A Fuzzy Reinforcement Learning Approach to Power Control in Wireless Transmitters

David Vengerov, Nicholas Bambos, and Hamid R. Berenji, *Fellow, IEEE*

***Abstract*—We address the issue of power-controlled shared channel access in wireless networks supporting packetized data traffic. We formulate this problem using the dynamic programming framework and present a new distributed fuzzy reinforcement learning algorithm (ACFRL-2) capable of adequately solving a class of problems to which the power control problem belongs. Our experimental results show that the algorithm converges almost deterministically to a neighborhood of optimal parameter values, as opposed to a very noisy stochastic convergence of earlier algorithms. The main tradeoff facing a transmitter is to balance its current power level with future backlog in the presence of stochastically changing interference. Simulation experiments demonstrate that the ACFRL-2 algorithm achieves significant performance gains over the standard power control approach used in CDMA2000. Such a large improvement is explained by the fact that ACFRL-2 allows transmitters to learn implicit coordination policies, which back off under stressful channel conditions as opposed to engaging in escalating "power wars."**

***Index Terms*—Actor-critic algorithms, fuzzy reinforcement learning, wireless power control.**

I. Introduction

UNLIKE the more traditional time-division multiple access (TDMA) or frequency-division multiple access (FDMA) wireless networks, the new generation of code-division multiple access (CDMA) networks do not restrict *a priori* the number of users that can share the same wireless channel. As new users are added to the channel, the quality of service of existing users gradually deteriorates due to increased mutual interference.

Transmitter power control algorithms allow users to dynamically share the bandwidth of a wireless channel, optimizing the channel throughput. In addition, intelligent power control allows mobile users to extend their battery life.

Since the previous generation of mobile phones did not have any data transfer capabilities, most of the algorithms considered by the research literature [6]–[10] have focused just on optimizing the signal-to-interference (SIR) ratio, a characteristic requirement for voice-oriented "continuous" traffic. For example, Foschini and Miljanic [6] introduced a simple distributed power control algorithm for maintaining a fixed level of SIR, which was later adapted into the IS-95 standard and then into CDMA2000. As a result, their algorithm also became a standard benchmark in the research literature, against which all new algorithms are compared.

Data traffic is less sensitive to delays than voice traffic, but it is more sensitive to transmission errors. Reliability can be assured via retransmissions, which cannot be used in continuous voice traffic domains. Therefore, delay tolerance of data traffic can be exploited for design of efficient transmission algorithms that adapt the power level to the backlog of packets awaiting transmission as well as to the current interference level in the channel.

The main dilemma that a data transmitter faces in controlling its power is the following. When high interference is observed in the channel, the transmitter recognizes that it will have to spend a lot of power to ensure successful transmission. Therefore, the transmitter might choose to back off, buffer the arriving packets and wait for the interference to subside before transmitting again. However, as the buffer is filling up with newly arriving packets, the queueing delay rises and the chance of buffer overflow increases, which pushes the transmitter to become more power-aggressive in order to reduce its backlog.

When several transmitters are sharing the same channel, interference becomes responsive to transmitter's actions. That is, a power-aggressive transmitter may cause other ones to go into a backoff mode or also become aggressive. Therefore, optimal transmission policies should consider not only the channel dynamics but also patterns in the behavior of other transmitters. Since this information is not known *a priori*, it needs to be learned through direct interaction with the channel or with a simulation model.

The *reinforcement learning* [11] framework can be used for learning optimal behavior policies in situations such as the one described above. To the best of our knowledge, no reinforcement learning algorithms have been applied to the data power control problem, and the only analytical investigations of this problem formulation are due to Bambos and Kandukuri [2]. They have derived an optimal policy for a single wireless transmitter when interference is random and either follows a uniform distribution or has a Markovian structure. In reality, distribution of interference is not known *a priori*, and the analytical solution of Bambos and Kandukuri cannot be applied.

The paper is organized as follows. In Section II we present a centralized dynamic programming formulation of the problem faced by multiple transmitters sharing the same channel. We then show that the DP approach cannot provide even a numerical solution to this problem for realistic sizes of the state space due to the curse of dimensionality. In Section III we present a distributed reinforcement learning algorithm, ACFRL, that

Manuscript received February 3, 2004; revised August 17, 2004. This paper was recommended by Editor J. B. Oommen.

D. Vengerov is with Sun Microsystems Laboratories, Sunnyvale, CA 94086 (e-mail: dive1743@yahoo.com).

N. Bambos is with Stanford University, Stanford, CA 94305 USA (e-mail: bambos@stanford.edu).

H. R. Berenji is with Intelligent Inference Systems Corporation, Mountain View, CA 94035 USA (e-mail: berenji@iiscorp.com).

Digital Object Identifier 10.1109/TSMCB.2005.846001

has a much lower computational complexity and is theoretically capable of solving this problem. In Section IV we describe ACFRL-2—an adaptation of ACFRL, which has a much faster initial convergence and can thus be deployed in real applications. The ACFRL-2 algorithm is designed to work off-line with a system simulator, and after a good enough policy has been learned, it can be deployed in real time and fine-tuned online using ACFRL. The convergence properties of the ACFRL-2 algorithm are analyzed in Section V. In Section VI we describe the simulation setup for our experiments of applying ACFRL-2 to the problem of transmitter power control in wireless networks. In Section VII we present simulation results for the ACFRL-2 algorithm and discuss the implications of these results. Section VIII concludes the paper.

II. Dynamic Programming Formulation

Our formulation of the wireless power control problem follows that of Bambos and Kandukuri [2]. We consider a communication link operating in discrete time, indexed by $n \in \{1, 2, 3, \ldots\}$. The transmitter is equipped with a FIFO queue (buffer) of size B. Let p_{rn} be the power that the rth transmitter uses to transmit a packet during the nth time slot. Let G_{kr} be the power gain (loss) from the transmitter of the rth link to the receiver of the kth one. The interference experienced by rth link at any point of time is given by

$$I_r = \sum_{k \neq r} G_{kr} p_k + \eta_r \tag{1}$$

where $\eta_r > 0$ is the thermal noise power at its receiver node [1].

If the channel interference is I and power p is used to transmit a packet in a time slot, then the packet is successfully received with probability $Pr(success|p, I)$

$$Pr(success|p, I) = 1 - e^{-\frac{p}{I\delta}} \tag{2}$$

where $\delta > 0$, with higher values indicating higher level of transmission noise.

We assume for simplicity that conditional on (p,I), packet transmission events of any transmitter are statistically independent of each other. If transmission succeeds, the packet is removed from the queue and the transmitter attempts to transmit the next packet in the queue. If transmission does not succeed, the packet remains at the head of the queue for later retransmission. We assume that the transmitter is immediately notified at the end of each time slot whether or not the packet was received successfully. The notification message also contains the value of the interference observed at the receiver, which stays constant until the beginning of the next time step.

Finally, we denote by b_{rn} the backlog of transmitter r (number of packets in the transmitter's queue) at time n. When a packet arrives to a full buffer, it gets dropped and a cost L is incurred.

The task of transmitter r is to choose at the beginning of each time step the power p_{rn} to transmit the head packet after observing the current interference level I_{rn} and the current backlog b_{rn}. In each time slot n the rth link incurs a cost which is based on two components: the power cost p_{rn} and the backlog cost b_{rn}. For simplicity, we consider the case of a linear cost function

$$C_{rn} = \alpha p_{rn} + b_{rn}. \tag{3}$$

The objective of transmitter r is to minimize the average cost over all time steps by controlling the powers $\{p_{r1}, p_{r2}, \ldots, p_{rn}, \ldots, p_{rN}\}$ in consecutive time slots.

