

Adaptive resource allocation in telecommunications

Timothy X Brown and Hui Tong

Dept. of Electrical and Computer Engineering
University of Colorado, Boulder, CO 80309-0530
{timxb,tong}@colorado.edu

ABSTRACT

This paper looks at the general problem of resource allocation in telecommunication networks. It gives an overview of the problem and argues for adaptive methods in the complex telecommunication environment. In particular it discusses a general methodology known as reinforcement learning. The paper presents two examples—admission control in packet data networks, and battery management for mobile communication.*

Keywords: Resource allocation, Packet networks, Power control, Neural networks, Reinforcement learning

1. ROLE OF RESOURCE ALLOCATION IN TELECOMMUNICATIONS

A basic telecommunication premise is that a capital intensive infrastructure is shared by as many customers as possible. This philosophy permeates all levels of telecommunications design. In this environment it can be argued that the fundamental problem of telecommunications revolves around allocating the limited resources. This takes place on many time scales from sub-seconds (e.g. call setup) to years (e.g. provisioning and expansion) and on spatial scales from individual devices (e.g. power management) to entire networks (e.g. facilities planning). Table 1 gives a sampling of resource allocation problems.

The resource allocation problem can generically be framed as a set of resource requests that must be assigned to a set of resources. For instance we can consider incoming calls as resource requests, and available trunks as the resources. A resource allocation controller must make decisions about the requests so as to maximize a utility. We divide this problem into two versions, static and dynamic. The static problem is given all requests and resources, a utility is defined for every possible assignment of request to resources, and the goal is to maximize this utility. This is an optimization task that can be addressed by a number of methods such as hill climbing,²⁵ simulated annealing,¹ genetic algorithms,²⁴ or linear programming.⁴ Examples of such a task are packet arbitration and facilities planning in Table 1.

The dynamic problem differs in that requests arrive over time, decisions have future consequences, and the utility incorporates system activity over some time horizon. For instance, a call setup mechanism receives call setup requests over time, routed calls occupy network trunks for the duration of the call, and the utility is the time averaged network revenue. A greedy controller simply makes decisions that maximize immediate revenue. This reduces to the static

*Appears in Proceedings of the SPIE international Symposium on Optical Science, Engineering, and Instrumentation. Denver, CO, July 18–23, 1999

Channel allocation in wireless cellular networks ^{30,33}
Cellular Handover Decisions ²⁶
Arbitration of packet contention in data networks ^{7,15}
Call setup in switches ^{8,31}
Call admission control in data networks ^{12,13,23,29}
Routing in data networks ⁶
Network Management ²⁰
Capacity Allocation and Planning ^{11,17,28}
Facilities Planning ¹⁴

Table 1. Examples of telecommunications resource allocations problems.

problem. More interesting is when the greedy control is not optimal. For instance, the optimal control may reject some call requests because the only available routes occupy trunks over too many hops that could carry several other calls. The traditional approach to this problem is dynamic programming.²

Telecommunication problems often have large state spaces, involve a confluence of statistical processes that may be easy to model but difficult to analyze, and have multiple constraints. For this reason practical closed form resource allocation solutions are rare and solutions are ad hoc or embed conservative or heroic assumptions. To address this we consider adaptive resource allocation. By adaptive resource allocation, we mean the allocation decisions depend on prior experience. Examples of adaptive techniques include neural networks and the more general problem of statistical classification,^{5,16} and reinforcement learning.³

Telecommunication problems are attractive applications for adaptive methods because good, simple to implement, simulation models exist for them in the engineering literature that are both widely used and results on which are trusted, because there are existing solutions to compare with, because small improvements over existing methods can lead to significant savings in the long run, because they have discrete states, and because there are many potential commercial applications.

In this paper we will emphasize one methodology for resource allocation known as reinforcement learning. It specifically target dynamic resource allocation problems and is amenable to a number of problems. The next section will outline the basic reinforcement framework followed by two applications of the framework.

2. REINFORCEMENT LEARNING

This section will briefly describe the decision framework and its background, and then describe a useful approach to the decision problem.

2.1. Markov and Semi-Markov Decision Problems

We consider problems in the following form. For more details, the reader is directed to the literature.^{2,3} We are controlling a system that is always in a discrete state, s . At each state, the controller must choose an actions a from the set of actions at state s , $A(s)$. With each action, the system makes a stochastic transition from state s to s' with probability $p(s, a, s')$ and receives a (possibly) stochastic reward $r(s, a, s')$. The goal of the controller is at every state to maximize the future discounted rewards:

$$J(s) = E \left\{ \int_0^{\infty} e^{-\beta t} r(t) dt | s \right\}, \quad (1)$$

where $E\{\cdot|s\}$ is the expectation over possible trajectories given that we start at state s , $r(t)$ is the total revenue rate at time t , and β is a discount factor that makes immediate profit more valuable than future profit. If time is discrete, then we can simplify to:

$$J(s) = E \left\{ \sum_{t=0}^T \gamma^t r(t) | s \right\}, \quad (2)$$

where γ is the discrete discount factor.[†]

To state the goal in another way, we are seeking a policy, π , that in every state gives an action $\pi(s) \in A(s)$ and this policy maximizes $J(s)$ for every state.

