Dynamic Traffic Splitting to Parallel Wireless Networks with Partial Information: A Bayesian Approach

S. Bhulai[1,2], G.J. Hoekstra[2,3], J.W. Bosman[2], and R.D. van der Mei[2,1]

[1]VU University Amsterdam, Department of Mathematics, The Netherlands
[2]CWI, Probability and Stochastic Networks, Amsterdam, The Netherlands
[3]Thales, Innovation Research & Technology, Huizen, The Netherlands

Abstract

Contemporary wireless networks are based on a wide range of different technologies providing overlapping coverage. This offers users a seamless integration of connectivity by allowing to switch between networks, and opens up a promising area for boosting the performance of wireless networks. Motivated by this, we consider a networking environment in which users are able to select between the available wireless networks to minimize the mean processing times for file downloads in the presence of background traffic. The information available to the user is only the total number of jobs in each network, rather than the per-network numbers of foreground and background jobs. This leads to a complex partial information decision problem which is the focus of this paper.
We develop and evaluate a Bayesian learning algorithm that optimally splits a stream of jobs that minimizes the expected sojourn time. The algorithm learns as the system operates and provides information at each decision and departure epoch. We evaluate the optimality of the partial information algorithm by comparing the performance of the algorithm with the "ideal" performance obtained by solving a Markov decision problem with full state information. To this end, we have conducted extensive experiments both numerically and in a simulation testbed with the full wireless protocol stack. The results show that the Bayesian algorithm has close to optimal performance over a wide range of parameter values.

Key words: Traffic Splitting, Processor Sharing, Concurrent Access, Flow-level Performance, Bayesian dynamic programming, Partial Information Model

1. Introduction

Many of today's wireless networks have already closely approached the Shannon limit on channel capacity, leaving complex signal processing techniques room for only modest improvements in the data transmission rate [8]. A powerful alternative to increase the overall data rate then becomes one in which multiple, likely

different, networks are used concurrently because (a) the spectrum is regulated among various frequency bands and corresponding communication network standards, and (b) the overall spectrum usage remained to be relatively low over a wide range of frequencies [10]. The concurrent use of multiple networks simultaneously has opened up enormous possibilities for increasing bandwidth, improving reliability, and enhancing Quality of Service (QoS) in areas that are covered by multiple wireless access networks. Currently, the efficient use of multiple networks concurrently is an active area of both research [32] and standardisation efforts [11]. However, despite the enormous potential for quality improvement, only little is known about how to fully exploit this potential. This raises the need for new splitting algorithms for concurrent access that are simple, easy to implement, yet effective.

In general, the more detailed information about the state of the system is known (e.g., the number of flows, measured round-trip times and the network load), the higher potential capacity improvements. However, in practice there is often no such detailed information available, or at best only some coarse-grained and aggregated statistics. Therefore, the challenge is to achieve efficient network utilization levels and good end-user application performance, based on information that is only partially available. To address this challenge, we propose a dynamic Bayesian control algorithm that incorporates learning while optimally splits the traffic to the different networks.

In this paper we study the splitting problem in a queueing-theoretical context. We study a model consisting of N non-identical parallel networks that are modeled as processor-sharing PS queues that serve $N+1$ streams of jobs. Stream 0 is called the foreground stream, and streams $1, \ldots, N$ are called the background streams. Jobs of background stream i are served exclusively at PS node i. Each job of the foreground stream has to be routed to one of the PS nodes on the basis of information on the total number of (background and foreground) jobs at each of the nodes. Job sizes are assumed to be exponentially distributed. The goal is to develop a dynamic policy that minimizes the expected sojourn time of foreground jobs. Motivated by practice, we assume that the decision maker is not able to distinguish the number of foreground and background jobs in the network, but instead only has information to the total number of jobs. The challenge is to deal with this partial information problem. In this paper, we address this problem through a learning mechanism, where the decision maker makes a statistical inference on the distribution of the numbers of foreground and background jobs after the each decision. To this end, we model the system as a Bayesian decision problem. In this context, the decision making involves learning on the partially observed states while at the same time optimal actions are chosen. Experiments, conducted both numerically and in a wireless simulation testbed, show that the algorithm has a performance that is close to the case in which full information is

available. This makes the algorithm a promising first step towards efficient traffic splitting in practice.

The main motivation for this paper stems from applications of traffic splitting in the context of multiple overlapping wireless networks. This application domain leads to a number of model assumptions that are not covered by existing papers in the literature. First, an important aspect is the *presence of background traffic* (i.e., streams of jobs that are handled by one specific network only and hence are not split), which may have a strong impact on the performance and optimality of the splitting of the (foreground) stream of jobs; the inclusion background traffic adds an interesting complexity to the model. Second, in practice background and foreground traffic can often not be separated, because the traffic from multi-homed systems is difficult to distinguish from the traffic in networks of single-homed systems, so that the only information available is the total number of jobs in the networks. An optimal foreground-splitting algorithm ideally has knowledge on the numbers of foreground and background jobs in the system. As a consequence, we have to deal with the availability of *partial information* (at best we only know the sum of the number of foreground and background jobs for each network) only, which adds another level of complexity to the model, and requires for smart methods to do so.

The contribution in this paper is two-fold. On the methodological side, it is known that Bayesian learning algorithm are notoriously difficult to implement and to derive optimal policies from. While the vast majority of papers (see, e.g., [4, 6, 14, 29]) propose simplified structures of the optimal policy, we reduce the dimensionality of the state space by using structural properties of the problem. In addition, we show that efficient discretization of the state space leads to numerical tractability (see also Remark 2.1). On the practical side, the proposed algorithm is new in the context of traffic splitting in the presence of concurrent access for wireless networks. Such algorithms open up highly promising means to boost bandwidth in the wireless networks. The algorithm is effective, yet easy to apply in practical systems, and as such extremely useful for real wireless network deployments.

In the literature, a variety of fundamental and applied studies have been focused to the splitting and scheduling jobs to multiple nodes. The available results and techniques are briefly outlined below. In the context of telecommunication systems, the concurrent use of multiple network resources in parallel was already described for a Public Switched Digital Network (PSDN) [9], where inverse multiplexing was proposed as a technique to perform the aggregation of multiple independent information channels across a network to create a single higher-rate information channel. Various approaches have appeared to exploit multiple transmission paths in parallel. For example, by using multi-element antennas, as

adopted by the IEEE 802.11n standard [1], at the physical layer or by switching datagrams at the link layer [7, 21], and also by using multiple TCP sessions in parallel to a file server [31]. In the latter case, each available network transports part of the requested data in a separate TCP session. Previous work has indicated that downloading from multiple networks concurrently may not always be beneficial [12], but in general significant performance improvements can be realized [13, 17, 19]. Under these circumstances of using a combination of different network types, in particular, the transport layer-approaches, have shown their applicability [19] as they allow appropriate link layer adaptations for each TCP session.