The problem described above naturally falls within the realm of dynamic programming. We will now analyze the complexity of the dynamic programming solution to the above problem, simplified by assuming only two transmitters, no packet arrivals, deterministic η, $G_{kr} = 1$, and cost weight $\alpha = 1$. Let J_n be the optimal cost to go at time n, defined according to the Bellman's principle of optimality using equation below, where backlog, interference, and power are all referring to the nth time step and are indexed according to the transmitter-receiver link

$$\begin{aligned} J_n(b_1, I_1, b_2, I_2) = \inf_{p_1, p_2} \{ & p_1 + p_2 + B(b_1) + B(b_2) \\ & + Pr(p_1, I_1) Pr(p_2, I_2) \\ & \times J_{n+1}(b_1 - 1, p_1 + \eta, b_2 - 1, p_2 + \eta) \\ & + (1 - Pr(p_1, I_1))\, Pr(p_2, I_2) \\ & \times J_{n+1}(b_1, p_1 + \eta, b_2 - 1, p_2 + \eta) \\ & + Pr(p_1, I_1)\, (1 - Pr(p_2, I_2)) \\ & \times J_{n+1}(b_1 - 1, p_1 + \eta, b_2, p_2 + \eta) \\ & + (1 - Pr(p_1, I_1))\, (1 - Pr(p_2, I_2)) \\ & \times J_{n+1}(b_1, p_1 + \eta, b_2, p_2 + \eta) \} \end{aligned} \tag{4}$$

where (1) was used to determine the connection between the power used by one transmitter and the interference observed at the other receiver and $Pr(p, I)$ is a short hand for $Pr(success|p, I)$. The boundary constraint for this problem is

$$J_{N+1}(b_1, I_1, b_2, I_2) = B_T(b_1) + B_T(b_2) \tag{5}$$

where $B_T(b_r)$ is the terminal cost due to backlog remaining in the queue of the rth transmitter.

In order to evaluate J_N analytically, derivatives of the right-hand side (r.h.s) of (4) with respect to p_1 and p_2 need to be computed and set simultaneously to 0. However, solution of the two simultaneous equation will involve logarithms of differences of functions, which can only be solved numerically. Thus, we might as well carry out the numerical minimization of the r.h.s. of (4) right away, by numerically computing the derivatives of J_N and using a first or second order gradient method. In the simplest case of a first order method, each optimization step would require 8 evaluations of J_N in order to compute the required derivatives. If evaluation of J_N requires M computations, then in order to evaluate J_{N-k} we would need to make $8^k M^{k+1}$ computations. Therefore, solving (4) through numerical optimization is possible only for very short time horizons.

Another alternative for solving (4) is to discretize each state variable into d points and evaluate J_n at each of those points. If each evaluation takes M computations, then evaluation of J_{N-k} will require $d^4 Mk$ computations. While this is feasible for 2 transmitters, solving J_{N-k} for the case of R transmitters will require $d^{2R} Mk$ computations. This is not possible with the

current computer power for the realistic case of $d = 10$ and $R = 10$. Thus, we can conclude, that even the simplified version of the multitransmitter power control problem cannot be solved using dynamic programming methods for realistic scenarios.

III. Reinforcement Learning Solution

Approximate dynamic programming approach [5] relies on using function approximation architectures for dealing with the curse of dimensionality and on domain simulations for estimating the problem uncertainties and tuning the parameters of these architectures. This approach is also sometimes called reinforcement learning, since it uses online learning in the following Markov decision process framework: a learner observes its state, selects and implements the most appropriate action for this state, transfers to a new state, observes reinforcement (cost or reward) associated with this transition, observes the new state, etc.

Recently, Konda and Tsitsiklis [12] presented an approximate DP algorithm based on the actor-critic (AC) paradigm [13] and proved convergence of actor's parameters to a local optimum under certain technical assumptions. After that, Berenji and Vengerov [3], [4] instantiated their algorithm with a fuzzy rulebase actor and proved that it satisfies the convergence assumptions of Konda and Tsitsiklis. However, their Actor-Critic Based Fuzzy Reinforcement Learning (ACFRL) algorithm converged extremely slowly in the power control problem, where the memory of each learner's action is long-lasting and at the same time a lot of randomness is present in the problem. In fact, in many experiments, the ACFRL algorithm did not show any signs of convergence during any reasonable length of time (up to an hour of simulation time—millions of simulated time steps).

In the next section we describe our new algorithm (ACFRL-2) extending the prior work in [12] and [3] and [4], which avoids the problem of slow convergence and can thus be deployed in real applications. Before explaining the modifications done in ACFRL-2, we first present the general actor-critic framework proposed by Konda and Tsitsiklis (K&T).

The *actor* in their framework represents a mapping between states and actions for the learner (e.g. mapping between backlog/interference observations and the transmitter's power), and the *critic* represents an adviser to the actor, which evaluates actor's performance. The actor then uses this performance evaluation to change its parameters in the gradient direction, minimizing the average cost per time step.

More formally, an actor in the K&T framework is a randomized stationary policy (RSP) parameterized by a vector θ, which assigns to each state $s \in S$ a probability $\mu_\theta(a|s)$ of choosing action $a \in A$. Both of the sets S and A are assumed to be discrete. The critic performs its evaluation by learning the Q-function $Q : S \times A \to R$, mapping state–action pairs into "Q-values." In order to define the Q-function, we first need to define the average cost $\bar{\rho}(\theta)$ for an RSP parameterized by a vector θ (RSP_θ)

$$\bar{\rho}(\theta) = \sum_{s,a} c(s,a)\mu_\theta(a|s)\pi_\theta(s) \tag{6}$$

where $c(s,a)$ is the cost of selecting action a in state s and $\pi_\theta(s)$ is a steady state probability of being in state s under RSP_θ. Then, the "differential cost function" $V_\theta : S \to R$ is defined as a solution to the Poisson equation

$$\bar{\rho}(\theta) + V_\theta(s) = \sum_a \mu_\theta(a|s)\left(c(s,a) + \sum_{s'} p(s'|s,a)V_\theta(s')\right) \tag{7}$$

where $p(s'|s,a)$ is the probability of transferring to a state s' from state s if action a is selected. Intuitively, $V_\theta(s)$ can be interpreted as the expected future excess cost, with respect to the average cost $\bar{\rho}(\theta)$, if the actor is started in state s. Finally, $Q_\theta(s,a)$ is defined by

$$Q_\theta(s,a) = c(s,a) - \bar{\rho}(\theta) + \sum_{s'} p(s'|s,a)V_\theta(s'). \tag{8}$$

If the Q-values associated with the optimal policy are available for all state–action pairs, then the optimal policy can be enacted by taking in each state s the action a that maximizes $Q(s,a)$. This observation is central to the widely used Q-learning approach to solving reinforcement learning problems. While the iterative Q-learning approach converges to the optimal Q-values, in practice it can be carried out only for small state and action spaces, where every action can be performed in every state infinitely many times. As the size of the state space grows, visiting every state becomes more and more improbable, and Q-values need to be generalized across similar states using a function approximation architecture. More importantly, as the size of the action space grows, it becomes more and more difficult to sample every action in any given state. Therefore, presence of a parameterized actor (function approximation architecture), which maps states into actions, is required in problems with very large state and action spaces.