As an example in a network connection admission control problem, the state space might be the number and type of connection on every link in the network plus the current call request, the allowed actions are to either reject or accept the call, the transition probability is defined by the arrival and departure process, and the reward is nothing if we reject the call and the revenue generated by the call if we accept the call. The policy would tell us in every situation whether to accept or reject the call.

[†]Some readers may argue that the correct criteria for optimization is some notion of total reward or average reward rather than discounted reward. This is achieved within our framework via the Tauberian approximation¹⁸ i.e., β is chosen to be sufficiently close to 0 (or γ sufficiently close to 1).

The transition probabilities and the rewards only depend on the current state and action and not on any previous state. For this reason these are called Markov Decision Problems (MDP) if time is discrete or Semi-Markov Decision Problems (SMDP) if time is continuous.

These problems can be solved by dynamic programming. But, two significant obstacles must be overcome, the curse of modeling, and the curse of dimensionality. The first problem is that the transition probabilities of the system under control may not be known explicitly. In the call admission problem, the arrival and departure rates may not be known and may vary depending on the state so that estimating the transition probabilities must be included either explicitly or implicitly when finding a policy. The second problem is that the state space can be huge. For instance, even a modest network can have 10^{50} or more possible configurations. A policy for this many states would be difficult to write down let alone compute. General methods for treating these problems are described in the next sections. For simplicity we will discuss only SMDP with MDP being an obvious special case.

2.2. Q-Learning

Reinforcement learning methods solve SMDP problems by learning good approximations to the optimal value function, J^* , given by the solution to the Bellman optimality equation which takes the following form:

$$J^*(s) = \max_{a \in A(s)} [E_{\Delta t, s'} \{c(s, a, s') + \gamma(\Delta t)J^*(s')\}] \quad (3)$$

where Δt is the random time until the next event, and $\gamma(\Delta t)$ is the effective discount for the next state s' , $\gamma(\Delta t) = e^{-\beta \Delta t}$.

We learn an approximation to J^* using Watkin's Q-learning algorithm. Bellman's equation can be rewritten in Q-values as

$$J^*(s) = \max_{a \in A(s)} Q^*(s, a) \quad (4)$$

We define a set of Q-values by initializing them to random values, simulating the system, and then iteratively updating them with each state transition. For any set of Q-values a policy is defined by choosing the action a in state s that maximizes $Q(s, a)$.

Whatever our decision at a state transition, we update our value function as follows: on a transition from state s to s' on action a in time Δt ,

$$Q(s, a) = (1 - \alpha)Q(s, a) + \alpha \left(r(s, a, s') + \gamma(\Delta t) \max_{b \in A(s')} Q(s', b) \right) \quad (5)$$

where α is the rate of adaptation. In order for Q-learning to perform well, all potentially important state-action pairs (s, a) must be explored. A number of mechanisms are possible such as η -exploration that with probability η chooses a random action otherwise it follows the action recommended by the current Q-values. Whatever exploration we choose, we still use (5) to update Q-values using the action b recommended by the Q-learning.

Note that nowhere do we explicitly use the transition probabilities in this algorithm. This addresses the curse of modeling. Thus, it is possible to use this algorithm using simulation or on-line.

2.3. Function Approximation

The Q-learning in the previous section implicitly uses table lookup, i.e., a table of Q-values is kept and learned for each state-action pair. Theoretical convergence results exist for table lookup that make it desirable when state spaces are small. For large state spaces, there are a number of methods for reducing the number of parameters and this section will outline a few of the most significant.

In state aggregation, a set of states is given a single Q-value and a visit to any state in the set triggers an update of this Q-value. In this way, the state space is partitioned into sets with one Q-value for each partition. Such partitions are often implicit in specific application formulations. The complete state might consist of many features, but only a few important features are used explicitly and in this way aggregating over the unimportant features.

The Q-values can also be considered functions with parameters, θ . In this case (5) becomes:

$$\theta = \theta + \alpha \nabla_{\theta} Q(s, a; \theta) \left(r(s, a, s') + \gamma(\Delta t) \max_{b \in A(s')} Q(s', b; \theta) - Q(s, a; \theta) \right) \quad (6)$$

Ideally the number of parameters is very small and thus an approximate set of Q-values can be learned quickly. The simplest such approximation is a function that is linear in a set of fixed features of the state space:

$$Q(s, a; \theta) = \sum_k \theta(k) \phi_k(s, a), \quad (7)$$

where the $\{\phi_k\}$ are set of fixed functions. As an example, in a networking problem we might use features such as the load on each link in the network, the time of day, etc. More complicated functions of the features using different approximator architectures such as neural networks are possible.

The latter function approximation methods have limited theoretical foundation, but in practice are used quite often and perform well.

With this brief introduction to reinforcement learning we look at two applications of reinforcement learning to dynamic resource allocation telecommunication problems.

