbf{A}$. The transition probability function $\mathbf{P}$ describes the dynamics of the environment as a response to the agent's decision. The reward function specifies the reward incurred at state $s_k \in \mathbf{S}$ under control action $a_k \in \mathbf{A}$. The interaction between the agent and environment in MDP is illustrated in Figure 1. At time k, the control agent detects $s_k \in \mathbf{S}$ and decides an action $a_k \in \mathbf{A}$. The action a_k causes the state to evolve from s_k to s_{k+1} with probability $P_{s_k, s_{k+1}}(a_k)$ and reward $R(s_k, a_k)$ corresponding to the agent's action will be obtained.

The solution of the MDP consists of finding the decision policy $\pi : \mathbf{S} \rightarrow \mathbf{A}$ so as to maximize the objective function. In this paper, we focus on the average reward per stage as follow

$$\rho^{\pi}(s_0) = \lim_{n \rightarrow \infty} \frac{1}{n} E_{\pi} \left[\sum_{k=0}^{n-1} R\Big(s_k, \pi(s_k)\Big) \right], \quad s_k \in \mathbf{S}, \pi(s_k) \in \mathbf{A}, \tag{1}$$

where $\rho^{\pi}(s_0)$ is the average reward obtained using decision policy π when the initial state is s_0. Since we are interested in maximizing the average throughput per total consumed energy, this criterion exactly describes our objective function. We note that the expectation operation in (1) is a conditional expectation given a particular policy. The optimal policy is the decision rule that maximizes the average reward per stage ρ^{π} over all possible policy π.

The optimal solution of single-agent MDP can be obtained by solving the Bellman's equation [7]

$$\rho^* + h^*(s) = \max_{a \in \mathbf{A}(s)} \left[R(s, a) + \sum_{s'=1}^{|S|} P_{s,s'}(a) h^*(s') \right], \quad \forall s \in \mathbf{S}, \tag{2}$$

where ρ^* is the optimal average reward per stage and $h^*(s)$ is known as relative state value function for each state s. When the state transition probability is not available, the Reinforcement Learning (RL) [9] algorithm provides a systematic way to solve the MDP. The Actor-Critic (AC) algorithm, one type of RL algorithm is shown in Table I. Notice that α, β and ϵ determine the learning rate of the state value function, average reward per state and actor preference, respectively [9].

TABLE I
Actor-Critic Algorithm

Actor-Critic Algorithm

Initialize α, β, ϵ, $k = 0$, $h(s_k) = 0$ for all $s_k \in \mathbf{S}$, and $\rho_k = 0$.
Set preference function $p(s, a) = 0, \quad \forall s \in \mathbf{S}, \forall a \in \mathbf{A}(s)$.
Set s_0 arbitrarily.
Loop for $k = 0, 1, 2, \ldots$
1. Choose a_k in s_k according to Gibbs softmax method:
$\pi(s_k, a_k) = Pr(a_k = a | s_k = s) = e^{p(s,a)} / \sum_b e^{p(s,b)}$
2. Get Reward from current decision and observe next state s_{k+1}:
$r = R(s_k, a_k)$
3. Evaluation of temporal difference (error)
$\delta = r + h(s_{k+1}) - h(s_k) - \rho_k$
4. Update relative state value function and average reward per state
$h(s_k) = h(s_k) + \alpha\delta$
$\rho_{k+1} = \rho_k + \beta\delta$
5. Update actor preference
$p(s, a) = p(s, a) + \epsilon\delta$
End Loop.

III. Simple Extension of AC algorithm to multi-agent MDP

The single-agent MDP shown in Figure 1 can be generalized to more complex multi-agent case, where several agents simultaneously learn their environment and each of the agent's action influences the evolution of the other agents' state. The multi-agent MDP scenario is shown in Figure 2. At time k, the i^{th} control agent detects its state s_k^i and decides action a_k^i. Simultaneously, other agents detect their states and choose their actions. The decisions from each agent, $a_k^1, a_k^2, \cdots, a_k^n$ cause the i^{th} agent's state to evolve from s_k^i to s_{k+1}^i with probability $P_{s_k^i, s_{k+1}^i}(a_k^1, a_k^2, \cdots, a_k^n)$. Corresponding to its decision, the i^{th} agent obtains its reward $R^i(s_k^i, a_k^i)$. We note that the major difference of multi-agent MDP with the single-agent MDP is that the state evolution in each agent depends on all of the actions from the interacting agents. Due to this mutual coupling of each agent's decision, the problem becomes very complicated. And the optimal decision for each agent depends on the strategy employed by other agents.

