Optimal Concurrent Access Strategies in Mobile Communication Networks

Sandjai Bhulai [1,3], Gerard Hoekstra [2,3], and Rob van der Mei [3,1]

***Abstract*—Current wireless channel capacities are closely approaching the theoretical limit. Hence, further capacity improvements from complex signal processing schemes may only gain modest improvements. Multi-path communication approaches, however, combine the benefits of higher performance and reliability by exploiting the concurrent usage of multiple communication networks in areas that are covered by a multitude of wireless access networks. So far, little is known on how to effectively take advantage of this potential.**

Motivated by this, we consider parallel communication networks that handle two types of traffic: foreground and background. The foreground traffic stream of files should be directed to the network that requires the least time to transfer the file. The background streams are always directed to the same network. It is not clear up front how to select the appropriate network for each foreground stream. This may be performed by a static selection policy, based on the expected load of the networks. However, a dynamic policy that accounts for the network status may prove more elegant and better performing.

We first propose a dynamic model that optimally assigns the foreground traffic to the available networks based on the number of fore- and background streams in both networks. However, in practice all traffic streams may be served by one application server. Thus, it may not be feasible to distinguish foreground from background traffic streams. This limitation is accounted for in our second, partial observation model that considers limited observability for dynamic network selection. We compare these static and dynamic models to each other and to the well-known Join the Shortest Queue (JSQ) model. The results are illustrated by extensive numerical experiments.

***Index Terms*—Concurrent access, Markov decision processes, optimal control, partial observation model, processor-sharing queues.**

I. INTRODUCTION

THE fundamental limit on wireless channel capacity is closely approached by many of today's wireless networks, which leaves complex signal processing techniques room for only modest improvements [1]. In areas covered by a multitude of wireless access networks the concurrent use of those networks, to which we refer to as concurrent access, to realize high-capacity enhancements becomes an interesting option to respond to the sustained growth of wireless communications. Concurrent access may aggregate high capacity communication means over lower capacity networks to improve the reliability and performance of communication towards applications. Consequently, large capacity improvements are within reach because the frequency spectrum is regulated among various frequency bands and corresponding communication network standards, and the overall spectrum usage remains to be relatively low over a wide range of frequencies [2]. Despite the enormous potential for performance improvement, only little is known about how to fully exploit this potential.

In the literature on telecommunication systems, the concurrent use of multiple network resources in parallel was already described for a Public Switched Digital Network (PSDN) [3]. Here 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 IEEE802.11n standard [4], at the physical layer or by switching datagrams at the link layer [5], [6], and also by using multiple TCP sessions in parallel to a fileserver [7]. 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 [8], but in general significant performance improvements can be realized [9], [10], [11]. Under these circumstances of using a combination of different network types, in particular the transport layer-approaches, have shown their applicability [11] as they allow appropriate link layer adaptations for each TCP session.

Although the technology seems in place to realize concurrent access, many problems arise in practice in the area of retransmission strategies, resequencing, buffer control techniques, and efficiently scheduling the data traffic among the various paths accordingly. Application traffic carried by a reliable transport protocol, such as TCP or SCTP, may also experience the drawback that, in addition to application layer interactions, time is consumed by connection setup handshakes and increasing the congestion window size before finally getting into steady state. If these actions need to be performed for multiple connections, the time required to get the TCP sessions in a state in which efficient scheduling can be properly based on the monitored session variables (e.g., observed round trip time, link capacity) may disqualify this method when also considering its implications on the added complexity to end nodes.

A main requirement for the widespread use of traffic splitting algorithms for concurrent access is that the algorithms are simple, yet effective. Motivated by this we consider a network

[1] VU University Amsterdam, Faculty of Sciences, De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands.

[2] Thales, Department of Innovation Research & Technology, Bestevaer 46, 1271 ZA Huizen, The Netherlands.