The K&T framework relies on the following inner product expression of the average cost gradient

$$\frac{\partial}{\partial\theta^i}\bar{\rho}(\theta) = \left\langle Q_\theta, \psi^i_\theta(s,a)\right\rangle_\theta = \sum_{s,a}\eta_\theta(s,a)Q_\theta(s,a)\psi^i_\theta(s,a) \tag{9}$$

where $\psi^i_\theta(s,a) = (\partial/\partial\theta^i)\ln\mu_\theta(a|s)$ is the ith basis function and $\eta_\theta(s,a) = \mu_\theta(a|s)\pi_\theta(s)$ is the weighting factor, giving the probability of encountering the state–action pair (s,a) under RSP_θ. Analysis of the above equation shows that in order for the actor to change its parameters along the gradient of average cost, it needs to receive from the critic only a projection of the optimal Q-function onto the space of the basis functions ψ^i_θ, since the inner product of any two vectors u and v is the product of magnitudes of v and u_v, the projection of u onto v. That is, if Q^*_θ is this projection, then

$$\frac{\partial}{\partial\theta^i}\bar{\rho}(\theta) = \left\langle Q^*_\theta, \psi^i_\theta(s,a)\right\rangle_\theta. \tag{10}$$

Since Q^*_θ is a projection onto the space of the basis functions ψ^i_θ, it can be expressed as

$$Q^*_\theta = Q^{r^*}_\theta(s,a) = \sum_{i=1}^{n} r^{*i}\frac{\partial}{\partial\theta^i}\ln\mu_\theta(a|s) \tag{11}$$

for some vector $r^* = (r^{*1}, \ldots, r^{*n})$. The critic in the K&T framework learns the parameter vector $\hat{r}$, which approximates r^*. The critic also stores ρ, an estimate of the average cost per step under the current policy, which is updated according to

$$\rho_{t+1} = \rho_t + \alpha_t \left(c(s_t, a_t) - \rho_t\right). \tag{12}$$

The critic's parameter vector $\hat{r}$ is updated as follows:

$$\hat{r}_{t+1} = \hat{r}_t + \alpha_t \left(c_t - \rho_t + \hat{Q}^{\hat{r}_t}_{\theta_t}(s_{t+1}, a_{t+1}) - \hat{Q}^{\hat{r}_t}_{\theta_t}(s_t, a_t)\right) z_t \tag{13}$$

where α_t is the critic's learning rate at time t, $\hat{Q}$ is an approximation to Q^* in terms of $\hat{r}$ and z_t is an n-vector, called the eligibility trace, to be described shortly. For any λ between 0 and 1, the $\mathrm{TD}(\lambda)$ critic updates z_t according to

$$z_{t+1} = \lambda z_t + \nabla_{\theta_t} \ln \mu_{\theta_t}(a_{t+1}|s_{t+1}). \tag{14}$$

The actor updates its parameters according to

$$\theta_{t+1} = \theta_t - \beta_t \hat{Q}^{\hat{r}_t}_{\theta_t}(s_{t+1}, a_{t+1}) \nabla_{\theta_t} \ln \mu_{\theta_t}(a_{t+1}|s_{t+1}) \tag{15}$$

where β_t is the actor's learning rate at time t.

In order to apply the above framework to any problem, the functional form of $\mu_\theta(a|s)$ needs to be selected. Following the work of Berenji and Vengerov [3], [4], we have used a fuzzy rulebase as the core of $\mu_\theta(a|s)$ for the power control problem. A fuzzy rulebase needed for our purposes is a function f that maps an input vector $s \in R^K$ into a scalar output a. This function is represented by a collection of fuzzy rules. A fuzzy rule i is a function f_i that maps an input vector $s \in R^K$ into a scalar p_i. We have used the following fuzzy rules.

Rule i: IF $(s^1$ is $S^1_i)$ and $(s^2$ is $S^2_i)$ and $\ldots (s^K$ is $S^K_i)$ THEN (p_i), where s^j is the jth component of s, S^j_i are input labels in rule i and p_i are tunable coefficients. Each label is a function $\nu : R \to R$ that maps its input into a degree to which this input belongs to the fuzzy category (linguistic term) described by the label. The exact shape of the input labels for the power control problem will be given in Section VI. A fuzzy rulebase actor $f(s)$ with M rules can then be expressed as

$$a = f(s) = \frac{\sum_{i=1}^{M} p_i w^i(s)}{\sum_{i=1}^{M} w^i(s)} \tag{16}$$

where $w^i(s)$ is the weight of rule i. We used the product inference for computing the weight of each rule: $w^i(s) = \prod_{j=1}^{K} \nu_{S^i_j}(s^j)$. Finally, the power used by the transmitter is sampled from a Gaussian distribution centered at a

$$\mu_\theta(a|s) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(a - a_\theta(s))^2}{2\sigma^2}}. \tag{17}$$

The Gaussian distribution was chosen only for the ease of computer simulation. As Section V will show, any symmetric probability mass function could have been used, which assigns a nonzero probability to selecting every possible action. Also, while the proof is carried out only for discrete action spaces, the ACFRL-2 algorithm can work just as well for continuous actions spaces, and Section VI provides experimental results confirming this claim.

As will be explained in Section VI, the parameters p_i will represent the recommended power output for each fuzzy state observed by the transmitter. These parameters will form the vector θ tuned by the ACFRL-2 algorithm. In the next section we describe the ACFRL-2 algorithm (which we apply to both single and multiple transmitter problems) and in Section V we analyze its convergence properties for the case of the single transmitter problem.

IV. ACFRL-2 Algorithm

The main difficulty in the considered power control problem is that a long wait is required for learning the benefit of using a higher or a lower transmission power in each state. That is, a smaller than average cost per step can be incurred because a higher than recommended power used at time t_0 successfully transmitted a packet and decreased the future backlog for $t > t_0$. However, the backlog can stay at low levels for $t > t_0$ because fewer packets have arrived into the transmitter, which will affect all the future costs until the systems renews itself by emptying the backlog. The difficulty of determining the effects of using a higher or a lower power is aggravated by the fact that the near-optimal policies use decreasing transmission power for lower values of backlog, which makes renewals very infrequent. As a result, it takes a very long time to learn accurate Q-values for randomized stationary policies (RSPs) in problems with characteristics described above. Therefore, the K&T framework, which continually adapts the Q-values and simultaneously uses them to update an RSP-type actor will lead to a very long and noise learning, as was confirmed by our initial experiments in the power control problem.

We propose a solution to this problem, which consists of determining the most likely action $a_\theta(s)$ in each state s using the current fuzzy rulebase parameters θ and then learning the Q-values for two separate policies, one sampling actions $a < a_\theta(s)$ and the other sampling actions $a > a_\theta(s)$ in all encountered states. As a result, the critic can clearly observe the long-term effects of using a higher or a lower power in all states, which makes it much easier to deduce the correct direction for changing actor's parameters.

Implementing this idea is particularly simple with $\mu_\theta(a|s)$ being a fuzzy rulebase giving the mean of the Gaussian distribution from which the final output is sampled. More specifically, implementing ACFRL-2 requires separating the updates to the average cost per step, critic's parameters and actor's parameters into distinct phases. During the first phase, the algorithm simulates the system for N time steps to estimate the average cost per step of the actor's policy. In the second phase only critic is learning based on the average cost ρ obtained during the previous phase. This phase consists of two simulation traces, one sampling actions from the positive half of a Gaussian distribution centered at the value recommended by the current form of the fuzzy rulebase actor and one sampling actions from the negative half of this Gaussian distribution. Correspondingly, two parameter vectors $\hat{r}_1$ and $\hat{r}_2$ are learned.

Finally, in the third phase, the critic is fixed and the actor is learning using (15). The actions for the actor are sampled from

the full Gaussian distribution. However, when the sampled action is higher than the average power suggested by its fuzzy rulebase, the critic's parameter vector $\hat{r}_1$ is used to compute the Q-value of the current state–action pair in (15). Similarly, when the sampled action is lower than the average power suggested by the fuzzy rulebase, the critic's parameter vector $\hat{r}_2$ is used for computing the Q-value. The above three phased are repeated for a desired number of time steps until the learned policy performs well enough for practical purposes. If the maximum possible accuracy is desired and the time is not an issue, the above three phases are repeated until any of the actor's parameters begins to oscillate around some value, which indicates that the neighborhood of the optimal values has been reached. At that point, one can switch to the ACFRL algorithm to perform the fine-tuning of actor's parameters.

As opposed to the fully probabilistic exploration at every time step suggested by Konda and Tsitsiklis, the systematic exploration described above is very beneficial in highly stochastic problems with long renewal periods and long-lasting memory of actions, as it allows the critic to observe more clearly the benefit of each action. Our experimental results show that the ACFRL-2 algorithm converges to the neighborhood of optimal parameters almost deterministically (moving every actor's parameter in the right direction after every repetition of the three-phase procedure described above), while the ACFRL algorithm moves actor's parameters almost as an unbiased random walk even when all parameters are obviously lower or higher than their optimal values.