To solve the posed problem, we propose to extend the single-agent AC algorithm in Table I to independently learn

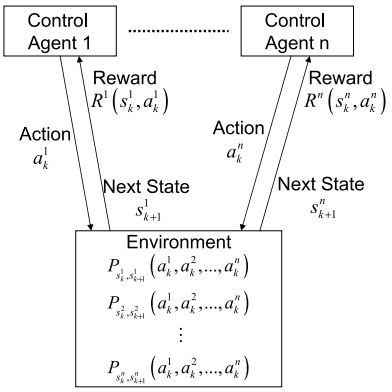


Fig. 2. Interaction between agents and environment in multi-agent MDP

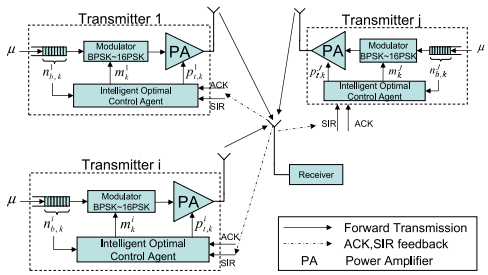


Fig. 3. Interaction of nodes with distributed optimal control agent

policy in each agent. In this independent learning, each agent learns its transmission strategy by assuming that the agent itself is the only agent that influences the evolution of its state. Each agent uses only its local state information to do the decision, that is the agent does not take into account the state, action and reward involved in other agents' decision making process. Although the proposed independent AC learning is not optimal, but it has several advantages. First, since no global information (the state and the decision of other agents) is used in the learning process, less control handshaking (each node doesn't need to exchange its state information and the decision employed) is required. Second, the simple extension of AC algorithm has the same computational complexity as the single-agent scenario. Each agent requires only to update the relative state value function, average reward and actor preference function of the occurring state and action (Table I). Third, the proposed algorithm works very well compared to the CSIR policy as discussed in section V.

IV. Throughput Maximization in Multi-Node Communication

In this section, we formulate the average throughput maximization per total consumed energy in multi-node wireless sensor networks as an MDP as illustrated in Figure 3. We assume that the receiving node has enough receiver sets in communicating with different transmitters. We study the transmission strategy that chooses the modulation and transmit power to maximize the average throughput (1) while adapting to the channel condition and transmitter buffer in each transmitter. In the following, the channel experienced by each node and the reward function are explained. The channel model forms the multi-agent MDP scenario.

Suppose there are N nodes that want to simultaneously communicate with the receiver, the SIR of link i can be expressed as [10]

$$\Gamma^i(p_t^1, \cdots, p_t^N) = \frac{W A_t^i p_t^i}{R(\sum_{j \neq i} A_t^j p_t^j + \sigma^2)}, \tag{3}$$

where W and R are the system bandwidth and transmission rate, p_t^i is the transmission power employed by node i, A_t^i is the path loss corresponding to link i, and σ^2 is the thermal noise. The path loss A_t^i depends on the distance between the transmitter i and the receiver, that is $A_t^i = c/(d^i)^4$, where d^i is the distance between the transmitter i and the receiver. Equivalently, (3) can be written in dB as

$$\Gamma^i(\eta^i, p_t^i)dB = 10\log_{10}\left(\frac{W p_t^i}{R}\right) - \eta^i, \tag{4}$$

where $\eta^i = 10\log_{10}\left(\frac{\sum_{j \neq i} A_t^j p_t^j + \sigma^2}{A_t^i}\right)$ is the equivalent interference of link i. From (4), it is obvious that transmit power employed by other nodes influences the link quality of node i.