In a queueing-theoretical context, there is very little literature on partial information models. Bellman [3] was the first to study decision problems with a transition law that is not completely known. He observed that the problem could be transformed into an equivalent full observation problem by augmenting the state space with the set of probability distributions defined on the domain of the unknown quantity (i.e., the unobserved state, or the unknown parameter) and updating it by Bayes' rule. The transformation of the partial information problem to the complete information model, however, comes with added computational difficulties, since policies are defined over a continuum of states. This is the fundamental problem in developing algorithms for computing optimal policies [28]. There is some work in the theoretical domain to characterize the structure of the optimal policy (see, e.g., [5, 2, 15, 24]). Even then, finding the optimal policy computationally for a general Bayesian decision problem is intractable. Approaches dealing with this are to be satisfied with suboptimal solutions or to develop algorithms that can exploit problem characteristics (see, e.g., [23, 29, 34, 14, 4, 6]). We refer to [25, 27, 33, 22] for some surveys on computational techniques.

The organization of the paper is as follows. In Section 2 we describe the model and introduce the notation. Moreover, we discuss the full information model in Section 2.1 and the Bayesian analysis in Section 2.2. In Section 3 we discuss the numerical results, comparing the performance of our Bayesian approach to the fully observable MDP, not only in a queueing-theoretical setting where networks are modeled as PS nodes, but also in a simulation setting where the full wireless protocol stack is implemented. Finally, in Section 4 we address a number of topics for further research.

2. Problem formulation

In this section we describe the concurrent access problem in greater detail. We model N mobile networks as PS servers so that multiple jobs are served simultaneously. Accordingly, in our model we consider server selection policies instead of network selection policies. There are $N + 1$ streams of jobs in the

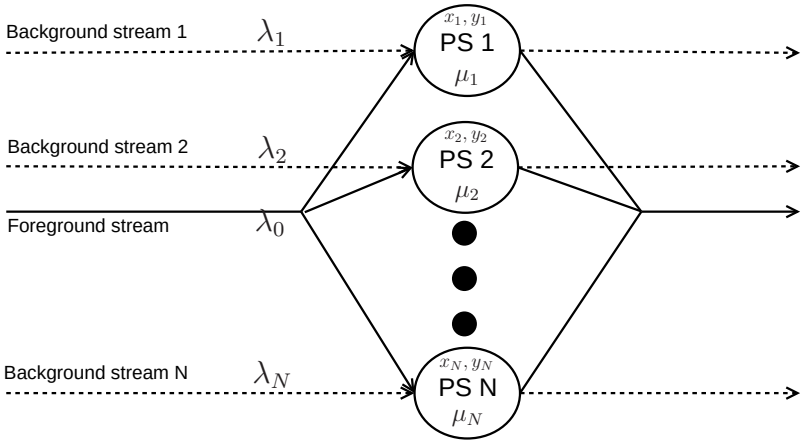

Figure 1: The concurrent access network.

system. Stream i generates a stream of jobs for server i for $i = 1, \ldots, N$. Stream 0 generates a stream of jobs for which the jobs can be sent to either server 1 up to server N. Hence, streams 1 to N can be seen as *background* traffic, and stream 0 as *foreground* traffic. We assume that all streams are modeled by a Poisson process with parameters $\lambda_0, \ldots, \lambda_N$, respectively. After a job enters the system, it demands service from the system. We assume that the service times follow an exponential distribution with mean service time $1/\mu_i = \beta_i$ for $i = 0, \ldots, N$. Then, the occupation rates ρ_i are defined by $\rho_i = \lambda_i \beta_i$. Note that for stability we require that the total load $\rho := \rho_1 + \cdots + \rho_N < N$. Based on the above information, there is a central decision maker that has to decide on the distribution of the foreground jobs over the N servers. Let N be the number of foreground jobs in the system (at all servers). Then, the aim of the decision maker is to minimize the expected average number of foreground jobs in the system. Note that this is directly related to the sojourn times of the foreground traffic (see Figure 1).

In the sequel we will study two dynamic models: the optimal server selection model with full and partial observability. We first describe the full information model.

2.1. Full observation model

In this subsection we allow the decision maker to dynamically send the jobs to any server. To find the optimal policy for making this decision, we model this as a Markov decision problem. To this end, let the state space $\mathcal{S} = \mathbb{N}_0^{2N} = \{0, 1, 2, \ldots\}^{2N}$. A tuple $s = (x_1, \ldots, x_N, y_1, \ldots, y_N) \in \mathcal{S}$ denotes that there are x_i foreground jobs and y_i background jobs at server i for $i = 1, \ldots, N$. For each job, the set of actions is given by $\mathcal{A} = \{1, \ldots, N\}$, where $a \in \mathcal{A}$ denotes sending the job to server a. When action a is chosen in state s, there are two possible events in the system; first, an arrival of a job can occur with rate λ_i or a job can finish

his service with rate μ_i for $i = 0, \ldots, N$. The transition rates are thus given by $p = p(s, a, s')$, when the system is in state s, action a is taken and the next state is s', as follows:

$$p(s,a,s') = \begin{cases} \lambda_0, & \text{if } s' = s + e_a, \\ \lambda_i, & \text{if } s' = s + e_{i+N} \text{ for } i = 1, \ldots, N, \\ \mu_0, & \text{if } s' = s - e_i \text{ and } x_i > 0 \text{ for } i = 1, \ldots, N, \\ \mu_i, & \text{if } s' = s - e_{i+N} \text{ and } y_i > 0 \text{ for } i = 1, \ldots, N, \\ 0, & \text{otherwise,} \end{cases}$$

for $s, s' \in \mathcal{S}$ and $a \in \mathcal{A}$, where e_i is the zero-vector with a one at the i-th entry. The term λ_0 represents an arrival of a foreground job that is assigned to node a, λ_i represents an arrival of a background job at node i, μ_0 represents the departure rates at all node i in which foreground jobs are available, and μ_i represents the departure rate of a background job at node i, if available. Since we are interested in the number of foreground jobs in the system, we take the cost function c equal to $c(s) = x_1 + \cdots + x_N$. The tuple $(\mathcal{S}, \mathcal{A}, p, c)$ defines the Markov decision problem.

Next, we uniformize the system (see Section 11.5 of [30]). To this end, we assume that the uniformization constant $\lambda_0 + \cdots + \lambda_N + \sum_{i=1}^{N} \max\{\mu_0, \mu_i\} = 1$; we can always get this by scaling. Uniformizing is equivalent to adding dummy transitions (from a state to itself) such that the rate out of each state is equal to 1; then we can consider the rates to be transition probabilities. Define a *deterministic policy* π as a function from $\mathcal{S}$ to $\mathcal{A}$, i.e., $\pi(s) \in \mathcal{A}$ for all $s \in \mathcal{S}$. Note that the optimal policy can be found within this class (see [16]). Let $u_t^\pi(s)$ denote the total expected costs up to time t when the system starts in state s under policy π. Note that for any stable and work-conserving policy, the Markov chain satisfies the unichain condition, so that the average expected costs $g(\pi) = \lim_{t\to\infty} u_t^\pi(s)/t$ is independent of the initial state s (see Proposition 8.2.1 of [30]). The goal is to find a policy π^* that minimizes the long-term average costs, thus $g = \min_\pi g(\pi)$.