V. Convergence Properties of ACFRL-2

In this section we analyze the convergence properties of the ACFRL-2 algorithm for the case of a single transmitter facing environment with interference sampled from any fixed probability mass function (PMF). We assume that $\mu_\theta(a|s)$ is a symmetric PMF, which assigns a nonzero probability to selecting every possible action. We first present sufficient convergence assumptions for the general K&T framework for the case of finite state and action spaces.

A0) The learning rate sequences $\{\alpha_t\}$, $\{\beta_t\}$ are positive, nonincreasing, and satisfy $(\beta_t/\alpha_t) \to 0$ as well as

$$\delta_t > 0 \quad \text{for} \quad t > 0, \quad \sum_t \delta_t = \infty \quad \sum_t \delta_t^2 < \infty$$

where δ_t stands for either α_t or β_t.

A1) For each $\theta \in R^n$, the Markov chains $\{S_m\}$ of states and $\{S_m, A_m\}$ of state–action pairs are irreducible and aperiodic, with stationary distributions $\pi_\theta(s)$ and $\eta_\theta(s,a) = \pi_\theta(s)\mu_\theta(a|s)$, respectively, under the RSP μ_θ.

A2) For all $s \in S$ and $a \in A$, the map $\theta \to \mu_\theta(a|s)$ is twice differentiable.

A3) $\mu_\theta(a|s) > 0$, and for all $\theta \in R^n$, $s \in S$, $a \in A$.

A4) The feature vectors $\psi_\theta^i(s,a) = (\partial/\partial\theta^i)\ln\mu_\theta(a|s)$ are uniformly linearly independent. That is, there exists $\epsilon > 0$ such that for all $x \in R^n$,

$$\|x'\psi_\theta\|_\theta^2 \geq \epsilon|x|^2. \tag{18}$$

The sum conditions in assumptions A0 are the standard requirements for a stochastic gradient algorithm to converge. The condition on the ratio of the learning rates ensure that asymptotically the actor seems stationary to the critic, which allows the critic to learn asymptotically the accurate gradient direction for changing actor's parameters.

Assumption A1 is required to ensure that the average cost in the considered problem is well defined. The stochastic nature of arrival and departures from the finite buffer model of the wireless transmitter ensures that the backlog forms an irreducible and aperiodic Markov chain. Since interference is assumed to be randomly sampled from a fixed PMF, the total state combining backlog and interference satisfies assumption A1. Since actions are chosen from a fixed PMF, the sequence of state–action pairs $\{S_m, A_m\}$ also forms an irreducible and aperiodic Markov chain. Assumptions A2 and A3 are sufficient assumptions in order for the gradient of the average cost $\bar{\rho}(\theta)$ to be well defined and expressed compactly as in (9).

Assumption A2 can be verified directly, by differentiating the output of the actor with respect to all parameters and observing that all derivatives exist. To save space, we point the reader to [4] for exact expressions of these derivatives. Assumption A3 holds because of the action PMF properties we assumed at the beginning of this section.

In order to understand the need for Assumption A4, note that when the actor's parameter vector is θ, the critic attempts to approximate the projection of the actual Q-value onto the space of actor's features, as shown in (11). If the actor's features are linearly dependent, the projection operator is ill-conditioned. This possibility is prevented by assumption A4. This assumption was verified for the proposed fuzzy rulebase actor by Berenji and Vengerov [4].

The above assumptions are sufficient to guarantee convergence of the K&T framework if all actions are sampled in every state, which is done in ACFRL. However, separating exploration into two parts in ACFRL-2 creates unexpected difficulties, even though each action is chosen in the long run with the same frequency as when all actions are sampled in ACFRL. That is, while in ACFRL the same vector $\hat{r}$ of critic's parameters gets updated whether the final action is sampled from the right side of the PMF or from the left side, in ACFRL-2 we cannot simply average the resulting vectors $\hat{r}_1$ and $\hat{r}_2$ obtained after sampling from the right side of the PMF for N time steps and then sampling from the left side for N time steps while the actor is held fixed. Intuitively, the reason for this is that different sampling policies have different basis vectors $\psi_\theta^i(s,a) = (\partial/\partial\theta^i)\ln\mu_\theta(a|s)$, and hence parameter vectors r^* in (11) cannot simply be averaged.

More precisely, let $a_\theta(s)$ be the transmission power suggested by the actor's fuzzy rulebase in the state s. If the final action is sampled from the right side of the PMF centered at $a_\theta(s)$, then $a > a_\theta(s)$; otherwise, $a < a_\theta(s)$. Now, observe that for any fixed θ, the ACFRL critic learns to approximate the vector r^*, giving the solution to $Mr^* = Q^*$, where M is a matrix with columns ψ_θ^i, $i = 1, \ldots, n$ and Q^* is a vector of Q-values for all state–action pairs in $S \times A$. Let M_1 be the "top half" of M with entries $\psi_\theta^i(s,a)$ that correspond to state–action pairs with $a > a_\theta(s)$. Similarly, let M_2 be the "bottom half" of M with

entries that correspond to state–action pairs with $a < a_\theta(s)$. Let the components of M_1 and M_2 be ordered in such a way that $M = [M_1^T, M_2^T]^T$. Also, let Q_1^* be the "top half" of Q^* and Q_2^* be the "bottom half" of Q^*. Let the components of Q_1^* and Q_2^* be arranged so that $Q^* = [Q_1^{*T}, Q_2^{*T}]^T$.

When sampling from the right side and from the left side of the action PMF, ACFRL-2 aims to learn differential Q-functions Q_1^* and Q_2^* with respect to the same average cost $\bar{\rho}(\theta)$ of the policy μ_θ, which samples all actions. That is, when the actor is fixed and phase 2 of the ACFRL-2 is repeated continually, the critic converges to r_1^* and r_2^* such that

$$M_1 r_1^* = Q_1^* + E_1 \quad M_2 r_2^* = Q_2^* + E_2 \tag{19}$$

where E_1 and E_2 are the error terms. These errors are introduced because the restricted exploration during the critic's learning in ACFRL-2 samples states according to a different steady-state distribution than the one resulting from the full exploration. Therefore, if we average r_1^* and r_2^*, then we have $1/2[M_1^T, M_2^T]^T(r_1^* + r_2^*) = 1/2[(M_1 r_1^* + M_1 r_2^*)^T, (M_2 r_1^* + M_2 r_2^*)^T]^T = 1/2[(Q_1^* + M_1 r_2^*)^T, (M_2 r_1^* + Q_2^*)^T]^T$. Since we have no control over $M_1 r_2^*$ and $M_2 r_1^*$, averaging r_1^* and r_1^* can lead to a very poor approximation of the Q-function of policy μ_θ performing the full exploration.

Therefore, as suggested in Section IV, the ACFRL-2 critic simply stores the vectors $\hat{r}_1$ and $\hat{r}_2$ obtained during its learning phase and then computes the Q-value of a state–action pair (s,a) during actor's update in phase III of the algorithm as $\hat{Q}_1(s,a) = \sum_{i=1}^n \psi^i(s,a)\hat{r}_1^i$ if a was sampled from the right side of the PMF and as $\hat{Q}_2(s,a) = \sum_{i=1}^n \psi^i(s,a)\hat{r}_2^i$ if a was sampled from the left side of the PMF.

Theorem 1: When the actor's parameters are fixed at θ and phase 2 of the ACFRL-2 is repeated continually, the critic converges to r_1^* when sampling actions from the left side of the action PMF and to r_2^* when sampling actions from the right side. Moreover, these vectors can be used to provide the gradient direction for changing the actor's parameters using the above procedure if all parameters lie on the same side of their optimal values (i.e., $\theta < \theta^*$ or $\theta > \theta^*$, where θ^* is the optimal parameter vector).