3. CONNECTION ADMISSION CONTROL

This section will focus on connection admission control for broadband multimedia communication networks. These networks are unlike the current internet in that voice, video, and data calls arrive and depart over time and, in exchange for giving QoS guarantees to customers, the network collects revenue for calls that it accepts into the network. In this environment, admission control decides what calls to accept into the network so as to maximize the earned revenue while meeting the QoS guarantees of all carried customers.

Meeting QoS requires a decision function that decides when adding a new call will violate QoS guarantees. Given the diverse nature of voice, video, and data traffic, and their often complex underlying statistics, finding good QoS decision functions has been the subject of intense research.^{9,23,34} Recent results have emphasized that robust and efficient QoS decision functions require on-line adaptive methods.¹⁰

Given we have a QoS decision function, deciding which of the heterogeneous arriving calls to accept and which to reject in order to maximize revenue suggests a dynamic program problem. The rapid growth in the number of states with problem complexity has led to reinforcement learning approaches to the problem.²⁹

In this paper we consider the problem of finding a control policy that simultaneously meets QoS guarantees and maximizes the networks earned revenue. We present a general approach to meeting such multicriteria that solves this problem and potentially many other applications.

3.1. Problem Description

This section describes the admission control problem model that will be used. To emphasize the main features of the problem, networking issues such as queuing that are not essential have been simplified or eliminated. It should be emphasized that these aspects can readily be incorporated back into the problem.

We focus on a network access point at the edge of the network. Users attempt to access the network over time and the network immediately chooses to accept or reject the call. If accepted, the call generates traffic in terms of bandwidth as a function of time. At some later time, the call terminates and departs from the network. For each call accepted, the network receives revenue at a fixed rate over the duration of the call. The network measures QoS metrics such as transmission delays or packet loss rates and compares them against the guarantees given to the calls.

Thus the problem is described by the call arrival, traffic, and departure processes; the revenue rates; QoS metrics; QoS constraints; and network model. The choices used in this paper are given in the next paragraph.

Calls are divided into discrete classes indexed by i . The calls are generated via a Poisson arrival process (arrival rate λ_i) and exponential holding times (mean holding time τ_i). Within a call the bandwidth is an ON/OFF process where the traffic is either ON at rate b_i or OFF at rate zero with exponential holding times (mean μ_i^{ON} , and μ_i^{OFF}). The revenue rates are r_i . The network element has a fixed bandwidth B that connects to the network. The total

bandwidth used by accepted calls varies over time. The QoS metric is the fraction of time that the total bandwidth exceeds the network bandwidth (i.e. the overload probability, p). The QoS guarantee is an upper limit, p^* .

In previous work each call had a constant bandwidth over time so that the effect on QoS was predictable. Variable rate traffic is safely approximated by assuming that it always transmits at its maximum or peak rate. Such so-called *peak rate* allocation under-utilizes the network; in some cases by orders of magnitude less than what is possible. Stochastic traffic rates in real traffic, the desire for high network utilization/revenue, and the resulting potential for QoS violations characterize this problem.

3.2. Semi-Markov Decision Processes

At any given point of time, the system is in a particular configuration, x , defined by the number of each type of ongoing calls. At random times a call arrival or a call termination event, e , can occur. The configuration and event together determine the state of the system, $s = (x, e)$. When an event occurs, the learner has to choose an action feasible for that event. The choice of action, the event, and the configuration deterministically define the next configuration and the payoff received by the learner. Then at some subsequent random time another event occurs, and this cycle repeats. The task of the learner is to determine a policy that maximizes the discounted sum of payoffs over an infinite horizon. Such a system constitutes a finite state, finite action, SMDP.

In this paper we restrict the maximization to policies that never enter states that violate QoS guarantees. In general SMDP, due to stochastic state transitions, meeting such constraints may not be possible (e.g. from any state no matter what actions are taken there is a possibility of entering restricted states). In this problem service quality decreases with more calls in the system and adding calls is strictly controlled by the admission controller so that meeting this QoS constraint is possible.

We use Q-learning (5) to solve this problem with the following policy:

Call Arrival: When a call arrives, the Q-value of accepting the call and the Q-value of rejecting the call is determined. If rejection has the higher value, we drop the call. Else, if acceptance has the higher value, we accept the call.

Call Termination: Nothing interesting happens on this event because call departure is not a decision point for RL.[‡]

3.3. Combining Revenue and Quality of Service

A straight-forward formulation to meeting the multicriteria is as follows. For each criteria, j , we estimate a separate set of Q-factors, $Q^j(s, a)$. Each is updated via on-line Q-learning. These are then combined post facto at the time of decision via some function $Q(\cdot)$ so that:

$$Q(s, a) = Q(\{Q^j(s, a)\}). \tag{8}$$

For example in this paper the two criteria are estimated separately as Q^p and Q^q and

$$Q(s, a) = Q(Q^p(s, a), Q^q(s, a)) = \begin{cases} Q^p(s, a) & \text{if } Q^q(s, a) < p^* \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

The structure of this problem allows us to estimate Q^q without using (5). As stated, the QoS is an intrinsic property of a state and not of future states so it is independent of the policy. This allows us to collect QoS statistics about each state and treat them in a principled way (e.g. computing confidence intervals on the estimates). Using these QoS estimates, the set of allowable states contracts monotonically over time eventually converging to a fixed set of allowable states. Since the QoS constraint is guaranteed to reach a fixed point asymptotically, the Q-learned policy also approaches a fixed point at the optimal policy via standard Q-learning proofs. A related scheme is analyzed in¹⁸ suggesting that similar cases will also converge to optimal policies.