Let $V(s)$ be a real-valued function defined on the state space. This function will play the role of the relative value function, i.e., the asymptotic difference in total costs that results from starting the process in state s instead of some reference state. The long-term average optimal actions are a solution of the optimality equation (in vector notation) $g + V = TV$, where T is the dynamic programming

operator acting on V defined as follows

$$\begin{aligned} TV(s) = &\sum_{i=1}^{N} x_i + \lambda_0 \min_{a\in\{1,\ldots,N\}} \{V(s+e_a)\} + \sum_{i=1}^{N} \lambda_i V(s+e_{i+N}) + \\ &\sum_{i=1}^{N} \tfrac{x_i}{x_i+y_i}\mu_0 V(s-e_i) + \sum_{i=1}^{N} \tfrac{y_i}{x_i+y_i}\mu_i V(s-e_{i+N}) + \\ &\Big(1-\lambda_0 - \sum_{i=1}^{N} \big[\lambda_i + \tfrac{x_i}{x_i+y_i}\mu_0 + \tfrac{y_i}{x_i+y_i}\mu_i\big]\Big) V(s). \end{aligned} \tag{1}$$

The first term in the expression $TV(s)$ models the direct costs, the second term deals with the arrivals of foreground jobs, whereas the third term deals with the background jobs. The fourth and fifth terms concern service completions for foreground and background jobs, respectively. The last line is the uniformization constant.

The optimality equation $g + V = TV$ is hard to solve analytically in practice. Alternatively, the optimal actions can also be obtained by recursively defining $V_{l+1} = TV_l$ for arbitrary V_0. For $l \to \infty$, the maximizing actions converge to the optimal ones (for existence and convergence of solutions and optimal policies we refer to [30]). Note that for numeric computation the state space needs to be truncated to obtain a finite state space. In practice, one determines the truncation by systematically increasing the truncation bound until no significant changes in the average costs occur.

2.2. Bayesian partial information model

The dynamic server selection model with full information uses a state description $(x_1, \ldots, x_N, y_1, \ldots, y_N)$ with $2N$ entries. However, in practice, distinguishing the foreground traffic from the background traffic might not be feasible. In these cases, one can only observe the state $(z_1, \ldots, z_N)$ with $z_i = x_i + y_i$ for $i = 1, \ldots, N$. Now, the dynamic control policy that we derived in the previous section cannot be applied straightforwardly. To apply the control policy one needs to create a mapping from $(z_1, \ldots, z_N)$ to $(x_1, \ldots, x_N, y_1, \ldots, y_N)$, so that (an estimate of the) full information is recovered. Note that it is not sufficient to create a mapping solely based on $(z_1, \ldots, z_N)$ at each decision epoch, since it does not use the information contained in the sample path, i.e., many sample paths can lead to the same state $(z_1, \ldots, z_N)$. Therefore, we will use Bayesian learning that takes into account the complete history of states in the estimation procedure. We shall call $z = (z_1, \ldots, z_N) \in \mathbb{N}_0^N$ the observation state. In order to learn about the division between the number of foreground and background jobs, we will denote by $u_i(n)$ the probability that at server i there are n foreground jobs for $i = 1, \ldots, N$. The probability distribution u_i will serve the purpose of

information about the states that cannot be observed; hence, $u = (u_1, \ldots, u_N)$ is called the belief state. Note that the belief state space is of high dimension, namely $\prod_{i=1}^{N} \{u_i \in [0,1]^{\mathbb{N}_0} \mid \sum_{x \in \mathbb{N}_0} u_i(x) = 1\}$.

Based on the observation and belief states, we construct a state space for the Bayesian dynamic program consisting of the vectors $s = (z, u)$. Note that every arrival and departure gives the system information on how to update the belief state. Suppose that state s is given and that an arrival of foreground job that is admitted to server i occurs. The new state s_{af_i} is then given by $s_{af_i} = (z + e_i, u')$ where $u_i'(x) = u_i(x-1)$ for $x > 0$ and $u_i'(0) = 0$, and where $u_j'(x) = u_j(x)$ for $j \neq i$. In case of arrival of a background job to server i, we have a new state $s_{ab_i} = (z + e_i, u)$.

In case of departures, we have a similar state transformation. When a foreground job leaves server i, then we have corresponding states $s_{df_i} = (z - e_i, u')$ with $u_i'(x) = u_i(x+1)$ for $x \geq 0$. Similarly, when a background job leaves server i, then we have $s_{db_i} = (z - e_i, u)$. Naturally, these transitions cannot be observed, so we take the expectation with respect to the probability distribution u to average over all sample paths. This gives a new dynamic programming operator in which learning is incorporated. This is given by

$$\begin{aligned} TV(s) = \sum_{x_1 \in \mathbb{N}_0} \cdots \sum_{x_N \in \mathbb{N}_0} u_1(x_1) \cdots u_N(x_N) \Big[\sum_{i=1}^{N} x_i + \sum_{i=1}^{N} \lambda_i V(s_{ab_i}) + \\ \lambda_0 \min\{V(s_{af_1}), \ldots, V(s_{af_N})\} + \\ \sum_{i=1}^{N} \tfrac{x_i}{z_i} \mu_0 V(s_{df_i}) + \sum_{i=1}^{N} \tfrac{z_i - x_i}{z_i} \mu_i V(s_{db_i}) + \\ \Big(1 - \lambda_0 - \sum_{i=1}^{N} \big[\lambda_i + \tfrac{x_i}{z_i}\mu_0 + \tfrac{z_i - x_i}{z_i}\mu_i\big]\Big) V(s) \Big]. \end{aligned} \tag{2}$$

Note that the basic idea to transform Equation (1) into Equation (2) is to take the conditional expectation with respect to the belief state distribution u. Under this condition, the foreground and background jobs can be distinguished so that the structure of the equation resembles the one of the fully observed problem. However, only the transitions to the new belief state need to be adjusted so that the information that has been learned is taken into account. These transitions are provided above.

We end this section with two remarks.

Remark 2.1 (Complexity): Note that the dynamic programming operator for the Bayesian model (2) resembles the dynamic programming operator of the full

observation model (1). However, the state space of the Bayesian model is of significantly higher dimension as the state variables for the background traffic are continuous. Hence, solving the optimality equation $g + V = TV$ is notoriously hard, both analytically and numerically. In general, the Bayesian updates result in posterior distributions that cannot be captured by a nice structural form. In our problem, however, the decision maker can distinguish foreground and background upon arrival leading to an arrival process with deterministic state transitions. It is only the departures that carry uncertainty with them. This leads to a state transition function, as described above, which keeps the dimensionality of the state space at reasonably low levels. In this way, the structure of the problem makes the Bayesian model a tractable approach (after discretization of the state space). Also note that for arbitrary nodes i and j, the decision as to whether an incoming foreground job should join node i or j, does not depend on the other nodes. Hence, in the decision making one can compare node 1 and 2, take the best node and compare it to node 3, take the best of that comparison and compare it to node 4, and so forth. This leads to a sequence of $N - 1$ comparisons. Therefore, the Bayesian approach scales linearly in running time with the number of nodes N.

Remark 2.2 (Accuracy): In a general Bayesian setting, the belief state represents a probability distribution that represents the likelihood that the process is in a particular state. The accuracy of this estimate, generally, tends to deteriorate as the process progresses due to accumulated errors. In our problem setting, the accuracy of the estimates tends to improve as jobs leave the system. As more jobs leave the system, the support of the posterior distribution reduces to a smaller set of states, limiting the possibilities for errors. In fact, upon departure of the last job in a particular node, the posterior distribution of that node is independent of the past, since the state is exactly known. Thus, all probability mass is concentrated on having 0 jobs in that node. Hence, an empty node leads to a belief state that corresponds to the true state for that node. This observation increases the accuracy of our algorithm due to stability of the system.