In the application of wireless sensor networks, the throughput and energy consumption are two critical parameters. We employ the number of packets that successfully transmitted per total energy consumption as our reward function. Similar to [10], we employ reward function for each node as follow

$$R^i((n_b^i, \eta^i), (m^i, p_t^i)) = \begin{cases} \frac{L_b}{L}\frac{R \cdot m^i \cdot S^i(\Gamma^i, m^i)}{L \cdot (p_t^i + f(n_b^i))} \times 10^{-3} & \mathrm{n_b^i \neq 0, p_t^i \neq 0} \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

The expression (5) has unit packets/mJoule. n_b^i and η^i denote the number of packet queued in the buffer and the link interference, m^i and p_t^i denote the modulation level and transmit power. L_b indicates the number of bits in one packet, L is the number of bits after adding error detecting code. R bits/s is the system transmission rate. $f(\cdot)$ models the buffer processing power. Including the buffer processing energy will minimize the possibility of buffer overflow, that is as the packets queued in the buffer become larger, the reward function becomes smaller and the agent will exercise all actions to increase the throughput. $S^i(\Gamma^i, m)$ denotes the packet correct reception probability, Γ^i is the effective link SIR as in (4). The superscript i corresponds to the i^{th} agent. We assume the buffer processing cost is linear function of packets queued in the buffer [3]. The buffer cost, packet error probability expression and other parameters are summarized in Table II.

Our optimization objective is to maximize average reward per stage (1) in each node, where the reward function is described as in (5). Comparing (1) and (5), we see that the i^{th} agent's state is described as the aggregate of buffer content

and link interference experience by each node, i.e.: $s^i \equiv (n_b^i, \eta^i)$. The control space consists of modulation level and transmit power, $a^i \equiv (m^i, p_t^i)$. The above formulation has the following interpretation, before a packet transmission, the i^{th} transmitter is in some state s_k^i, the transmitter uses this information to determine the modulation and power to maximize the average throughput per total consumed energy. At the end of a packet transmission, each transmitter obtains feedback from receiver containing the quantized estimated its link quality η_k^i and ACK/NACK. We assume this feedback information can be obtained by the transmitter without any error. If ACK is received then the transmitter will send the following packet at the next transmission. Otherwise, it has to retransmission the packet. The feedback information is used by the agent to update its state. We note that each agent's action mutually influences other agent's state as in multi-agent MDP.

Our proposed independent AC learning scheme acts as follows. Initially, every node will be initialized as in Table I. The agent decides the modulation and transmit power and obtains the reward (5). This reward is used to update relative state function, average reward per stage and the actor preference, which will be used to make the decision in the next time instant. In making the decision, the agent explores all possible actions that result in different evolution of the environment and select the policy that maximizes the throughput per unit energy. After taking the action, the agent updates its state based on the feedback information. Every node independently learns a good transmission policy in a distributed manner.

V. Numerical Results

To assess the performance of the proposed multi-node learning algorithm, we construct the simulation using parameters shown in Table II. We simulate the multi-node system with 3 nodes communicating with one receiver. The location of the transmitting nodes are 340, 460, 570 meters away from the receiver. Node 1 is the nearest node to the receiver and node 3 is the farthest node from the receiver. For comparison purpose, we simulate the greedy CSIR scheme. In this scheme, every transmitter tries to transmit at the highest throughput possible maintaining a fixed link SIR. The transmitter always chooses BPSK to transmit when there is only one packet in the queue; when there are 2 packets queued, the transmitter chooses QPSK without considering BPSK. Similarly, when there are more than 4 packets queued in the buffer the transmitter will always use 16PSK. For each modulation, the transmitter selects power level to achieve a constant link SIR. We use (7, 11, 16, 22) dB as the targeted link SIR for BPSK to 16PSK, respectively. These link SIRs can achieve more than 90% of packet correct reception probability.

The AC algorithm is initialized with $\alpha = 0.05$, $\beta = 0.0005$ and $\epsilon = 0.01$. The learned control policies for different nodes for $\mu = 2.0$ is shown in Figures 4. In these figures, the larger the channel interference implies the worse channel, the buffer content refers to the number of packets queued in the buffer. It is clear from the figures, the interference range experienced by each node is different due to the distance from the receiver. In each node, when the channel is good, the agent tend to choose higher modulation level with suitable power. When the channel is worse, the agent tends to use lower modulation level. For the same interference level, the agent tends to use higher modulation when buffered packets is larger, provided the required transmit power is in the allowed range. We note that the AC algorithm jointly decides the modulation level and transmit power adapting to the buffer condition and interference level. Comparing the policies in node 3 and node 1, we observe that node 3 tends to use higher transmit power to compensate the farther distance. At the same time, node 3 (the farthest node) will not use as high modulation level as node 1 when the buffer content become large to guarantee the correct reception. In this sense, we argue that AC algorithm learns the control policies that not only balance the incoming traffic rate and buffer condition, but also adapt to the distance and link quality in each node.