[3] Centre for Mathematics and Computer Science (CWI), Probability and Stochastic Networks, Science Park 123, 1098 XG Amsterdam, The Netherlands.

in which a (file download) application server responds to file download requests from mobile client devices by choosing the most suitable network path that connects both devices. This greatly reduces the complexity in end nodes compared to other approaches because the monitoring and the scheduling of individual packets over multiple links for multiple ongoing sessions per node is not needed. Instead, the application server is aware of the ongoing transfers in each network, which is the network status information where our optimization is based upon. We distinguish two types of network status information; the first in which the foreground transfers in all networks can be distinguished from the background and the second in which only the total number of transfers for each network can be observed.

It is not clear up front how one can take advantage of the availability of the different networks in the presence of background traffic, which is assumed here to consist of other file downloads from devices that can only use one network. Hence, there is a need to derive optimal concurrent access strategies to compare the different assignment strategies.

We study this traffic-selection problem in a queueing theoretical context and model the concerned communication networks as Processor Sharing (PS) nodes and the file transfers as jobs that need to be processed by the nodes. PS-based queueing models are applicable to a variety of communication networks (see [12], [13], [14]), including CDMA 1xEV-DO, WLAN, and UMTS-HSDPA. In fact, PS models can actually model file transfers over WLANs accurately [15], hence taking into account the complex dynamics of the file transfer application and its underlying protocol-stack, including their interactions.

In this paper we study dynamic concurrent access strategies, i.e., adapting to the current network status, that require only the number of download portions in progress (known by the application server) and are simple to enforce by deciding upon the assignment over the networks once. In particular, we study two models. First, we consider the model in which arriving jobs can be sent to a network based on the observable fore- and background streams. Second, a model is considered in which only the total number of ongoing streams is known, and Bayesian learning is applied to select the best network, based on the known state information. We derive the optimal dynamic network selection policy for the first and second model and compare the models to a simple static selection model and the dynamic JSQ-model. The results are illustrated by extensive numerical experiments.

The paper is organized as follows. In Section II, we formulate the concurrent access problem. Then, we continue with the the dynamic selection model in Section III, followed by the Bayesian selection model in Section IV. Finally, we compare the different models in Section V, which are followed by final conclusions in Section VI.

II. Problem Formulation

In this section we describe the concurrent access problem in greater detail. We model k 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 $k+1$ streams of jobs in the system. Stream i generates a stream of jobs for server i for $i = 1, \dots, k$. Stream 0 generates a stream of jobs for which the jobs can be sent to either server 1 up to server k. Hence, streams 1 to k 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 $\lambda_0, \dots, \lambda_k$, respectively.

After a job enters the system, it demands service from the system. We assume that the service times follow a general distribution with mean service time β_i for $i = 0, \dots, k$. Then, the occupation rates ρ_i are defined by $\rho_i = \lambda_i \beta_i$. Based on the above information, there is a central decision maker that has to decide on the distribution of the foreground jobs over the k 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 $\mathbb{E}N$, the expected average number of foreground jobs in the system. Note that this is directly related to the sojourn times of the foreground traffic.

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

III. The dynamic server selection model

In this section 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 $\mathcal{S} = \mathbb{N}_0^{2k} = \{0, 1, 2, \dots\}^{2k}$. A tuple $s = (x_1, \dots, x_k, y_1, \dots, y_k) \in \mathcal{S}$ denotes that there are x_i foreground jobs and y_i background jobs at server i for $i = 1, \dots, k$. For each job, the set of actions is given by $\mathcal{A} = \{1, \dots, k\}$, 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 $i = 0, \dots, k$. The transition rates p are thus given by $p(s, a, s')$ as follows: $p(s, a, s')$

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

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. Since we are interested in the number of foreground jobs in the system, we take the cost function c equal to $c(s) = x_1 + \dots + x_k$. The tuple $(\mathcal{S}, \mathcal{A}, p, c)$ defines the Markov decision problem.

Next, we uniformize the system (see Section 11.5 of [16]). To this end, we assume that the uniformization constant $\lambda_0 + \dots + \lambda_k + \sum_{i=1}^{k} \max\{\mu_0, \mu_i\} = 1$; 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 [17]). 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 [16]). 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

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

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 [16]). Table I shows the results when the above Markov decision problem is solved by iterating $V_{l+1} = TV_l$.