Proof: Convergence of the critic to r_1^* if actions are sampled only from the left side of the PMF and to r_2^* if actions are sampled only from the right side follows from the results established in [14]: $\mathrm{TD}(\lambda)$ learning converges with a linear approximation architecture if the states are sampled according to the steady-state distribution resulting from applying the current policy, which is the case during on-policy exploration. A more technical condition required for convergence of $\mathrm{TD}(\lambda)$ is a linear independence of feature functions, whose linear combination is used to approximate the optimal Q-value. This condition holds in ACFRL-2 because of assumption A4.

It now remains to prove that the optimal parameter vectors for the ACFRL-2 critic provide the gradient direction for the actor. Let W_θ be a diagonal matrix with entries $\eta_\theta(s,a)$ for all state–action pairs (s,a), where each entry gives a steady state probability of being in a state s and choosing action a under the policy μ_θ. Then, rewriting (10) in the matrix form

$$\frac{\partial}{\partial \theta^i}\bar{\rho}(\theta) = \left\langle Q_\theta^*, \psi_\theta^i \right\rangle_\theta = \psi^{i^T} W Q^* \tag{20}$$

where we have omitted the subscript θ from the r.h.s. of the above equation and from the rest of the discussion to make notation less cumbersome.

Let the components of W be arranged so that the top half of the diagonal contains entries corresponding to state–action pairs with $a > a_\theta(s)$ and the bottom half of the diagonal contains entries corresponding to state–action pairs with $a < a_\theta(s)$. Then, we want to show that

$$\begin{aligned}&\mathbf{sgn}\left(\psi^{i^T} W \left[Q_1^{*T}, Q_2^{*T}\right]^T\right)\\ &\quad = \mathbf{sgn}\left(\psi^{i^T} W \left[\left(Q_1^* + E_1\right)^T, \left(Q_2^* + E_2\right)^T\right)\right]^T\Bigg).\end{aligned} \tag{21}$$

In the proof below, we will consider only the case of $\theta < \theta^*$, since the same reasoning but with opposite signs will hold in the case when $\theta > \theta^*$.

With the policy μ_θ given by (17), we have

$$\begin{aligned}\psi_\theta^i(s,a) &= \frac{\partial}{\partial \theta^i}\ln \mu_\theta(a|s) = \frac{\frac{\partial}{\partial \theta^i}\mu_\theta(a|s)}{\mu_\theta(a|s)}\\ &= \frac{1}{2\sigma^2}\left(a - a_\theta(s)\right)\frac{\partial}{\partial \theta^i}a_\theta(s).\end{aligned} \tag{22}$$

Therefore, as the above equation shows, the use of a symmetric PMF for exploration in ACFRL-2 implies that $\psi^i(s,a) = -\psi^i(s, a_\theta(s) - (a - a_\theta(s)))$ for $i = 1, \ldots, n$. As a result, $M_1 = -M_2$, which implies that

$$Q_1^* = M_1 r^* = -M_2 r^* = -Q_2^*. \tag{23}$$

As was explained at the end of Section III, θ^i is the transmission power suggested by the ith fuzzy rule. Therefore, $\theta < \theta^*$ implies that in each state it is more preferable to use a higher transmission power than a lower one. That is, if $a > a_\theta(s)$ then $Q^*(s,a) < Q^*(s, a_\theta(s) - (a - a_\theta(s)))$. Furthermore, since $Q_1^* = -Q_2^*$ from (23), we have

$$Q_1^* < 0 \quad Q_2^* > 0. \tag{24}$$

Recalling the arrangement of components of $M = [M_1^T, M_2^T]^T$ and using (22) we get that

$$M_1 > 0 \quad M_2 < 0. \tag{25}$$

Combining this result with (24), we get

$$\psi^{i^T} W \left[Q_1^{*T}, Q_2^{*T}\right]^T < 0, \qquad i = 1, \ldots, n \tag{26}$$

which is consistent with $\theta < \theta^*$, as (20) shows that increasing θ^i will decrease $\bar{\rho}(\theta)$.

Let us consider now the signs of E_1 and E_2. Recall that E_1 and E_2 are defined by equations

$$M_1 r_1^* = Q_{\text{high}}^* = Q_1^* + E_1 \tag{27}$$

$$M_2 r_2^* = Q_{\text{low}}^* = Q_2^* + E_2 \tag{28}$$

where Q_{high}^* is a vector of Q-values for the policy that uses a higher than recommended power in each state by sampling it from the right side of the PMF. Similarly, Q_{low}^* is a vector of Q-values for the policy that uses a lower than recommended power in each state by sampling it from the left side of the PMF. Also, recall that by construction of the ACFRL-2 algorithm both Q_{high}^* and Q_{low}^* are *differential* Q-value vectors with respect to *the same* average cost $\bar{\rho}$ of the policy that samples all actions. Therefore, when $\theta < \theta^*$ and the fuzzy rulebase actor consequently recommends in all states a smaller than optimal transmission power, we have

$$Q_{\text{high}}^* < Q_1^* \quad Q_{\text{low}}^* > Q_2^* \tag{29}$$

because $Q_{\text{high}}^*(s,a)$ denotes expected average cost per step after taking action a in state s and following a more beneficial (the one with a smaller expected average cost) policy thereafter than the one for which Q_1^* is computed. Similarly, $Q_{\text{low}}^*(s,a)$ denotes expected average cost per step after taking action a in state s and following a less beneficial (the one with a larger expected average cost) policy thereafter than the one for which Q_2^* is computed. Finally, by comparing (29) with (27) and (28), we conclude that

$$E_1 < 0 \quad E_2 > 0. \tag{30}$$

Therefore, by combining this result with (24) and (25), we conclude that

$$\psi^{i^T} W \left[\left(Q_1^* + E_1 \right)^T, \left(Q_2^* + E_2 \right)^T \right]^T < 0, \qquad i = 1, \ldots, n. \tag{31}$$

Comparing the above equation with (26), we see that (21) holds. We have now proven that if the ACFRL-2 critic has converged, then the Q-values it will supply to the actor will result in actor's parameters being updated in the gradient direction. **Q.E.D.**

Our experimental results in the power control problem show that the separated exploration in ACFRL-2 provides the correct direction for updating all actor's parameters at every iteration of ACFRL-2, which allows to use a large learning rate for the actor and results in a very fast convergence to a very good near-policy. In contrast, exploration of the full action space in ACFRL results in very noisy updates where the correct direction for each parameter is chosen approximately half of the time, which requires the use of a very small learning rate for the actor and leads to a very slow convergence.

In practice, the optimal learning strategy would be to initialize all actor's parameters below their optimal values and use ACFRL-2 while all parameters are increasing. When any one of the parameters will decrease, the neighborhood of optimal values has been reached, and ACFRL with a full Gaussian exploration can then be used during critic's learning if convergence to exact optimal parameter values is required. However, as results in Section VII show, such a switch for fine tuning the results is not necessary to achieve good performance in the power control problem.

VI. Simulation Setup

We used the algorithm of Foschini and Miljanic (F&M) [6] as a benchmark for our ACFRL-2 algorithm. In the F&M algorithm, a transmitter has a desired signal-to-interference ratio, which depends on the ratio of the transmitter's power to the interference present at that time. Hence, in order to maintain a constant level of SIR, the transmitter raises its power when interference increases and lowers its power when interference decreases. The asynchronous implementation of the F&M algorithm for multiple transmitters is known to converge geometrically fast to the unique Pareto optimal power assignment when the system is feasible [8]. The Pareto optimal assignment in this case means that no transmitter can change its power without decreasing its SIR.

The F&M algorithm was developed to satisfy the needs of voice transmissions, which require a low delay but are essentially not sensitive to transmission errors or lost data up to a certain threshold. Data transmission, on the other hand, can accept some delay but have no tolerance for transmission errors or lost data. Since we focused on data transmissions in this paper, the Pareto optimality of the F&M algorithm no longer holds under our assumptions.

However, F&M algorithm still provides a valuable benchmark for our simulations. For a given arrival rate, higher transmission probability targets in the F&M algorithm correspond to smaller average levels of backlog. For any level of observed interference, F&M algorithm uses the smallest power necessary to support the required transmission probability target. Therefore, even for data transmissions, F&M algorithm is the optimal myopic algorithm (i.e., algorithm that optimizes only the immediate cost). By tuning the transmission probability target, it is possible to find the best tradeoff between minimizing the average backlog and minimizing the immediate power required to support this backlog.