TABLE II
Simulation parameters

Packet size	$L_b = 64$, $L = 80$
System Parameters	$W = 10$Mhz, $R = 100$kbits/s, $T_p = 0.8$ms, $\sigma^2 = 5 \times 10^{-15}$W
Channel Model	d=[320,460,570] m, $A_t^i = 0.097/(d^i)^4$, $\sigma^2 = 5 \times 10^{-15}$W
Buffer Cost	$f(n_b) = 0.05(n_b + 4)$ if $n_b \neq \max(n_b)$, $\max(n_b) = 15$, $f(\max(n_b)) = 3$
modulation level	m=1,2,3,4 (BPSK,QPSK,8PSK,16PSK),
Packet success probability	$S(\Gamma, m)$=$(1 - P(\Gamma, m))^L$ $P(\Gamma, m)$=$erfc(\sqrt{\Gamma} * \sin(\frac{\pi}{2^m}))$
Transmit power	$p_t = [0, 0.2, ..., 2]$ Watt
SIR range	$\Gamma = [0, 1, ..., 24]$ dB
Quantized Interference	$\eta = [-16, -15, ..., 14]$ dB

Figure 5 shows the average throughput learned by the AC algorithm and CSIR policy for packet arrival rate $\mu = 2.0$. It is obvious that in both policies, the node nearer to the receiver will effectively have higher packet throughput per energy, since it requires less energy for achieving the same throughput. Figure 6 shows the throughput that can be achieved for the AC learned policy and the CSIR policy for various packet arrival rate. From this figure, the CSIR and AC policies achieve similar throughput when the packet arrival rate is low ($\mu \leq 1$). But when the packet arrival rate becomes large, the AC algorithm can achieve higher throughput compared with CSIR policy. In particular, the AC algorithm achieves 1.5 more throughput for node 1 when $\mu = 3.0$ and it achieves 6.3 and 7.1 times throughput respectively for node 2 when $\mu = 3.0$ and node 3 when $\mu = 2.0$. We note that the CSIR policy in node 3 is not able to transmit anything for the packet arrival rate beyond $\mu = 2.0$. In this situation, the energy in node 3 will be completely wasted without able to transmit anything. Using

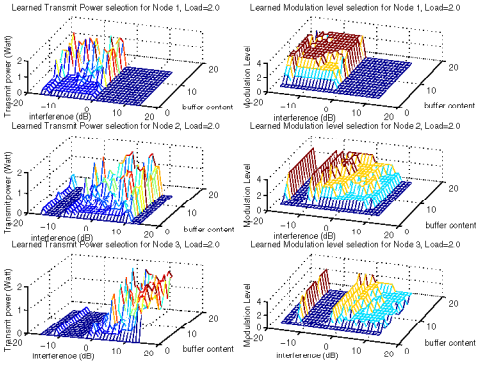
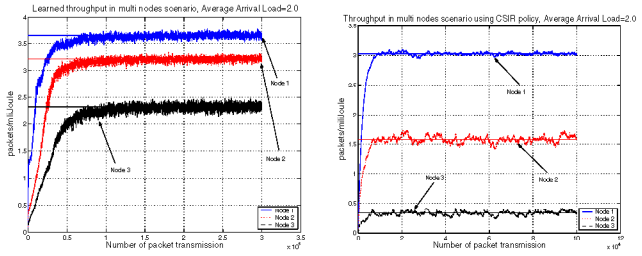


Fig. 4. Learned policies, packet arrival load $\mu = 2.0$



(a) Learned average throughput, packet arrival load $\mu = 2.0$

(b) CSIR throughput, packet arrival load $\mu = 2.0$

Fig. 5. Learned and CSIR throughput per unit energy for packet arrival load $\mu = 2.0$