Note that it is fairly straightforward to extend the dynamic programming operator to account for other service-time distributions as well. For example, one could consider phase-type service distributions, which are dense in the class of all non-negative distributions [18]. This can be done by adding extra variables to the state space to count the number of jobs in all the phases. Thus, for an Erlang-n distribution, we would have a $2k \cdot n$-dimensional state space, as well as for a hyperexponential(n) service distribution. We have also done experiments with an Erlang-2 and a hyperexponential(2) distribution. The results are comparable to the results in Table I.

IV. The partial observation model

The dynamic server selection model uses a state description $(x_1,\ldots,x_k,y_1,\ldots,y_k)$ with $2k$ 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 $(z_1,\ldots,z_k)$ with $z_i = x_i + y_i$ for $i = 1,\ldots,k$. 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 $(z_1,\ldots,z_k)$ to $(x_1,\ldots,x_k,y_1,\ldots,y_k)$, so that (an estimate of the) full information is recovered. Note that it is not sufficient to create a mapping solely based on $(z_1,\ldots,z_k)$ 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 $(z_1,\ldots,z_k)$. Therefore, we will use Bayesian learning that takes into account the complete history of states in the estimation procedure.

We shall call $z = (z_1,\ldots,z_k) \in \mathbb{N}_0^k$ 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 $i = 1,\ldots,k$. The probability distribution u_i will serve the purpose of information about the states that cannot be observed; hence, $u = (u_1,\ldots,u_k)$ is called the information state. Note that the information state space is of high dimension, namely $\prod_{i=1}^k \{u_i \in [0,1]^{\mathbb{N}_0} \mid \sum_{x\in\mathbb{N}_0} u_i(x) = 1\}$.

Based on the observation and information 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 information 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

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

V. Comparison of the models

In this section we compare the different models with each other. For illustrative purposes we restrict ourselves to a system

TABLE I
DYNAMIC SERVER SELECTION: THE AVERAGE NUMBER OF FOREGROUND JOBS $\mathbb{E}N$

ρ_1/ρ_2	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.0	0.101	0.102	0.103	0.104	0.105	0.107	0.108	0.109	0.110	0.111
0.1	0.102	0.107	0.109	0.112	0.114	0.117	0.119	0.121	0.123	0.124
0.2	0.103	0.109	0.116	0.120	0.123	0.127	0.131	0.135	0.138	0.141
0.3	0.104	0.112	0.120	0.128	0.133	0.139	0.145	0.152	0.158	0.163
0.4	0.105	0.114	0.123	0.133	0.144	0.152	0.161	0.171	0.181	0.191
0.5	0.107	0.117	0.127	0.139	0.152	0.167	0.180	0.195	0.213	0.231
0.6	0.108	0.119	0.131	0.145	0.161	0.180	0.202	0.225	0.254	0.289
0.7	0.109	0.121	0.135	0.152	0.171	0.195	0.225	0.264	0.314	0.384
0.8	0.110	0.123	0.138	0.158	0.181	0.213	0.254	0.314	0.406	0.562
0.9	0.111	0.124	0.141	0.163	0.191	0.231	0.289	0.384	0.562	0.963

$\rho_0 = 0.1$ and $\beta_i = 1$ for $i = 0, 1, 2$

ρ_1/ρ_2	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.0	0.551	0.571	0.596	0.625	0.658	0.698	0.743	0.796	0.854	0.921
0.1	0.571	0.608	0.639	0.676	0.719	0.771	0.832	0.906	0.995	1.104
0.2	0.596	0.639	0.687	0.734	0.790	0.858	0.943	1.049	1.187	1.372
0.3	0.625	0.676	0.734	0.802	0.875	0.967	1.085	1.243	1.465	1.797
0.4	0.658	0.719	0.790	0.875	0.980	1.106	1.277	1.523	1.909	2.557
0.5	0.698	0.771	0.858	0.967	1.106	1.293	1.556	1.975	2.737	4.112
0.6	0.743	0.832	0.943	1.085	1.277	1.556	2.003	2.844	4.671	–
0.7	0.796	0.906	1.049	1.243	1.523	1.975	2.844	5.077	–	–
0.8	0.854	0.995	1.187	1.465	1.909	2.737	4.671	–	–	–
0.9	0.921	1.104	1.372	2.557	2.557	4.112	–	–	–	–