The policies considered in the ACFRL-2 algorithm extend the F&M algorithm by explicitly considering the backlog as a decision variable, which these policies to trade off immediate power gains versus future cost reduction due to decreased backlog. Our simulation results demonstrate that this leads to a significant cost reduction.

Bambos and Kandukuri [2] have shown that the optimal power function is hump-shaped with respect to interference, with the height as well as the center of the hump steadily increasing with backlog. Therefore, the following rules used in the ACFRL-2 actor have a sufficient expressive power to match the complexity of the optimal policy

If (backlog is SMALL) and (interference is SMALL) then (power is p_1)
If (backlog is SMALL) and (interference is MEDIUM) then (power is p_2)
If (backlog is SMALL) and (interference is LARGE) then (power is p_3)
If (backlog is LARGE) and (interference is SMALL) then (power is p_4)

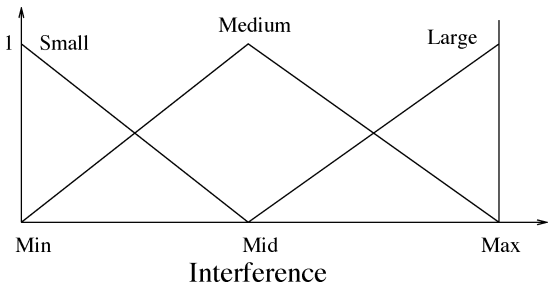


Fig. 1. Interference fuzzy labels used by the transmitters.

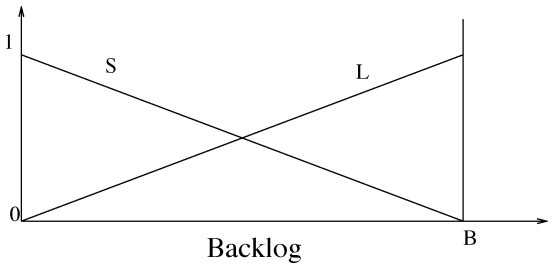


Fig. 2. Backlog fuzzy labels used by the transmitters.

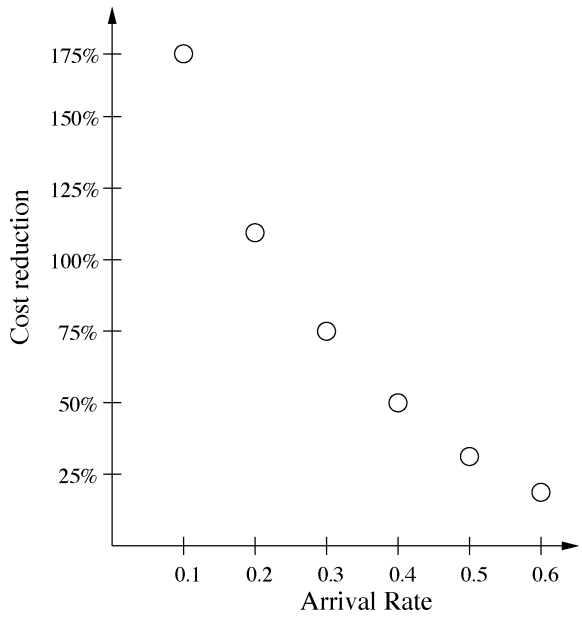


Fig. 3. Cost improvement by ACFRL-2 over F&M algorithm in a nonresponsive environment.

If (backlog is LARGE) and (interference is MEDIUM) then (power is p_5)
If (backlog is LARGE) and (interference is LARGE) then (power is p_6),

where p_1 through p_6 are the tunable parameters. The shapes of the input labels for interference and backlog are shown in Figs. 1 and 2.

The main difficulty in the considered power control problem is that a long wait is required for determining the benefit of using a higher or a lower power. Because both arrivals and successful transmissions are stochastic, many traces are needed in order to distinguish the true value of a policy from random effects. However, as the next section shows, ACFRL-2 was able to deal successfully with this difficulty.

VII. Results and Discussion

The following parameter settings were used in our experiments:

- Initial Backlog $= 0$;
- Buffer Size $= 20$;
- Overflow Cost $L = 100$;
- Power Cost Factor $\alpha = 1$;
- Transmission Noise $\delta = 1$;
- Power gains $G_{jk} = 1$;
- Temporal differencing parameter in the ACFRL-2 critic, $\lambda = 0.95$.

In the first set of experiments, we compared performance of the ACFRL-2 algorithm to that of the F&M algorithm in a nonresponsive environment characterized by a random interference uniformly distributed in [0,100]. In order to evaluate performance of the transmitter using the ACFRL-2 algorithm, the optimal integer-valued constant power level was first determined. After that, the transmitter adopted a power control strategy described by the six fuzzy rules of the previous section, with parameters p_1 through p_6 being initialized at the optimal constant power level found above. Then, the parameters p_1 through p_6 were tuned using the ACFRL-2 algorithm until performance stopped improving. Finally, the resulting policy was tested for 1000 trials.

The results are presented in Fig. 3, which shows the cost reduction of the policy learned by ACFRL-2 over the F&M algorithm for different arrival rates. The target SIR in the F&M algorithm was tuned to provide the best performance while still keeping a stable backlog. To get a sense of the absolute numbers, the costs incurred by the F&M algorithm for the six arrival rates shown in the figure were (9.7, 17.6, 25.8, 34.9, 45.1, 56.9), while the costs incurred by the final ACFRL-2 policy were (3.5, 8.4, 14.7, 23.4, 33.8, 47.9).

We were able to simulate arrival rates only up to 0.6 in this and in the next set of experiments for the following reason. When the arrival rate and correspondingly the required transmission probability approaches 1, the required power approaches infinity, as can be seen from (2). Therefore, for an arrival rate sufficiently close to 1, it will be less costly for the transmitter to reduce the power to 0 and let the backlog overflow than to use a very high power and try to keep a stable backlog. For the current level of power cost factor $\alpha = 1$, this change in behavior occurs for arrival rate equal to 0.7.

In the second set of experiments we considered a more realistic and a more challenging scenario of a responsive noisy environment with varying path gains. Since the logic of the F&M algorithm consists of raising or lowering the transmission power when interference increases or decreases, we chose F&M algorithm for modeling the responsive nature of the environment. That is, the simulation model consisted of two transmitters, one using ACFRL-2 and the other using F&M, and interference experienced by each one was described by (1), with the random noise η_j being uniformly distributed in [20, 30].

The power gain from (1) varied inversely proportional to the distance squared, with proportionality constant equal to 1. The

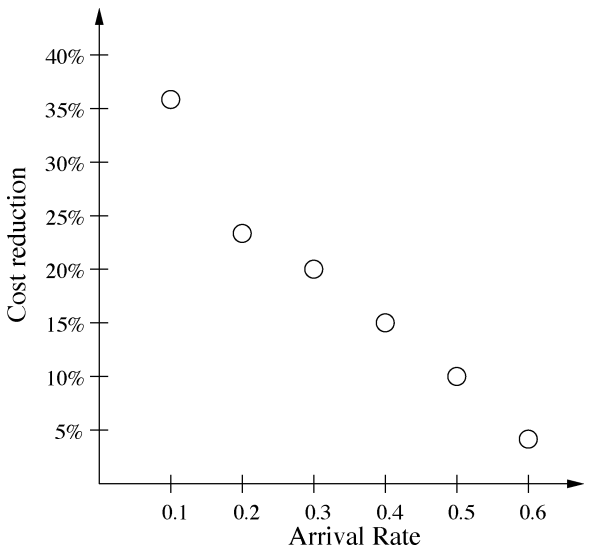


Fig. 4. Cost improvement by ACFRL-2 over F&M algorithm in a responsive environment.