[‡]Therefore it is ignored and updates are delayed until the next arrival. An interarrival period may contain zero, one, or more departures. The reward is suitably modified to integrate over the actual $r(t)$ for this period. The effect is to add more variance to the Q-factor estimates in exchange for eliminating all of the states where the event is a departure.

3.4. Simulation Results

The experiments use the following model. The total bandwidth is normalized to 1.0 unit of traffic per unit time. The target overflow probability is $p^* = 10^{-3}$. Two source types are considered with the properties shown in Table 2. As noted before, call holding times are exponential and the arrivals are Poisson. For the first experiment, the ON/OFF holding times are exponentially distributed, while for the second experiment, they are Pareto distributed. The Pareto distribution is considered to be a more accurate representation of data traffic.

Table 2. Experimental parameters

<i>Parameter</i>	Source Type	
	<i>I</i>	<i>II</i>
ON rate, b	0.08	0.2
Mean ON period, $1/\nu^{\text{ON}}$	5	5
Mean OFF period, $1/\nu^{\text{OFF}}$	15	45
Hyperbolic exponent, $u + 1$	2.08	2.12
Call arrival rate, λ	0.067	0.2
Call holding time, $1/\mu$	60	60
Immediate payoff, r	5	1

In the experiments, for each state-action pair, (s, a) , $Q^p(s, a)$ is updated using (5). As stated, in this case the update of $Q^q(s, a)$ does not need to use (5). Since random exploration is employed to ensure that all potentially important state-action pairs are tried, it naturally enables us to collect statistics that can be used to estimate QoS at these state-action pairs, $Q^q(s, a)$. As the number of visits to each state-action pair increases, the estimated $Q^q(s, a)$ becomes more and more accurate and, with confidence, we can gradually eliminate those state-action pairs that will violate QoS requirement. As a consequence, $Q^p(s, a)$ is updated in a gradually correct subset of state-action space in the sense that QoS is met for any action within this subspace. Initial Q-values for RL are artificially set such that Q-learning started with the *greedy* policy (the greedy policy always accepts as long as QoS is met).

After training is completed, we apply a test data set to compare the policy obtained through reinforcement learning (RL) with alternative heuristic policies. The final QoS measurements obtained at the end of the RL training while learning QoS are used for testing different policies. To test the RL policies, when there is a new call arrival, the algorithm first determines if accepting this call will violate QoS. If it will, the call is rejected, else the action is chosen according to $a = \arg \max_{a \in A(s)} Q(s, a)$, where $A(s) = \{1=\text{accept}, 0=\text{reject}\}$. For the QoS constraint we use three cases: Peak rate allocation; Statistical multiplexing function learned on-line, denoted QoS learned; Given statistical multiplexing function a priori, denoted QoS given. We examine six different cases: (1) RL: QoS given; (2) RL: QoS learned; (3) RL: peak rate; (4) A heuristic that only accepts calls from the most valuable class, i.e., type I, with given QoS; (5) Greedy: QoS given; (6) Greedy: peak rate.

From the results shown in Fig. 1, it is clear that simultaneously doing Q-learning and QoS learning converges correctly to the RL policy obtained by giving the QoS a priori and doing standard Q-learning only. We see significant gains (about 15%) due to statistical multiplexing: (6) vs (5), and (3) vs (1). The gains due to RL are about 25%: (6) vs (3), and (5) vs (2). Together they yield about 40% increase in revenue over conservative peak rate allocation in this example. It is also clear from the figure that the RL policies perform better than the heuristic policies.

We repeat the above experiments with Pareto distributed ON and OFF periods, using the same parameters listed in Table 2. The results are shown in Fig. 2. Obviously, RL yields similar revenue gains with the different ON/OFF distributions.

3.5. Summary

This section shows that a QoS constraint could be incorporated into a RL solution to maximizing a network's revenue, using a vector value Q-learning function. The formulation is quite general and can be applied to many possible constraints. The approach, when applied to a simple networking problem, increases revenue by up to 40%. Future research includes: using neural networks or other function approximators to deal with more complex problems for which lookup tables are infeasible; and extending admission control to multi-link routing.