$\rho_0 = 0.5$ and $\beta_i = 1$ for $i = 0, 1, 2$

ρ_1/ρ_2	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.0	1.201	1.283	1.384	1.511	1.672	1.883	2.167	2.571	3.187	4.051
0.1	1.283	1.400	1.530	1.696	1.914	2.212	2.642	3.308	4.393	5.790
0.2	1.384	1.530	1.712	1.936	2.245	2.694	3.405	4.640	6.636	–
0.3	1.511	1.696	1.936	2.263	2.728	3.477	4.840	7.391	–	–
0.4	1.672	1.914	2.245	2.728	3.512	4.971	8.020	–	–	–
0.5	1.883	2.212	2.694	3.477	4.971	8.317	–	–	–	–
0.6	2.167	2.642	3.405	4.840	8.020	–	–	–	–	–
0.7	2.571	3.308	4.640	7.391	–	–	–	–	–	–
0.8	3.187	4.393	6.636	–	–	–	–	–	–	–
0.9	4.051	5.790	–	–	–	–	–	–	–	–

$\rho_0 = 0.9$ and $\beta_i = 1$ for $i = 0, 1, 2$

with two mobile networks. We first consider the static server selection model and the dynamic server selection model. For the static model we assume that the decision maker has control over a parameter $\alpha \in [0, 1]$ that determines the fraction of foreground jobs that are sent to server 1 (and thus a fraction $1 - \alpha$ is sent to server 2). For convenience, denote $\alpha_1 = \alpha$ and $\alpha_2 = 1 - \alpha$. Then server i has an effective occupation rate $\alpha_i \rho_0 + \rho_i$ for $i = 1, 2$. Note that for stability purposes we require that $\rho_0 < (1 - \rho_1) + (1 - \rho_2)$. Moreover, the two servers become independent, and from classical queueing theory results we have that

$$\mathbb{E}N = \frac{\alpha_1 \rho_0}{1 - (\alpha_1 \rho_0 + \rho_1)} + \frac{\alpha_2 \rho_0}{1 - (\alpha 2 \rho_0 + \rho_2)}.$$

Clearly, one can easily find the optimal value for α by minimizing the above expression. We have done this for several values of ρ_i for $i = 1, 2$. Table II shows the gain in using the optimal dynamic policy versus the optimal static policy computed as the relative difference

$$\frac{\mathbb{E}N_{\text{static}} - \mathbb{E}N_{\text{dynamic}}}{\mathbb{E}N_{\text{static}}} \cdot 100\%.$$

The results show that the gain is greatest when the system is balanced, i.e., when ρ_1 and ρ_2 are comparable. When the system is more skewed, i.e., ρ_1 and ρ_2 differ significantly from each other, the gains decrease. It is apparently the case that in a balanced system, the performance is more sensitive to the decisions that are made. Since the static policy is completely independent of any information in the system, it does not make the subtle choices that the dynamic policy makes.

We compare our dynamic policy also with the 'join the shortest queue' (JSQ) policy. Upon arrival of a job, the JSQ policy selects the server with the least number of jobs, i.e., the policy selects $\arg\min_{i=1,2}\{x_i + y_i\}$. Note that the JSQ policy can be used in absence of detailed information on the different job types. It only needs the total number of jobs present at each server to make a decision. Hence, this policy can be used when only the total number of transfers is given. Table III depicts the gain of the dynamic server selection policy over the JSQ policy. The table shows that the performance of the JSQ policy is very good for low to moderate occupation rates of the servers. The policy has a performance that is within 1% of the performance of the dynamic server selection policy. However, when the occupation rates increase, the discrepancy in performance between the JSQ policy and the dynamic server selection policy grows larger. Note that these experiments were done for $\beta_i = 1$ for $i = 0, 1, 2$. When the service rates of the different networks are chosen to be more skewed instead of comparable, the JSQ policy has even worse performance.