3. Numerical Experiments

To assess the performance of our Bayesian algorithm for efficiently assigning downloads with concurrent access, we have performed extensive numerical experimentation, comparing the results of the Bayesian algorithm to the results of the fully observable MDP in which the foreground and background jobs can be distinguished. The comparison is done in two settings. Section 3.1 compares the performance of the full observation MDP with the Bayesian MDP under the model described in Section 2. Section 3.2 evaluates the performance of the algorithms in a state-of-the-art wireless network simulation package in which the full wireless protocol stack is implemented. We have performed a large number

of experiments with a wide range of parameter settings. The results are outlined below.

The number of model parameters is significantly large for already moderate numbers of nodes, prohibiting extensive numerical experiments over the full spectrum of parameter combinations. To highlight the main effects of parameter changes, we considered two-node scenarios, motivated by the fact that in the context of wireless networks, at most a few parallel networks will be used simultaneously per user. Moreover, the file-size distributions were taken to be exponential (see also the remarks in Section 4) with means $\beta_0 = \beta_1 = \beta_2 = 1$. To allow for asymmetric network settings, we scale the job-size distribution in node i by C_i, i.e., the effective job size is therefore β_i/C_i for node $i = 1, 2$. To include scenarios for light and heavy foreground traffic, the foreground traffic $\rho_0 = \lambda_0\beta_0/\min\{C_1, C_2\}$ was varied as 0.1 and 0.9, and the background loads $\rho_1 = \lambda_1\beta_1/C_1$ and $\rho_2 = \lambda_2\beta_2/C_2$ were varied between 0.1 and 0.9.

3.1. Comparison of the fully observed MDP against the Bayesian algorithm

To compare the quality of the Bayesian approach discussed above to the fully observed MDP approach, we have calculated the mean of the sojourn time S of an arbitrary job for both policies, under a variety of parameter settings. To this end, for each scenario we have calculated the mean number of jobs, following the lines of Sections 2.1 and 2.2, and then used Little's formula to obtain $\mathbb{E}S$. Denoting by $\mathbb{E}[S|\text{Bayes}]$ and $\mathbb{E}[S|\text{full MDP}]$ the expected sojourn times under the Bayesian approach and the full MDP approach, respectively, the relative difference is defined as follows

$$\Delta\% = \frac{\mathbb{E}[S|\text{Bayes}] - \mathbb{E}[S|\text{full MDP}]}{\mathbb{E}[S|\text{full MDP}]} \times 100\%. \tag{3}$$

Note that the simulations have been run with 10^7 foreground jobs resulting in a 99% confidence interval of approximately 0.1% with respect to the point estimates.

Equal network capacities

Table 1 shows the results for light foreground load (with $\rho_0 = 0.1$) and for a variety background-load values (ρ_1, ρ_2) for the case of symmetric networks, and where the network capacities are equal (normalized to $C_1 = C_2 = 1$). More specifically, for each scenarios, Table 1 shows the triple $(\mathbb{E}[S|\text{Bayes}], \mathbb{E}[S|\text{full MDP}], \Delta\%)$, where $\Delta\%$ is defined in (3). Note that, due to symmetry, only the results for $\rho_2 \geq \rho_1$ are shown. Table 2 and Figure 2 show the results for medium to heavy foreground load (with $\rho_0 = 0.9$).

ρ_1 \ ρ_2	0.1	0.3	0.5	0.7	0.9
0.1	(1.073, 1.072, 0.2%)	(1.116, 1.115, 0.1%)	(1.165, 1.164, 0.1%)	(1.210, 1.210, 0.0%)	(1.245, 1.241, 0.4%)
0.2		(1.196, 1.195, 0.1%)	(1.273, 1.271, 0.2%)	(1.352, 1.350, 0.1%)	(1.416, 1.409, 0.5%)
0.3		(1.282, 1.277, 0.4%)	(1.393, 1.390, 0.2%)	(1.516, 1.514, 0.1%)	(1.628, 1.625, 0.2%)
0.4			(1.522, 1.519, 0.2%)	(1.715, 1.711, 0.2%)	(1.918, 1.911, 0.3%)
0.5			(1.674, 1.665, 0.5%)	(1.958, 1.952, 0.3%)	(2.318, 2.308, 0.4%)
0.6				(2.260, 2.255, 0.3%)	(2.910, 2.897, 0.5%)
0.7				(2.654, 2.641, 0.5%)	(3.878, 3.858, 0.5%)
0.8					(5.735, 5.705, 0.5%)
0.9					(11.064, 11.020, 0.4%)

Table 1: Comparison of $(\mathbb{E}[S|\text{Bayes}], \mathbb{E}[S|\text{full MDP}], \Delta\%)$ for foreground load $\rho_0 = 0.1$.

ρ_1 \ ρ_2	0.1	0.3	0.5	0.7	0.9
0.1	(1.59, 1.55, 2.3%)	(1.93, 1.88, 2.6%)	(2.53, 2.46, 2.9%)	(3.76, 3.70, 1.7%)	(8.90, 8.89, 0.2%)
0.2		(2.21, 2.15, 2.9%)	(3.08, 2.99, 2.8%)	(5.31, 5.23, 1.5%)	unstable
0.3		(2.60, 2.51, 3.5%)	(3.99, 3.86, 3.4%)	(9.74, 9.63, 1.1%)	unstable
0.4			(5.67, 5.60, 1.3%)	unstable	unstable
0.5			(10.87, 10.62, 2.4%)	unstable	unstable

Table 2: Comparison of $(\mathbb{E}[S|\text{Bayes}], \mathbb{E}[S|\text{full MDP}], \Delta\%)$ for foreground load $\rho_0 = 0.9$.

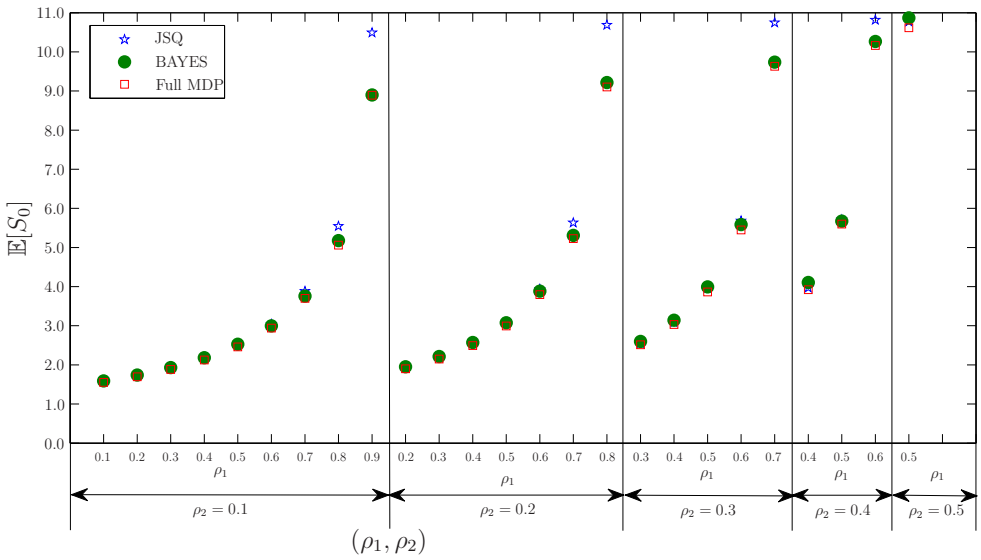

Figure 2: Comparison of $(\mathbb{E}[S|\text{Bayes}], \mathbb{E}[S|\text{full MDP}], \mathbb{E}[S|\text{JSQ}])$ for foreground load $\rho_0 = 0.9$.