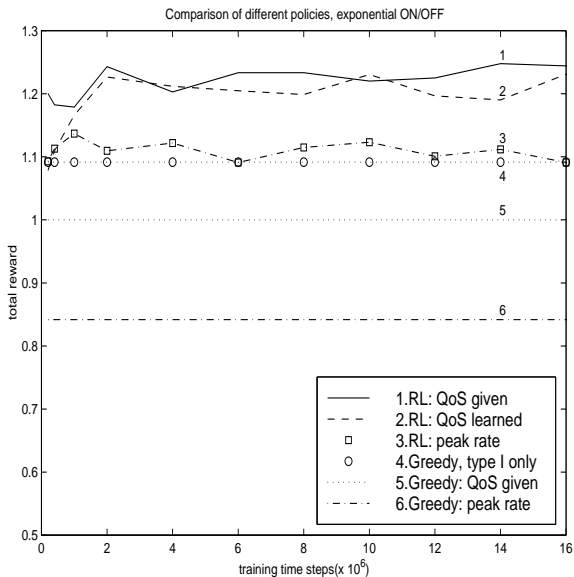


Figure 1. Comparison of total rewards of RL while learning QoS, RL with given QoS measurements, RL with peak rate, greedy policies and peak rate allocation, normalized by the greedy total reward – exponential ON/OFF.

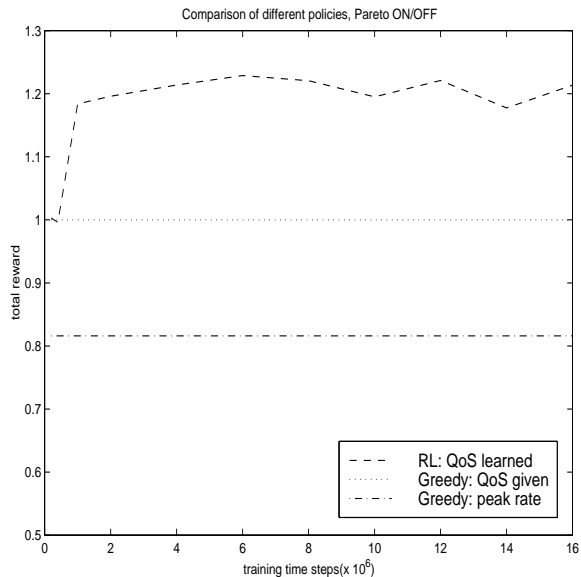


Figure 2. Comparison of total rewards of RL while learning QoS, greedy policy and peak rate allocation, normalized by the greedy total reward – Pareto ON/OFF.

4. BATTERY MANAGEMENT

This section will focus on power management for wireless packet communication channels. These channels are unlike wireline channels in that channel quality is poor and varies over time, and often one side of the wireless link is a battery operated device such as a laptop computer. In this environment, power management decides when to transmit and receive so as to simultaneously maximize channel utility and battery life.

A number of power management strategies have been developed for different aspects of battery operated computer systems such as the hard disk and CPU.^{21,22} Managing the channel is different in that some control actions such as shutting off the wireless transmitter make the state of the channel and the other side of the communication unobservable.

We consider the problem of finding a power management policy that simultaneously maximizes the radio communication's earned revenue while minimizing battery usage. Results show significant reductions in power usage.

4.1. Problem Description

The problem is comprised of five components as shown in Figure 3: mobile application, mobile radio, wireless channel, base station radio, and base station application. The applications on each end generate packets that are sent via a radio across the channel to the radio and then application on the other side. The application also defines the utility of a given end-to-end performance. The radios implement a simple acknowledgment/retransmit protocol for reliable transmission. The base station is fixed and has a reliable power supply and therefore is not power constrained. The mobile power is limited by a battery and it can choose to turn its radio off for periods of time to reduce power usage. Note that even with the radio off, the mobile system continues to draw power for other uses. The channel adds errors to the packets. The rate of errors depends on many factors such as location of mobile and base station, intervening distance, and levels of interference. The problem requires models for each of these components. To be concrete, the specific models used in this paper are described in the following sections. It should be emphasized that in order to focus on the reinforcement learning issues, simple models have been chosen. More sophisticated models can readily be included.



Figure 3. The five components of the radio communication system.

4.1.1. The Channel

The channel carries fixed-size packets in synchronous time slots. All packet rates are normalized by the channel rate so that the channel carries one packet per unit time in each direction. The forward and reverse channels are orthogonal and do not interfere.

Wireless data channels typically have low error rates. Occasionally, due to interference or signal fading, the channel introduces many errors. This variation is possible even when the mobile and base station are stationary. The channel is modeled by a two state Gilbert-Elliot model.¹⁹ In this model, the channel is in either a “good” or a “bad” state with a packet error probabilities p_g and p_b where $p_g < p_b$. The channel is symmetric with the same loss rate in both directions. The channel stays in each state with a geometrically distributed holding time with mean holding times h_g and h_b .

4.1.2. Mobile and Base Station Application

The traffic generated by the source is a bursty ON/OFF model that alternates between generating no packets and generating packets at rate b_{on} . The holding times are geometrically distributed with mean holding times h_{ON} and h_{OFF} . The traffic in each direction is independent and identically distributed.