TABLE II
COMPARISON: GAIN IN STATIC VERSUS DYNAMIC SERVER SELECTION

ρ_1/ρ_2	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.0	4.40%	8.32%	8.10%	6.77%	5.38%	3.96%	2.72%	1.70%	0.84%	0.18%
0.1	8.32%	10.03%	12.47%	11.96%	9.60%	7.25%	5.04%	3.20%	1.68%	0.63%
0.2	8.10%	12.47%	14.60%	17.06%	15.93%	12.35%	8.92%	5.82%	3.32%	1.37%
0.3	6.77%	11.96%	17.06%	20.51%	22.32%	20.05%	14.94%	10.18%	6.02%	2.69%
0.4	5.38%	9.60%	15.93%	22.32%	26.58%	28.13%	23.92%	16.75%	10.20%	4.61%
0.5	3.96%	7.25%	12.35%	20.05%	28.13%	33.21%	34.94%	27.95%	17.61%	8.32%
0.6	2.72%	5.04%	8.92%	14.94%	23.92%	34.94%	41.60%	42.36%	30.92%	15.13%
0.7	1.70%	3.20%	5.82%	10.18%	16.75%	27.95%	42.36%	51.41%	51.38%	30.22%
0.8	0.84%	1.68%	3.32%	6.02%	10.20%	17.61%	30.92%	51.38%	64.38%	62.51%
0.9	0.18%	0.63%	1.37%	2.69%	4.61%	8.32%	15.13%	30.22%	62.51%	107.69%

$\rho_0 = 0.1$ and $\beta_i = 1$ for $i = 0, 1, 2$

ρ_1/ρ_2	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.0	21.14%	24.64%	27.72%	29.82%	31.06%	30.96%	29.28%	24.94%	17.08%	8.58%
0.1	24.64%	26.51%	30.05%	32.90%	34.90%	35.73%	35.17%	32.32%	25.42%	13.22%
0.2	27.72%	30.05%	32.26%	35.83%	38.60%	40.53%	41.21%	39.73%	34.78%	21.54%
0.3	29.82%	32.90%	35.83%	38.56%	42.27%	45.34%	47.23%	47.49%	44.77%	35.30%
0.4	31.06%	34.90%	38.60%	42.27%	45.58%	50.00%	53.48%	55.66%	55.91%	54.41%
0.5	30.96%	35.73%	40.53%	45.34%	50.00%	54.69%	59.82%	64.52%	70.31%	106.04%
0.6	29.28%	35.17%	41.21%	47.23%	53.48%	59.82%	66.42%	74.57%	7.04%	–
0.7	24.94%	32.32%	39.73%	47.49%	55.66%	64.52%	74.57%	96.99%	–	–
0.8	17.08%	25.42%	34.78%	44.77%	55.91%	70.31%	7.04%	–	–	–
0.9	8.58%	13.22%	21.54%	35.30%	54.41%	106.04%	–	–	–	–

$\rho_0 = 0.5$ and $\beta_i = 1$ for $i = 0, 1, 2$

ρ_1/ρ_2	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.0	36.21%	40.10%	43.65%	46.75%	49.46%	51.72%	53.66%	55.15%	56.27%	25.43%
0.1	40.10%	42.85%	46.81%	50.60%	54.07%	57.43%	60.87%	65.18%	76.24%	141.79%
0.2	43.65%	46.81%	50.20%	54.64%	59.09%	63.76%	69.47%	81.01%	141.11%	–
0.3	46.75%	50.60%	54.64%	59.07%	64.56%	70.96%	81.64%	132.23%	–	–
0.4	49.46%	54.07%	59.09%	64.56%	70.83%	80.59%	121.92%	–	–	–
0.5	51.72%	57.43%	63.76%	70.96%	80.59%	116.41%	–	–	–	–
0.6	53.66%	60.87%	69.47%	81.64%	121.92%	–	–	–	–	–
0.7	55.15%	65.18%	81.01%	132.23%	–	–	–	–	–	–
0.8	56.27%	76.24%	141.11%	–	–	–	–	–	–	–
0.9	25.43%	141.79%	–	–	–	–	–	–	–	–