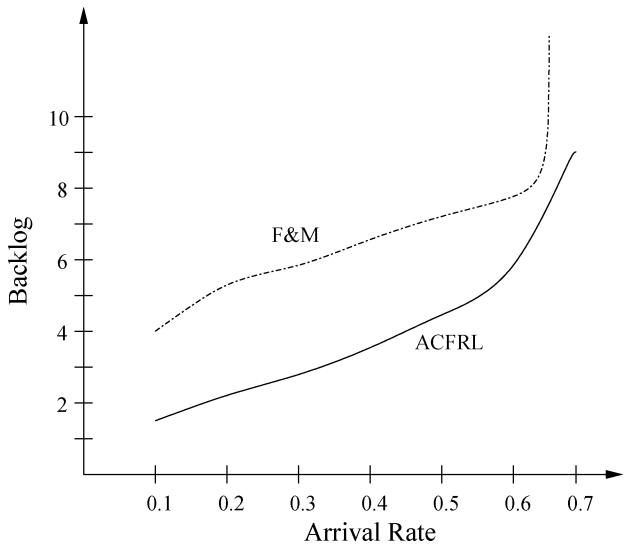


Fig. 5. Backlogs of F&M and ACFRL-2 algorithm for multitransmitter simulation.

distance between transmitters was initialized at 1.5 and followed a random walk with an increment of 0.01 within the interval [1,2]. Variation in the distance was used to model the varying path gain phenomenon found in real wireless channels.

As a benchmark, we considered a scenario where both transmitters were using the F&M algorithm. The experimental procedure consisted of first finding the optimal common target SIR for both transmitters using the F&M algorithm. Performance of the first transmitter was then recorded as a benchmark for the ACFRL-2 algorithm. Then the ACFRL-2 algorithm was evaluated following the steps used in the first set of experiments. The results are presented in Fig. 4. To get a sense of the absolute numbers, the costs incurred by the F&M algorithm were (6.1, 11.2, 17.9, 26.3, 37.2, 53.3), while the costs incurred by the final ACFRL-2 policy were (4.5, 9.1, 14.8, 22.8, 33.4, 50.9).

In our third set of experiments, we considered a realistic scenario of nine transmitters sharing the same channel and creating interference for each other according to (1). The random noise η_j was once again uniformly distributed in [20, 30]. For symmetry, we placed all transmitters at the distance of five units of length from each other. The transmitters were using the ACFRL-2 algorithm in an online fashion to tune their transmission policies. We stopped learning after 1000 trials and tested the resulting policies by recording the average power level and the average backlog among the nine transmitters. To evaluate these results, we considered the case when all nine transmitters were using the F&M algorithm with the target SIR being tuned so that the average power among the transmitters equals to that in the final ACFRL-2 policy. After that, we compared the average backlogs of the transmitters using the F&M algorithm and those using the policies learned by ACFRL-2. The results are presented in Fig. 5 for arrival rates 0.1 to 0.7.

The shape of the policy learned by ACFRL-2 in the first experiment is exactly as predicted by Bambos and Kandukuri [2] for this setup. That is, for low values of backlog the transmitter uses positive power only for low levels of interference and goes into a backoff mode for medium and high interference. Rather than fighting the interference with a high power level, the transmitter buffers the arriving packets and waits for a time slot with a low interference to transmit. However, this behavior is dangerous for high levels of backlog because the buffer can overflow, and ACFRL-2 learns to go only into a soft backoff mode for high levels of interference when backlog is large.

As arrival rate increases, the problem faced by the transmitter becomes more challenging. There is less freedom in using small power and buffering the arriving packets for high levels of interference because of the danger that the buffer can overflow. As a result, the optimal policy approaches the one used by the F&M algorithm—always maintaining the same transmission probability. This explains the decrease in the benefit of ACFRL-2 shown in Figs. 3 and 4 for increasing arrival rates.

Fig. 5 shows that ACFRL-2 is able to learn policies that reduce backlog by more than a factor of 2 for low arrival rates in comparison with the F&M algorithm. For the high arrival rate of 0.7, we found that F&M algorithm simply cannot keep the backlog stable when it attempts to transmit at the same power level as the one learned by the ACFRL-2 algorithm. Thus, ACFRL-2 algorithm can both increase the maximal network throughput by keeping the backlog stable at high arrival rates and increase the network efficiency by decreasing the competition between transmitters.

In order to understand the reasons for such a drastic performance improvement of ACFRL-2 over F&M, we considered a simplified problem of only three transmitters sharing the wireless channel. Also, after observing that the ACFRL-2 algorithm learns the hump-shaped backoff policy, we simplified the work of the actor by explicitly coding this knowledge into the structure of the fuzzy rulebase governing the power selection in each transmitter. That is, we fixed at 0 the parameters p_1 through p_3 of each rulebase.

TABLE I
COMPARISON OF AVERAGE BACKLOG OF THE ACFRL-2 POLICY WITH THE BENCHMARK POLICIES FOR LOW NOISE LEVEL

	Low stress	**High stress**
ACFRL-2 actor	backlog = 0.7	backlog = 0.9
F&M policy	backlog = 5.6	backlog = 7.9
Centralized policy	backlog = 2.1	backlog = 5.1

TABLE II
COMPARISON OF AVERAGE BACKLOG OF THE ACFRL-2 POLICY WITH THE BENCHMARK POLICIES FOR HIGH NOISE LEVEL

	Low stress	**High stress**
ACFRL-2 actor	backlog = 1.2	backlog = 1.4
F&M policy	backlog overflow	backlog overflow
Centralized policy	backlog = 4.7	backlog = 5.5

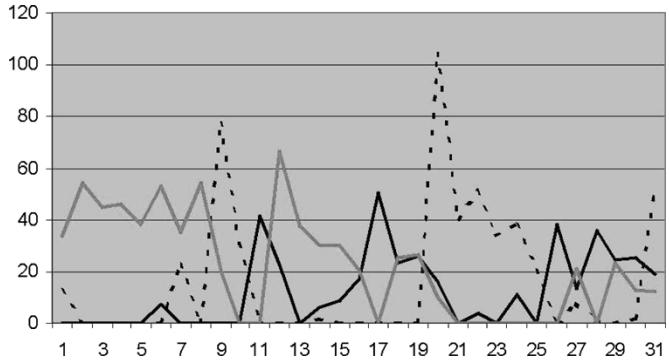


Fig. 6. Power levels over time of three transmitters using a policy learned by ACFRL-2.

We then used the same experimental procedure as above for comparing the ACFRL-2 algorithm with the F&M algorithm. The average backlogs of the two policies are presented in Tables I and II. These results show that ACFRL-2 significantly decreases the average system backlog in comparison with the F&M algorithm for the case of low external noise level. This improvement reaches a new level of importance for the case of high external noise level, where ACFRL-2 increases the maximum system throughput by keeping the backlog stable where the F&M policy leads to a backlog overflow.

We then plotted the backlogs of all the transmitters in a channel over time. Fig. 6 shows this plot for the final policy learned by ACFRL-2. As can be seen from that graph, the transmitters learn to implicitly coordinate their actions and take turns during the transmission. This behavior can also be inferred from the final values of the rulebase parameters p_1, p_2, and p_3. In all the learning scenarios, p_3 attained a large negative value, implying that each transmitter goes into a backoff mode when interference surpasses a certain threshold.

The above observation suggested comparing the ACFRL-2 policy against the policy where only the transmitter with the highest backlog is allowed to transmit in each time slot. This explicit turn-taking policy belongs to a class of efficient but not scalable centralized control policies with a single entity making fully informed decisions. Therefore, in our small-scale simulation we expected this policy to outperform ACFRL-2.

We tested this hypothesis by applying the centralized policy to the fuzzy rulebased transmitters and following a similar comparison process to the one used for the F&M policy. That is, we adjusted the tunable coefficients in the fuzzy rulebases until the average power used by three transmitters matched the power used by the transmitters tuned with ACFRL-2. The results are presented on the last line of Tables I and II. As these tables show, the transmitters using ACFRL-2 to tune their policies in a distributed fashion still outperformed the centralized policy. From this we can conclude that ACFRL-2 allows the transmitters to learn an optimal balance between transmitting simultaneously and taking turns for transmission.