4.1.3. The Radios

The radios can transmit data from the application and send it on the channel and simultaneously receive data from the other radio and pass it on to its application. The radios implement a simple packet protocol to ensure reliability. Packets from the sources are queued in the radio and sent one by one. Packets consist of a header and data. The header carries acknowledgements (ACK’s) with the most recent packet received without error. The header contains a checksum so that errors in the payload can be detected. Errored packets cause the receiving radio to send a packet with a negative acknowledgment (NACK) to the other radio instructing it to start retransmitting from the errored packet. The NACK is sent immediately even if no data is waiting and the radio must send an empty packet. Only unerrored packets are sent on to the application. The header is assumed to always be received without error[§]

Since the mobile is constrained by power, the mobile is considered the master and the base station the slave. The base station is always on and ready to transmit or receive. The mobile can turn its radio off to conserve power. Every ON-OFF and OFF-ON transition generates a packet with a message in the header indicating the change of state to the base station. These message packets carry no data. The mobile expends power at three levels— P_{OFF} , P_{ON} , and P_{tx} —corresponding to the radio off, receiver on but no packet transmitted, and receiver on packet transmitted.

4.1.4. Reward Criteria

Reward is earned for packets passed in each direction. The amount depends on the application. In this paper we consider three types of applications, an e-mail application, a real-time application, and a web browsing application. In the e-mail application, a unit reward is given for every packet received by the application. In the real time application a unit reward is given for every packet received by the application with delay less than d_{max} . The reward is zero otherwise. In the web browsing application, time is important but not critical. The value of a packet with delay d is $(1 - 1/d_0)^d$, where d_0 is the desired time scale of the arrivals.

The specific parameters used in this experiment are given in Table 3. These were gathered as typical values from.^{27,32} It should be emphasized that this model is the simplest model that captures the essential characteristics of the problem. More realistic channels, protocols, applications, and rewards can readily be incorporated but for this paper are left out for clarity.

[§] A packet error rate of 20% implies a bit error rate of less than 1%. Error correcting codes in the header can easily reduce this error rate to a low value. The main intent is to simplify the protocol for this paper so that time-outs and other mechanisms do not need to be considered.

Parameter Name	Symbol	Value
Channel Error Rate, Good	p_g	0.01
Channel Error Rate, Bad	p_b	0.20
Channel Holding Time, Good	h_g	100
Channel Holding Time, Bad	h_b	5
Source On Rate	b_{on}	0.5
Source Holding Time, On	h_{ON}	100
Source Holding Time, Off	h_{OFF}	100
Power, Radio Off	P_{OFF}	7 W
Power, Radio On	P_{ON}	8.5 W
Power, Radio Transmitting	P_{tx}	10 W
Real Time Max Delay	d_{max}	15
Web Browsing Time Scale	d_0	20

Table 3. Application parameters.

Component	States
Channel	{good,bad}
Application	{ON,OFF}
Mobile	{ON,OFF}
Mobile	{List of waiting and unacknowledged packets and their current delay}
Base Station	{List of waiting and unacknowledged packets and their current delay}

Table 4. Components to System State.

4.2. Markov Decision Processes

At any given time slot, t , the system is in a particular configuration, x , defined by the state of each of the components in Table 4. The system state is $s = (x, t)$ where we include the time in order to facilitate accounting for the battery. The mobile can choose to toggle its radio between on and off and rewards are generated by successfully received packets. The task of the learner is to determine a radio ON/OFF policy that maximizes the total reward for packets received before batteries run out.

The battery life is not a fixed time. First, it depends on usage. Second, for a given drain, the capacity depends on how long the battery was charged, how long it has sat since being charged, the age of the battery, etc. In short, the battery runs out at a random time. The system can be modeled as a stochastic shortest path problem³ whereby there exists a terminal state, $s = 0$, that corresponds to the battery empty in which no more reward is possible and the system remains permanently at no cost.

Given action $a(t)$, while in state $s(t)$ the terminal state is reached with probability $p_{s(t)}(a(t))$. Setting the value of the terminal state to 0, we can convert (2) to our new criterion to maximize:

$$J(s) = E \left\{ \sum_{t=0}^{\infty} c(t) \prod_{\tau=0}^{t-1} (1 - p_{s(\tau)}(a(\tau))) | s \right\},$$

In words, future rewards are discounted by $1 - p_s(a)$, and the discounting is larger for actions that drain the batteries faster. Thus a more power efficient strategy will have a discount factor closer to one which correctly extends the effective horizon over which reward is captured.

We learn a policy via Q-learning (5) where the discount factor, γ , depends on the state and action, i.e., $\gamma(s, a) = 1 - p_s(a)$.

Component	Feature
Mobile Application	number of new packets in the last 4 time slots
Mobile Radio	is radio ON or OFF
Mobile Radio	number of packets waiting at the mobile
Mobile Radio	wait time of first packet waiting at the mobile
Channel	number of errors received in last 4 time slots
Base Radio	number of time slots since mobile was last ON
Base Radio/Applic.	wait time of last received packet (= 0 if none sent)

Table 5. Decision Features Measured by Mobile Radio

4.2.1. Structural Limits to the State Space

For theoretical reasons it is desirable to use a table lookup representation. In practice since the decision is made at the mobile radio using information available to it, this is impossible for the following reasons. The state of the channel is never known directly. The receiver only observes errored packets. It is possible to infer the state, but, only when packets are actually received and channel state changes introduce inference errors.