Tables 1 and 2 reveal that the Bayesian algorithm has very good performance that is close to a system with full information for a wide range of parameter combinations of background loads (ρ_1, ρ_2). This is remarkable since the Bayesian algorithm has less information available than the full observation MDP, but learns sufficiently to make good decisions. Note that Table 1, with $\rho_0 = 0.1$, has a performance that is extremely close to the full observation model, with errors typically less than 0.5%. Table 2, with $\rho_0 = 0.9$, has a slightly degraded performance, with errors typically less than 3.5%, over a broad range of parameters settings. Nonetheless, the Bayesian algorithm clearly outperforms the widely used Join the Shortest Queue (JSQ) algorithm, see Figure 2. The slightly degraded performance of the Bayesian algorithm is due to the fact that the Bayesian algorithm has estimates of the probability distribution on the foreground and background traffic with high accuracy as more and more jobs leave the system. When the load moves from 0.1 to 0.9 the time it takes to return to more accurate estimates is longer, which explains the slightly degraded performance (see also Remark 2.2).

Unequal network capacities

To assess the usefulness of the Bayesian approach to the case of unequal network capacities, we have also performed experiments for the case $C_1 \neq C_2$. Table 3 below shows the results of the model considered in Table 1, but with the (normalized) network capacities for $C_1 = 1$ and $C_2 = 4$. Similarly, Table 4 shows the results for the models considered in Table 2, with network capacities $C_1 = 1$ and $C_2 = 4$.

ρ_1 \ ρ_2	0.1	0.3	0.5	0.7	0.9
0.1	(0.286, 0.286, 0.0%)	(0.370, 0.370, 0.0%)	(0.514, 0.513, 0.2%)	(0.742, 0.741, 0.2%)	(1.067, 1.062, 0.5%)
0.2	(0.286, 0.286, 0.0%)	(0.370, 0.370, 0.0%)	(0.516, 0.516, 0.1%)	(0.767, 0.766, 0.1%)	(1.167, 1.162, 0.4%)
0.3	(0.286, 0.286, 0.0%)	(0.370, 0.370, 0.0%)	(0.519, 0.519, 0.0%)	(0.792, 0.792, 0.1%)	(1.286, 1.281, 0.4%)
0.4	(0.286, 0.286, 0.0%)	(0.370, 0.370, 0.0%)	(0.521, 0.521, 0.1%)	(0.816, 0.815, 0.1%)	(1.424, 1.419, 0.3%)
0.5	(0.286, 0.286, 0.0%)	(0.370, 0.370, 0.0%)	(0.523, 0.523, 0.0%)	(0.838, 0.837, 0.1%)	(1.593, 1.587, 0.4%)
0.6	(0.286, 0.286, 0.0%)	(0.370, 0.370, 0.0%)	(0.525, 0.525, 0.0%)	(0.858, 0.858, 0.1%)	(1.800, 1.794, 0.3%)
0.7	(0.286, 0.286, 0.0%)	(0.370, 0.370, 0.0%)	(0.525, 0.525, 0.0%)	(0.875, 0.874, 0.1%)	(2.050, 2.044, 0.3%)
0.8	(0.286, 0.286, 0.0%)	(0.371, 0.371, 0.0%)	(0.526, 0.526, 0.0%)	(0.891, 0.891, 0.0%)	(2.366, 2.360, 0.2%)
0.9	(0.285, 0.285, 0.0%)	(0.371, 0.371, 0.0%)	(0.526, 0.526, 0.0%)	(0.901, 0.901, 0.0%)	(2.773, 2.769, 0.1%)

Table 3: Comparison of $(\mathbb{E}[S|\text{Bayes}], \mathbb{E}[S|\text{full MDP}], \Delta\%)$ for foreground load $\rho_0 = 0.1$ with $C_1 = 1$ and $C_2 = 4$.

Tables 3 and 4 show again that the Bayesian algorithm has excellent performance, doing much better than JSQ (see Figure 3). Under light foreground traffic load ($\rho_0 = 0.1$), the error is extremely small (0.5% or far less). When the foreground traffic load is increased to $\rho_0 = 0.9$ the error remains small with

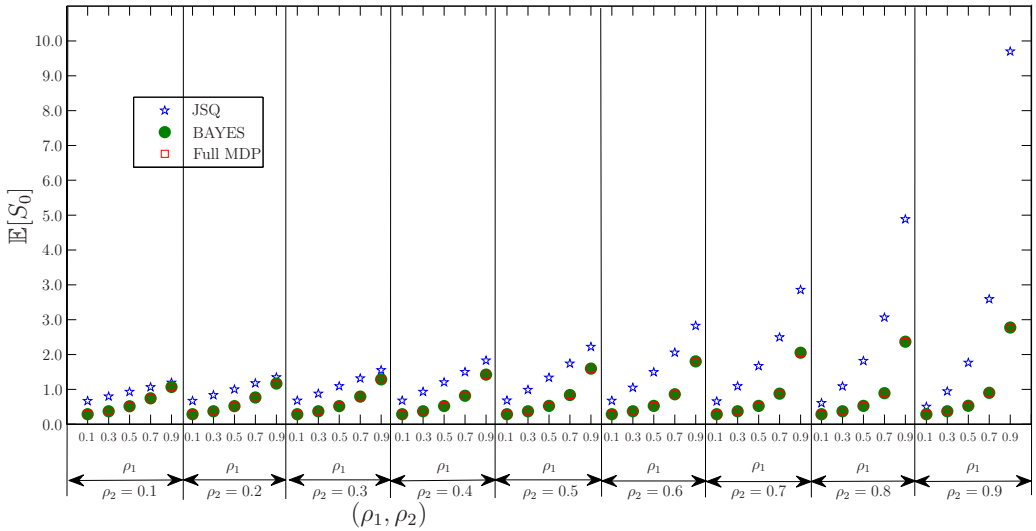

Figure 3: Comparison of $(\mathbb{E}[S|\text{Bayes}], \mathbb{E}[S|\text{full MDP}], \mathbb{E}[S|\text{JSQ}])$ for foreground load $\rho_0 = 0.1$ with $C_1 = 1$ and $C_2 = 4$.