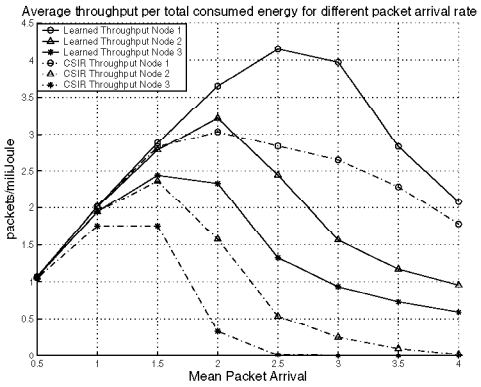


Fig. 6. Average throughput corresponding to different packet arrival load $\mu = 2.0$

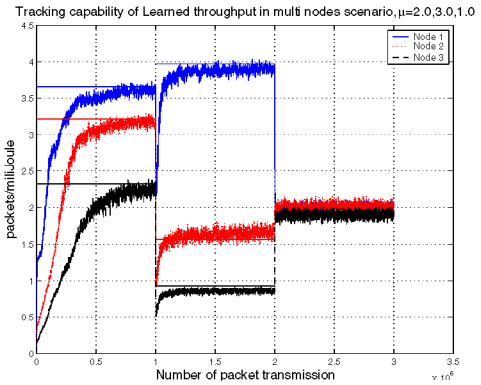


Fig. 7. Learned average throughput adapting to different packet arrival load μ=(2.0, 3.0, 1.0)

the AC algorithm, each node in the network is able to achieve higher throughput per unit energy for broad packet arrival rate. This is due to the fact that the AC algorithm has the ability to explore policies other than the greedy policy adapting to the channel condition and packet arrival rate. The greedy policy will obviously result in total breakdown of the network. Figure 7 shows the ability of the learning algorithm to track the different packet arrival rate. In this figure, the packet arrival rate varies as μ=(2.0, 3.0, 1.0). Based on the sample realization, the proposed algorithm adjusts the policy adapting to different packet arrival rate and the channel condition resulting from different location of the transmitters.

VI. Conclusion

We formulate the average throughput maximization per total expended energy in multi-node wireless sensor communications using the multi-agent MDP framework. We propose to independently learn the policy using an extension of single-agent Actor-Critic (AC) algorithm. Since the AC algorithm has the capability to learn and adapt to the channel condition, the learned policy can achieve more than 2 times throughput compared to the CSIR policy, particularly in high packet arrival rate and for node that is farther away from the receiver. Moreover, the algorithm is robust to track variation in packet arrival rate. Our study also indicates that the proposed scheme with learning capability provides a simple, online and distributed algorithm to achieve highly energy efficient scheme for wireless sensor network.

References

[1] W. Stark, H. Wang, A. Wrothen, S. Lafortune, and D. Teneketzis, "Low-energy wireless communication network design," *IEEE Wireless Comm.*, pp. 60–72, August 2002.

[2] A. J. Goldsmith and S. B. Wicker, "Design challenges for energy-constrainded ad hoc wireless networks," *IEEE Wireless Comm.*, pp. 8–27, August 2002.

[3] N. Bambos and S. Kandukuri, "Power controlled multiple access (PCMA) in wireless communication networks," *IEEE INFOCOM*, pp. 386–395, 2000.

[4] ——, "Multimodal dynamic multiple access in wireless packet networks," *IEEE INFOCOM*, 2001.

[5] T. Holliday, A. Goldsmith, and P. Glynn, "Wireless link adaptation policies: Qos for deadline constrained traffic with imperfect channel estimates," *IEEE ICC*, 2001.

[6] J. Razavilar, K. J. R. Liu, and S. I. Marcus, "Jointly optimized bit-rate/delay control policy for wireless packet networks with fading channels," *IEEE Trans. on Comm.*, pp. 484–494, March 2002.

[7] D. Bertsekas, *Dynamic Programming and Optimal Control*. Belmont, MA: Athena Scientific, 1995.

[8] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*. Norwell, MA: Kluwer Academics Publisher, 1999.

[9] R. S. Sutton and A. G. Barto, *Reinforcement Learning : An Introduction*. Cambridge, MA: MIT Press, 1998.

[10] D. J. Goodman and N. B. Mandayam, "Power control for wireless data," *IEEE Personal Communications Magazine*, vol. 7, pp. 48–54, April 2000.