Traditional wireline applications rarely communicate state information to the transport layer. This state information could also be inferred. But, given the quickly changing application dynamics, the application state is often ignored.

The most serious deficiency to a complete state space representation is that when the mobile radio turns OFF, it has no knowledge of state changes in the base station. Even when it is ON, the protocol does not have provisions for transferring directly the state information. Again, this implies that state information must be inferred.

4.2.2. Simplifying Assumptions

Beyond the structural problems of the previous section we must treat the usual problem that the state space is huge. For instance, assuming even moderate maximum queue sizes and maximum wait times yields 10^{20} states. If one considers e-mail like applications where wait times of minutes (1000's of time slot wait times) with many packets waiting possible, the state space exceeds 10^{100} states. Thus we seek a representation to reduce the size and complexity of the state space. This reduction is taken in two parts. The first is a feature representation that is possible given the structural limits of the previous section, the second is a function approximation based on these features.

The features are listed in Table 5. These are chosen since they are measurable at the mobile radio. For function approximation, we use state aggregation since it provably converges.

4.3. Simulation Results

A crude state aggregation was used with only 96 aggregate states. The battery termination probability, $p_s(a)$ was simply $P/10000$ where P is the power appropriate for the state and action chosen from Table 3. This was chosen to have an expected battery life much longer than the time scale of the traffic and channel processes.

Three policies were learned, one for each application reward criteria. The resulting policies are tested by simulating for 10^6 time slots.

The results are given in Table 6. The main criteria is the power reduction normalized by the maximum possible power reduction. The maximum possible power reduction is computed by assuming that all the packets are sent in one group without error and then the mobile shuts off. The table also lists the average reward per packet received by the application. For the e-mail application, which has no constraints on the packets, the average reward is identically one.

4.4. Summary

This section showed that reinforcement learning was able to learn a policy that significantly reduced the power consumption of a mobile radio while maintaining a high application utility. It used a novel variable discount factor that captured the impact of different actions on battery life. This was able to gain 77% to 90% of the possible power savings.

Application	Normalized Power Savings	Average Reward
E-mail	90%	1
Real Time	77%	0.99
Web Browsing	77%	0.74

Table 6. Simulation Results.

5. CONCLUSION

This paper looked at the general problem of resource allocation in telecommunication networks. It gave an overview of the problem and argued for adaptive methods in the complex telecommunication environment. In particular it discussed a general methodology known as reinforcement learning. The paper presented two examples—admission control in packet data networks, and battery management for mobile communication. In both cases it showed that significantly improved resource allocation was possible.

ACKNOWLEDGMENTS

This work was completed under NSF CAREER Award NCR-9624791 and NSF Grant NCR-9725778. The authors would like to thank Satinder Singh Baveja for introducing them to reinforcement learning. An earlier version of Section 3 appeared in 12.

REFERENCES

1. Aarts, E., Jorst, J., *Simulated Annealing and Boltzmann Machines*, John Wiley and Sons, Chichester, 1989.
2. Bertsekas, D.P., *Dynamic Programming: Deterministic and Stochastic Models*, Prentice-Hall, Englewood Cliffs, NJ, 1987.
3. Bertsekas, D.P., Tsitsiklis, J.N., *Neuro-Dynamic Programming* Athena Scientific, Belmont, MA, 1996.
4. Best, M.J., Ritter, K., *Linear Programming: Active Set Analysis and Computer Programs*, Prentice-Hall, Englewood Cliffs, N.J., 1985.
5. C. M. Bishop. *Neural Networks for Pattern Recognition*. Clarendon Press, Oxford, 1995.
6. Boyan, J.A., Littman, M.L., "Packet routing in dynamically changing networks: a reinforcement learning approach," in Cowan, J.D., et al., ed. *Advances in NIPS 6*, Morgan Kaufman, SF, 1994. pp. 671–678.
7. Brown, T.X, and Liu, K. H., "Neural Network Design of a Banyan Network Controller," *IEEE J. of Selected Areas in Comm.*, Vol. 8, No. 8, pp. 1428–1438, 1990.
8. Brown, T.X, "Neural Networks for Switching," in *Neural Networks in Telecommunications*, eds. B. Yuhas, et al. Kluwer, Boston, 1994. pp. 11–36.
9. Brown, T.X, "Adaptive Access Control Applied to Ethernet Data," *Advances in NIPS 9*, ed. M. Mozer et al., MIT Press, 1997. pp. 932–938.
10. Brown, T.X, "Adaptive Statistical Multiplexing for Broadband Communications," Invited Tutorial *Fifth IFIP Workshop on Performance Modeling & Evaluation of ATM Networks*, Ilkley, U.K., July, 1997.
11. Brown, T.X, "Bandwidth Dimensioning for Data Traffic," in *Proc. of the International Workshop on Applications of Neural Networks to Telecommunication 3*, ed. Alspector, J, et al. Erlbaum, Mahwah, NJ, 1997. pp. 88–96.
12. Brown, T.X, Tong, H., Singh, S., "Optimizing admission control while ensuring quality of service in multimedia networks via reinforcement learning," to appear in *Advances in Neural Information Processing Systems*, ed. M. Kearns et al., MIT Press, 1999.
13. Carlstrom, J., Nordstrom, E., "Reinforcement learning for control of self-similar traffic in broadband networks," in *Proc. of ITC 16*, June 1999, Edinburgh.
14. Chardaire, P., Sutter, A., Costa, M.-C., "Solving the dynamic facility location problem," *Networks*, vol.28, no.2, 1996. p. 117–24.
15. Donis, M.A., Lewis, L., Datta, U., "A hill-climbing approach to the queue depth assignment problem to ensure customer QoS subscriptions," *Proceedings Third IEEE Symposium on Computers and Communications. ISCC'98*. 1998. p. 632-6.