ρ_1 \ ρ_2	0.1	0.3	0.5	0.7	0.9
0.1	(0.369, 0.366, 0.8%)	(0.498, 0.493, 1.0%)	(0.704, 0.701, 0.39%)	(1.127, 1.115, 1.00%)	(2.867, 2.745, 4.4%)
0.2	(0.369, 0.367, 0.6%)	(0.501, 0.498, 0.6%)	(0.734, 0.722, 1.61%)	(1.208, 1.197, 0.90%)	
0.3	(0.369, 0.368, 0.3%)	(0.511, 0.504, 1.3%)	(0.754, 0.745, 1.20%)	(1.300, 1.290, 0.80%)	
0.4	(0.370, 0.369, 0.3%)	(0.514, 0.509, 0.9%)	(0.775, 0.769, 0.81%)		
0.5	(0.370, 0.369, 0.2%)	(0.516, 0.514, 0.5%)	(0.797, 0.792, 0.60%)		
0.6	(0.370, 0.370, 0.1%)	(0.521, 0.518, 0.6%)			
0.7	(0.370, 0.370, 0.1%)	(0.522, 0.521, 0.3%)			
0.8	(0.370, 0.370, 0.0%)				
0.9	(0.370, 0.370, 0.0%)				

Table 4: Comparison of $(\mathbb{E}[S|\text{Bayes}], \mathbb{E}[S|\text{full MDP}], \Delta\%)$ for foreground load $\rho_0 = 0.9$ with $C_1 = 1$ and $C_2 = 4$.

typical values of 1% up to a maximum of 4.4% for an unbalanced system.

Comparing the results with the results of the symmetric network capacity settings in Tables 1 and 2, we observe that the relative performance in the asymmetric case of the Bayesian algorithm is even better, especially when the foreground load is significant. This may be explained by the fact that in the asymmetric system with $C_2 = 4$, the Bayesian algorithm has more observations to learn from due to the increased number of departures in the node 2, and because an erroneous decision has little impact due to short busy periods in node 2. Only when node 2 is heavily loaded, the system performance is extremely sensitive to the proper scheduling actions, which explains the maximum error in Table 4.

In conclusion, the results in this section show that the Bayesian algorithm has a performance extremely close to the performance of a fully observed MDP both under symmetric and asymmetric systems.

3.2. Comparison of the fully observed MDP against the Bayesian algorithm in wireless networks

To evaluate the accuracy of the Bayesian approach in a realistic wireless networking environment, we have performed extensive experimentation in OPNET Modeler [20] that implements the full protocol stack of wireless equipment. We emphasize that this is not a trivial experiment, because the Bayesian learning and decision functionality must be embedded in the network terminals and the complex mapping of the physical and medium access control layer parameters to the PS node parameters must be performed. In this context, it was shown in [18] based on extensive network simulations how the download performance of TCP-based wireless networks can be modeled by PS nodes, by using the proper parameter mapping. In our experiments, the simulated response-time performance of TCP-based networks under traffic splitting under the Bayesian approach was benchmarked against the simulated response-time performance under the fully observable MDP traffic splitting. The results are outlined below.

3.2.1. Experimental configuration

Figure 4 shows the network topology in which the traffic splitting solution is supposed to operate. All wireless terminals download files from an application server, which may also be a dispatcher in front of several application servers (not shown). The application server has information about the number of ongoing downloads over each of the WLAN access networks, AP1 and AP2, but is unable to distinguish between the multi-homed and the single homed terminals, because there is no binding between both network addresses of the multi-homed terminals. Both WLAN access points operate on non-overlapping frequency channels to establish two non-interfering parallel paths to the application server from the multi-homed systems. The transmission links from the access points towards the

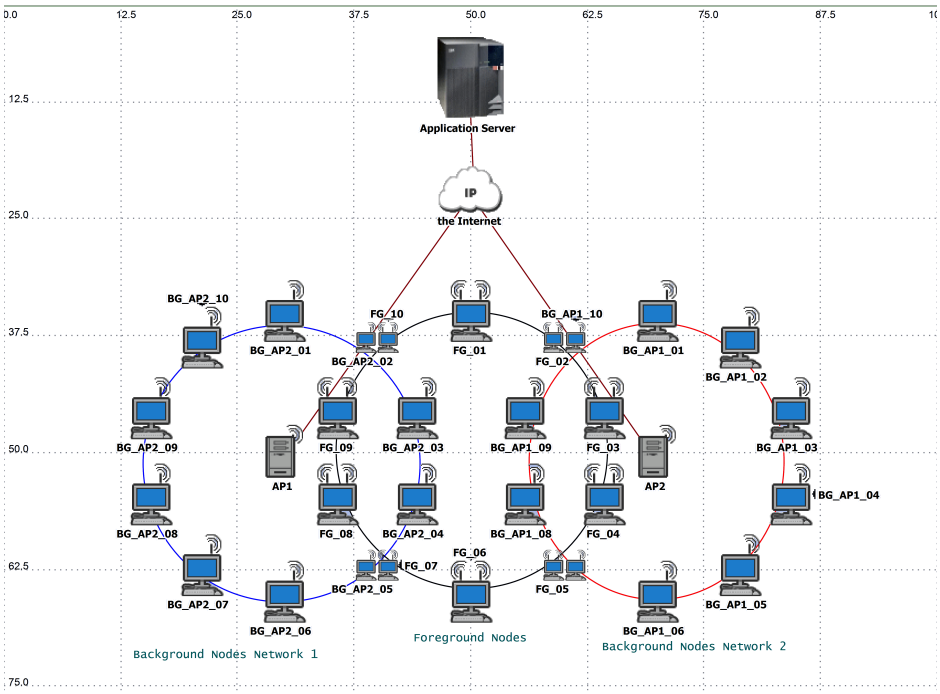

Figure 4: The simulated concurrent access network.

application server are considered to incur a negligible delay and loss to packets from and to the access points. This assumption is motivated by the much higher capacities and reliability offered in contemporary fixed-line carrier-grade Internet connections in comparison to the IEEE 802.11b access networks.

The analytic model from [18] captures the combined dynamics and protocol overhead of the 802.11 MAC, IP, TCP and application-layer into an explicit expression for the effective service time of a file download. Based on the effective service time, the effective load can be determined of the file transfers in our simulated WLAN networks with a flow-level $M/G/1$ Processor Sharing (PS) model.

In the simulated network there are ten multi-homed terminals (named FG_01-FG_10) that generate download requests (that are considered foreground jobs in the queuing model) with arrival rate λ_0. These foreground terminals are positioned between both access points in a circle with a radius of 15 meter. In addition there are ten single-homed terminals (named with prefix $BG\_AP1\_$) that generate background traffic in network 1 with file downloads arriving with rate λ_1 to the first network. The remaining ten single-homed terminals (named with prefix $BG\_AP2\_$) generate background traffic at rate λ_2 in network 2 in a similar fashion. All background terminals are positioned at an equal distance of 15 meter from their respective access point. The file download requests arrive according to an independent Poisson process and may have multiple file transfers in progress.

The MAC/PHY parameters of the WLAN stations are set in accordance to the widely deployed IEEE 802.11b standard amendment as it relies on the same MAC protocol basis as the contemporary higher rate (IEEE 802.11 a/g/n) amendments and has lower computational requirements for high-load network simulations.