VIII. CONCLUSION

In this paper we presented a new fuzzy reinforcement learning algorithm, ACFRL-2, for optimal decision making in highly stochastic problems with long renewal periods and long-lasting memory of actions that have very large state and action spaces. We presented a convergence proof for ACFRL-2 in the context of the power control problem, which relied on the "ordered action space" property of this problem. This property arose because the real world analog of our discretized model has a continuous action space. Therefore, we can conclude that ACFRL-2 algorithm has a potential of solving many previously unsolved real world problems that have very large state spaces and very large or continuous action spaces.

As a test case, we have applied ACFRL-2 algorithms to the wireless power control problem, which has continuous state and action spaces. The benchmark in our experiments was the standard algorithm of Foschini and Miljanic [6]. In a nonresponsive environment characterized by a random interference, the policies learned by the ACFRL-2 algorithm improved the cost of the F&M algorithm by 175% for low arrival rates and 18% for high arrival rates.

In a more challenging and a more realistic scenario of a responsive environment with varying path gains, the policies learned by the ACFRL-2 algorithm improved the cost of the F&M algorithm by 35% for low arrival rates and 5% for high arrival rates. The decreasing utility of the ACFRL-2 algorithm for high arrival rates is due to the fact that as environment becomes more antagonistic, the backoff behavior that the algorithm can potentially learn becomes less effective.

When ran a realistic simulation with nine transmitters sharing the same channel and causing interference to each other. We recorded the practical performance measures of average transmission delay and the maximal throughput. We found that for low arrival rates the ACFRL-2 algorithm reduced the average

delay by more than a factor of 2 in comparison with the F&M algorithm. Moreover, for high arrival rates the F&M algorithm simply could not keep a stable backlog while the ACFRL-2 algorithm was able to maintain a finite backlog thereby increasing the maximal network throughput.

Analysis of this performance improvement of ACFRL-2 over F&M revealed that the transmitters learn to implicitly coordinate their actions and take turns during the transmission. After comparing ACFRL-2 with an explicit turn-taking policy based on centralized control, we found that ACFRL-2 still performed better. This leads to a conclusion that ACFRL-2 allows the transmitters to learn an optimal balance between transmitting simultaneously and taking turns for transmission.

REFERENCES

[1] N. Bambos, "Toward power-sensitive network architectures in wireless communications: Concepts, issues and design aspects," *IEEE Pers. Commun. Mag.*, vol. 5, no. 3, pp. 50–59, Jun. 1998.

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

[3] H. R. Berenji and D. Vengerov, "On convergence of fuzzy reinforcement learning," in *Proc. 10th IEEE Int. Conf. Fuzzy Systems*, 2001, pp. 618–621.

[4] ——, "A convergent actor critic based fuzzy reinforcement learning algorithm with application to power management of wireless transmitters," *IEEE Trans. Fuzzy Syst.*, vol. 11, no. 4, pp. 478–485, Aug. 2003.

[5] D. Bertsekas and J. Tsitsiklis, *Neuro-Dynamic Programming*. New York: Atheneum, 1996.

[6] G. J. Foschini and Z. Miljanic, "A simple distributed autonomous power control algorithm and its convergence," *IEEE Trans. Vehic. Technol.*, vol. 42, no. 4, pp. 641–646, Nov. 1993.

[7] D. J. Goodman and N. B. Mandayam, "Power control for wireless data," *IEEE Pers. Commun.*, vol. 7, no. 2, pp. 48–54, Apr. 2000.

[8] D. Mitra, "An asynchronous distributed algorithm for power control in cellular radio systems," in *Proc. 4th WINLAB Workshop*, 1993, pp. 249–257.

[9] J. Monks, V. Bharghavan, and W. Hwu, "A power controlled multiple access protocol for wireless packet networks," in *Proc. IEEE Conf. Computer Communications (INFOCOM)*, vol. 1, 2001, pp. 1–11.

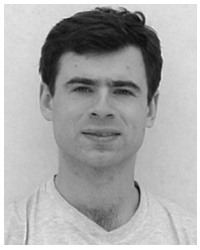
[10] J. Oh, T. Olsen, and K. Wasserman, "Distributed power control and spreading gain allocation in CDMA data networks," in *Proc. IEEE Conf. Computer Communications (INFOCOM)*, 2000, pp. 379–385.

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

[12] V. R. Konda and J. N. Tsitsiklis, "Actor-critic algorithms," *Adv. Neural Inf. Process. Syst.*, vol. 12, pp. 1008–1014, 1999.

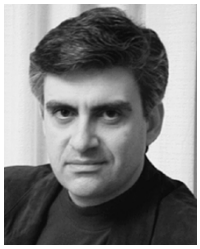
[13] R. S. Sutton and A. G. Barto, "Toward a modern theory of adaptive networks: Expectation and prediction," *Psychol. Rev.*, vol. 88, pp. 135–170, 1981.

[14] J. N. Tsitsiklis and B. Van Roy, "Average cost temporal-difference learning," *Automatica*, vol. 35, no. 11, pp. 1799–1808, 1999.



David Vengerov was born in Moscow, Russia, in 1976. He received the B.S. degree in mathematics and the M.S. degree in electrical engineering and computer science from the Massachusetts Institute of Technology, Boston, in 1997 and 1998, respectively, the M.S. degree in engineering economic systems and operations research and the Ph.D. degree in management science and engineering from Stanford University, Stanford, CA, in 2000 and 2004, respectively.

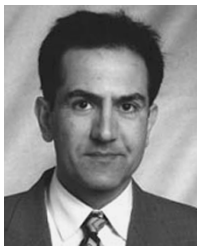
He joined Sun Microsystems Laboratories, Sunnyvale, CA, in 2003, where he is currently developing adaptive algorithms based on distributed intelligence and agent-based systems for domains such as for utility-based scheduling, dynamic data migration in hierarchical storage systems, load balancing, dynamic resource allocation in distributed computer systems, etc.



Nicholas Bambos received the Ph.D. degree in electrical engineering from the University of California, Berkeley, in 1989.

He is currently a Professor at Stanford University, Stanford, CA, holding a joint appointment in the Department of Electrical Engineering and the Department of Management Science. His current research interests include high-performance network architectures, wireless networks and power control, queueing modeling and congestion control, etc.

Dr. Bambos received the NSF Young Investigator Award, has been the Cisco Systems Chair at Stanford, and has received the IBM Faculty Award.



Hamid R. Berenji (F'02) received the B.S. degree from Iran University of Science and Technology, Tehran, in 1979, and the M.S. and Ph.D. degrees in systems engineering from the University of Southern California, Los Angeles, in 1980 and 1986, respectively.

He is currently a Fellow of the Berkeley Initiative on Soft Computing (BISC) at the EECS Department, University of California, Berkeley. He is a Senior Scientist with the Intelligent Inference Systems Corporation, Mountain View, CA, and has many years of experience in performing Research and Development at the Computational Sciences Division of NASA Ames Research Center, Moffett Field, CA. In a joint project between the NASA Ames Research Center and the NASA Johnson Space Center, he and his team developed a new controller for the Shuttle Training Aircraft (STA) that significantly improved its accuracy as tested in the ground hardware facility of NASA, Ellington Field, TX. He originated and extended the theory of generalized reinforcement learning. He has published about 100 technical publications including several book chapters, journal papers, and refereed conference proceeding papers. He was an Area Editor for the *Journal of Fuzzy Sets and Systems*.

Dr. Berenji is the winner of the 1999 NASA Space Act Award and a winner of the NASA Ames Director's Discretionary Fund. He was the Program Co-chairman of the 1993 IEEE Conference on Neural Networks and a Program Co-chairman of the 1994 IEEE Conference on Fuzzy Systems. He has served as an Associate Editor for the IEEE TRANSACTIONS ON NEURAL NETWORKS and the IEEE TRANSACTIONS ON FUZZY SYSTEMS.