16. Duda, R.O., Hart, P.E., *Pattern Classification and Scene Analysis*, Wiley & Sons, New York, 1973.
17. Engelbrecht, A.P., Cloete, I., "Dimensioning of Telephone Networks using a Neural Network as Traffic Distribution Approximator," in *Proc. of the International Workshop on Applications of Neural Networks to Telecommunication 2*, ed. Alspector, J, et al. Erlbaum, Hillsdale, NJ, 1995. pp. 72–79.
18. Gabor, Z., Kalmar, Z., Szepesvari, C., "Multi-criteria Reinforcement Learning," to appear in *International Conference on Machine Learning*, Madison, WI, July, 1998.
19. Goldsmith, A.J., Varaiya, P.P., "Capacity, mutual information, and coding for finite state Markov channels," *IEEE T. on Info. Thy.*, v. 42, pp. 868–886, May 1996.
20. Goodman, R.M., Ambrose, B., Latin, H., Finnell, H., "Network Operators Advice and Assistance (NOAA): A hybrid Neural Network / Expert System for Traffic Management," *IFIP*, San Francisco, April, 1993.
21. Govil, K., Chan, E., Wasserman, H., "Comparing algorithms for dynamic speed-setting of a low-power cpu," *Proceedings of the First ACM Int. Conf. on Mobile Computing and Networking (MOBICOM)*, 1995.
22. Helmbold, D., Long, D.D.E., Sherrod, B., "A dynamic disk spin-down technique for mobile computing. *Proceedings of the Second ACM Int. Conf. on Mobile Computing and Networking (MOBICOM)*, 1996.
23. Hiramatsu, A., "ATM Communications Network Control by Neural Networks," *IEEE T. on Neural Networks*, v. 1, n. 1, pp. 122–130, 1990.
24. Holland, J.H., *Adaptations in Natural and Artificial Systems*, MIT Press, 1992.
25. Juels, A., Wattenberg, M., "Stochastic Hillclimbing as a Baseline Method for Evaluating Genetic Algorithms," in Touretzky, D.S., et al., ed. *Advances in NIPS 8*, MIT Press, 1996. pp. 430–436.
26. Junius, M., Kennemann, O., "Intelligent Techniques for the GSM Handover Process," in *Proc. of the International Workshop on Applications of Neural Networks to Telecommunication 2*, ed. Alspector, J, et al. Erlbaum, Hillsdale, NJ, 1995. pp. 41–48.
27. Kravits, R., Krishnan, P., "Application-Driven Power Management for Mobile Communication," *Wireless Networks*, 1999.
28. Lewis, L., Datta, U., Sycamore, S., "Intelligent Capacity Evaluation/Planning with Neural Network Clustering Algorithms," in *Proc. of the International Workshop on Applications of Neural Networks to Telecommunication 3*, ed. Alspector, J, et al. Erlbaum, Mahwah, NJ, 1997. pp. 131–139.
29. Marbach, P., Mihatsch, O., Schulte, M., Tsitsiklis, J.N., "Reinforcement learning for call admission control and routing in integrated service networks," in Jordan, M., et al., ed. *Advances in NIPS 10*, MIT Press, 1998.
30. Nie, J., Haykin, S., "A Q-learning based dynamic channel assignment technique for mobile communication systems," to appear in *IEEE T. on Vehicular Technology*.
31. Park, Y.-K., Cherkassky, V., Lee, G., "Omega network-based ATM switch with neural network-controlled bypass queueing and multiplexing," *IEEE JSAC*, v. 12, n. 9, pp. 1471–80.
32. Rappaport, T.S., *Wireless Communications: Principles and Practice*, Prentice-Hall Pub., Englewood Cliffs, NJ, 1996.
33. Singh, S.P., Bertsekas, D.P., "Reinforcement learning for dynamic channel allocation in cellular telephone systems," in *Advances in NIPS 9*, ed. Mozer, M., et al., MIT Press, 1997. pp. 974–980.
34. Verma, S. Pankaj, R.K., Leon-Garcia, A., "Call admission and resource reservation for guaranteed quality of service services in the internet." *Computer Communications*, v. 21, n. 4, Apr. 1998. pp. 362–374.