Table 5 summarizes the IEEE 802.11 MAC parameters used in our analytic model to calculate the effective load values for the simulation runs.

parameter	value	parameter	value
mac	224 bits	ack	112 bits
difs	50 μs	R_b	$\{1, 11\} \cdot 10^6$ bps
sifs	10 μs	$Cw_{\min}$	31 slots
eifs	364 μs	phy	192 μs
δ	1 μs	τ	20 μs
R_c	10^6 bps		

Table 5: IEEE 802.11b MAC parameters

In this table, mac is the number of bits of overhead bits associated to a MAC data frame. The difs, sifs, eifs are the DCF, short and extended interframe spacing times, respectively. The δ is the propagation delay that is assumed in our analytic model. R_c is the transmission rate for WLAN acknowledgments of size ack bits, and R_b is the WLAN transmission rate for MAC data frames that is set to 1 or 11 Mbps. $Cw_{\min}$ corresponds to the minimum contention window in slots. Phy is the physical layer overhead, and τ is the slot time. In addition to the WLAN MAC, specific settings apply to the higher protocol layers and are outlined in Table 6.

variable	setting
X_{FTPget}	4096 bits
$X_{FTPclose}$	64 bits
TCP_{stack}	Full-Featured
X_{MSS}	11584 bits
$X_{tcp/ip}$	416 bits
w	70080 bits (8760 bytes)
X_{file}	$1.6 \cdot 10^6$ bits

Table 6: Network and application settings

In Table 6, X_{FTPget} is the size of the FTP GET-command that is issued for initiating a file download, $X_{FTPclose}$ is the size of the FTP CLOSE-command that concludes the file transfer at the application. The TCP stack used in our experiments is characterized in OPNET as 'Full-Featured', which is an enhanced version of TCP Reno that uses Selective Acknowledgments (SACK) [26] and has

a slightly smaller MSS, X_{MSS} (in bits), due to the use of timestamps to fit in the 1500 bytes that are used as the WLAN data frame payload. The number of TCP/IP overhead bits per segment is $X_{tcp/ip}$ bits. The maximum TCP receiver window size is indicated as w (in bits), and the file size as X_{file} (in bits).

Based on the parameter setting from Table 5 and 6 and respecting the engineering guidelines from [18] we can assume that the mean download response times in our simulation model can be accurately predicted from the effective load of the network using the $M/G/1$ PS model.

3.2.2. Experimental results

To assess the performance of the Bayesian algorithm in practice, we have performed simulations with the OPNET Modeler [20] with the full wireless protocol stack implemented for both the fully observable MDP and the Bayesian model. The OPNET simulations for the experimental results have been run with approximately $322,000$ foreground jobs and the background jobs ranging from roughly $644,000$ jobs to 5.1 million jobs depending on the load. In our simulation study we have considered two scenarios. One simulation scenario considers equal capacity networks in which all terminals are configured to use a WLAN transmission rate of 11 Mbps. For simulating a scenario in which the network capacity of both access network is unequal, the WLAN transmission rate used in AP2 is lowered to 1 Mbps, which reduces the medium capacity for processing file transfers by a factor of 5.79. In this scenario, the background load applied to AP2 is based on the lower capacity, whereas the foreground traffic intensity remains the same as for the equal capacity network.

We have executed 48 runs for the equal capacity scenario (24 runs for the fully observed MDP and 24 runs for the Bayesian MDP) and 80 runs for the unequal capacity scenario. All runs have completed a total simulation time of 300 hours per run of which 1 hour is the warm-up time leading to a wall clock time of approximately 75 hours per run. This experimental setup is sufficient to derive a 99% confidence interval of approximately 0.7% with respect to the point estimates. The results of the experiments for both scenarios are outlined in Table 7 and Table 8 for $\rho_0 = 0.1$ and a number of combinations ρ_1 and ρ_2. Note that the load values in this setting are obtained following the parameterization as defined and validated in [18].

The results in Table 7 and 8 show that the Bayesian approach again closely matches the performance of the full observation MDP model in a real networking environment for equal and non-equal capacity networks. This is most remarkable, since the OPNET Modeler is a packet-level simulator implementing the full protocol stack including TCP, IP, MAC, and PHY layers, whereas both the fully observable and the Bayesian MDP is based on a processor-sharing model for the networking environment. The results also illustrate that the Bayesian approach is a very powerful means that is practically applicable.

ρ_1 \ ρ_2	0.1	0.3	0.5	0.7	0.8
0.1	(0.355, 0.354, 0.31%)	(0.370, 0.369, 0.30%)	(0.385, 0.385, 0.05%)	(0.402, 0.400, 0.49%)	(0.408, 0.406, 0.46%)
0.2		(0.396, 0.395, 0.21%)	(0.420, 0.420, 0.10%)	(0.446, 0.446, 0.05%)	(0.456, 0.455, 0.11%)
0.3		(0.421, 0.421, 0.18%)	(0.460, 0.458, 0.33%)	(0.502, 0.498, 0.65%)	(0.521, 0.519, 0.48%)
0.4			(0.503, 0.501, 0.35%)	(0.565, 0.564, 0.18%)	(0.601, 0.599, 0.37%)
0.5			(0.551, 0.547, 0.61%)	(0.643, 0.639, 0.68%)	(0.700, 0.696, 0.53%)
0.6				(0.742, 0.737, 0.68%)	(0.831, 0.828, 0.29%)
0.7				(0.867, 0.865, 0.20%)	(1.028, 1.018, 1.01%)
0.8					(1.322, 1.319, 0.23%)

Table 7: Comparison of $(\mathbb{E}[S|\text{Bayes}], \mathbb{E}[S|\text{full MDP}], \Delta\%)$ in OPNET for $\rho_0 = 0.1$.

ρ_1 \ ρ_2	0.1	0.3	0.5	0.7	0.8
0.1	(0.417, 0.415, 0.4%)	(0.415, 0.415, 0.0%)	(0.416, 0.416, 0.0%)	(0.416, 0.416, 0.0%)	(0.415, 0.415, 0.0%)
0.2	(0.476, 0.474, 0.4%)	(0.477, 0.475, 0.3%)	(0.479, 0.476, 0.6%)	(0.475, 0.475, 0.0%)	(0.475, 0.475, 0.0%)
0.3	(0.556, 0.555, 0.2%)	(0.556, 0.555, 0.3%)	(0.553, 0.551, 0.4%)	(0.557, 0.556, 0.2%)	(0.555, 0.555, 0.0%)
0.4	(0.663, 0.660, 0.4%)	(0.665, 0.664, 0.3%)	(0.667, 0.666, 0.2%)	(0.667, 0.666, 0.1%)	(0.667, 0.664, 0.4%)
0.5	(0.807, 0.806, 0.1%)	(0.819, 0.818, 0.1%)	(0.827, 0.826, 0.1%)	(0.830, 0.829, 0.1%)	(0.835, 0.833, 0.2%)
0.6	(1.016, 1.013, 0.3%)	(1.052, 1.047, 0.5%)	(1.068, 1.065, 0.3%)	(1.083, 1.081, 0.2%)	(1.100, 1.099, 0.0%)
0.7	(1.322, 1.305, 1.3%)	(1.410, 1.390, 1.4%)	(1.482, 1.476, 0.4%)	(1.546, 1.543, 0.2%)	(1.597, 1.595, 0.1%)
0.8	(1.817, 1.777, 2.3%)	(2.057, 2.013, 2.2%)	(2.299, 2.273, 1.2%)	(2.675, 2.610, 2.5%)	(2.876, 2.862, 0.5%)

Table 8: Comparison of $(\mathbb{E}[S|\text{Bayes}], \mathbb{E}[S|\text{full MDP}], \Delta\%)$ in OPNET for $\rho_0 = 0.1$ with $C_1 = 1$ and $C_2 = 0.17$.

4. Conclusions

In this paper we study a model in which the sojourn time of foreground traffic is minimized in the presence of background traffic with the restriction that the different traffic types cannot be distinguished. We propose to adopt a Bayesian methodology to control the system. Our results shows that even though the system has fewer information than a fully observed system, the partial observability does not significantly compromise foreground sojourn times. Practically, the Bayesian setting is better applicable to a multi-network environment as full system observability cannot be assumed. The Bayesian method is very robust under different parameter settings.

The results raise a number of interesting questions for further research. First, in the current model we assume that the job sizes are exponentially distributed. An interesting question is how the Bayesian algorithm performs under more general job size distributions. For this purpose, our MDP approach could be extended by modeling the job size distributions by phase-type distributions, adding more challenges to the computational burden. Note that this comes with additional questions regarding the observability of the number of jobs in the different phases. Second, despite the fact that in practical deployments N is likely to be small, it is of theoretical interest to evaluate the performance and complexity of the Bayesian algorithm for larger N. Third, in this paper we primarily focused on the optimization of the foreground traffic, whereas in reality the performance of the background traffic may also be subject to QoS requirements. Inclusion of

such QoS constraints addresses an interesting and practically important topic.

5. Acknowledgments

The work reported in this paper was supported by the Netherlands Organisation for Scientific Research (NWO) under the Casimir project: Analysis of Distribution Strategies for Concurrent Access in Wireless Communication Networks.

References

[1] IEEE Standard 802.11n. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer specifications Enhancements for Higher Throughput. October 2009.

[2] S.C. Albright. Structural results for partially observable Markov decision processes. *Operations Research*, 27:1041–1053, 1979.

[3] R. Bellman. *Adaptive Control Processes: A Guided Tour*. Princeton University Press, 1961.

[4] R.I. Brafman. A heuristic variable grid solution method for POMDPs. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, pages 727–733, 1997.

[5] A.N. Burnetas and M.N. Katehakis. Optimal adaptive policies for Markov decision processes. *Mathematics of Operations Research*, 22:222–255, 1997.

[6] A.R. Cassandra. *Exact and Approximate Algorithms for Partially Observable Markov Decision Processes*. PhD thesis, Brown University, 1998.

[7] R. Chandra, P. Bahl, and P. Bahl. Multinet: Connecting to multiple IEEE 802.11 networks using a single wireless card. In *Proceedings of IEEE INFOCOM*, 2004.

[8] D. Cox. Fundamental limitations on the data rate in wireless systems. *IEEE Communications Magazine*, 46(12):16–17, 2008.

[9] J. Duncanson. Inverse multiplexing. *IEEE Communications Magazine*, 32(4):34–41, 1994.

[10] FCC. Report of the spectrum efficiency working group. Technical report, Federal Communications Commission Spectrum Policy Task Force, November 2002.

[11] Internet Engineering Task Force. Multipath tcp (mptcp) charter, April 2011. `http://datatracker.ietf.org/wg/mptcp/charter/`.

[12] C. Gkantsidis, M. Ammar, and E. Zegura. On the effect of large-scale deployment of parallel downloading. In *WIAPP '03: Proceedings of the Third IEEE Workshop on Internet Applications*, page 79, Washington, DC, USA, 2003.

[13] Y. Hasegawa, I. Yamaguchi, T. Hama, H. Shimonishi, and T. Murase. Deployable multipath communication scheme with sufficient performance data distribution method. *Computer Communications*, 30(17):3285–3292, 2007.

[14] M. Hauskrecht. *Planning and Control in Stochastic Domains with Imperfect Information*. PhD thesis, Massachusetts Institute of Technology, 1997.

[15] K.M. van Hee. *Bayesian Control of Markov Chains*. PhD thesis, Technical University of Eindhoven, 1978.

[16] O. Hernández-Lerma and J.B. Lasserre. *Discrete-Time Markov Control Processes: Basic Optimality Criteria*. Springer-Verlag, 1996.

[17] G.J. Hoekstra and F.J.M Panken. Increasing throughput of data applications on heterogeneous wireless access networks. In *Proceedings 12th IEEE Symposium on Communication and Vehicular Technology in the Benelux*, 2005.

[18] G.J. Hoekstra and R.D. van der Mei. Effective load for flow-level performance modelling of file transfers in wireless lans. *Computer Communications*, 33(16):1972–1981, 2010.

[19] H.Y. Hsieh and R. Sivakumar. A transport layer approach for achieving aggregate bandwidths on multi-homed mobile hosts. *Wireless Networks*, 11(1):99–114, January 2005.

[20] OPNET Technologies Inc. Opnet modeler, May 2009. `http://www.opnet.com/solutions/network_rd/modeler.html`.

[21] G.P. Koudouris, R. Agüero, E. Alexandri, J. Choque, K. Dimou, H.R. Karimi, H. Lederer, J. Sachs, and R. Sigle. Generic link layer functionality for multi-radio access networks. In *Proceedings 14th IST Mobile and Wireless Communications Summit*, 2005.

[22] P.R. Kumar. A survey of some results in stochastic adaptive control. *SIAM Journal of Control and Optimization*, 23:329–380, 1985.

[23] M. Littman, A. Cassandra, and L. Kaelbling. Learning policies for partially observable environments: Schaling up. In A. Prieditis and S. Russell, editors, *Proceedings of the Twelfth International Conference on Machine Learning*, pages 362–370, 1995.

[24] J.A. Loeve. *Markov Decision Chains with Partial Information*. PhD thesis, Leiden University, 1995.

[25] W.S. Lovejoy. A survey of algorithmic methods for partially observed Markov decision processes. *Annals of Operations Research*, 28:47–66, 1991.

[26] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow. TCP Selective Acknowledgment Options. RFC 2018, Internet Engineering Task Force, October 1996.

[27] G.E. Monahan. A survey of partially observable Markov decision processes: Theory, models, and algorithms. *Management Science*, 28:1–16, 1982.

[28] C.H. Papadimitriou and J.N. Tsitsiklis. The complexity of Markov decision processes. *Mathematics of Operations Research*, 12(3):441–450, 1987.

[29] R. Parr and S. Russell. Approximating optimal policies for partially observable stochastic domains. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1088–1094, 1995.

[30] M.L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 1994.

[31] P. Rodriguez, A. Kirpal, and E. Biersack. Parallel-access for mirror sites in the internet. In *INFOCOM*, pages 864–873, 2000.

[32] D. Sarkar, P.D. Amer, and R. Stewart. Guest editorial: Concurrent multipath transport. *Computer Communications*, 30(17):3215–3217, 2007.

[33] C.C. White III. A survey of solution techniques for the partially observed Markov decision process. *Annals of Operations Research*, 32:215–230, 1991.

[34] N.L. Zhang and W. Liu. Region-based approximations for planning in stochastic domains. In *Proceedings of the Thirteenth Annual Conference on Uncertainty in Artificial Intelligence*, pages 472–480, 1997.