$\rho_0 = 0.9$ and $\beta_i = 1$ for $i = 0, 1, 2$

Finally, we compare the Bayesian learning model to the dynamic server selection model. Note that the implementation of the Bayesian model is less straightforward than the other models that we have studied. The state space of the Bayesian model is not discrete due to the probability distributions that are part of the state space. Hence, we have restricted the possible values of the probability distributions to an equidistant grid on the interval $[0, 1]$. In doing so, the state space of the model becomes discrete and easier to implement with the sacrifice of a little accuracy. In our experiments, we chose to divide the interval $[0, 1]$ into 20 equidistant segments and adjusted the transitions of the probabilities accordingly such that the transitions were mapped back onto the grid. In Table IV one can see that the performance of the Bayesian model is superior to the JSQ policy. The optimal policy of the Bayesian model is within 1% of optimality (compared to the full information dynamic server selection model). For higher occupancy rates, the performance of the Bayesian model becomes significantly better than the performance of the JSQ policy. This has to do with the fact that the Bayesian model uses the complete history of actions to estimate the number of foreground and background jobs at the server. Even when the service rate parameters are chosen more skewed, the Bayesian model has a performance that is (usually) within 10% of optimality.

VI. CONCLUSIONS AND FUTURE WORK

In this paper we have studied the concurrent access problem in which a multi-antenna device can connect to multiple mobile networks. We answer the question how jobs of the device can utilize the networks in the best way. We do this by studying two settings: a setting in which different job types can be distinguished, and the setting in which only the total number of jobs at each server can be observed. For the former, we compare two policies: a static server selection policy (which does not need state information, but only the average server occupancy) and a dynamic server selection policy. The performance of the dynamic server selection policy is significantly better than the static policy. For the latter setting, we compare a join the shortest queue (JSQ) policy, and a Bayesian learning policy. Both policies have very good performance when the occupancy rates at the servers are low to moderate. However, when the occupancy rates grow larger, the performance of the Bayesian learning policy becomes significantly better than the JSQ policy.

There are a number of interesting avenues of further research. First, it would be interesting to study a job-split model in which jobs can be split and sent simultaneously over the different networks. The model is highly complex due to the

TABLE III
COMPARISON: GAIN IN JOIN THE SHORTEST QUEUE VERSUS DYNAMIC SERVER SELECTION

ρ_1/ρ_2	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.0	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
0.1	0.00%	0.01%	0.01%	0.01%	0.01%	0.01%	0.02%	0.04%	0.05%	0.12%
0.2	0.00%	0.01%	0.01%	0.03%	0.04%	0.04%	0.05%	0.07%	0.12%	–
0.3	0.00%	0.01%	0.03%	0.03%	0.05%	0.07%	0.08%	0.11%	–	–
0.4	0.00%	0.01%	0.04%	0.05%	0.05%	0.07%	0.10%	–	–	–
0.5	0.00%	0.01%	0.04%	0.07%	0.07%	0.08%	–	–	–	–
0.6	0.00%	0.02%	0.05%	0.08%	0.10%	–	–	–	–	–
0.7	0.00%	0.04%	0.07%	0.11%	–	–	–	–	–	–
0.8	0.00%	0.05%	0.12%	–	–	–	–	–	–	–
0.9	0.00%	0.12%	–	–	–	–	–	–	–	–

$\rho_0 = 0.1$ and $\beta_i = 1$ for $i = 0, 1, 2$

ρ_1/ρ_2	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.0	0.00%	0.00%	0.02%	0.06%	0.10%	0.14%	0.15%	0.19%	0.21%	0.24%
0.1	0.00%	0.02%	0.04%	0.09%	0.15%	0.21%	0.26%	0.32%	0.34%	0.85%
0.2	0.02%	0.04%	0.07%	0.13%	0.21%	0.30%	0.40%	0.50%	0.79%	–
0.3	0.06%	0.09%	0.13%	0.18%	0.27%	0.39%	0.55%	0.75%	–	–
0.4	0.10%	0.15%	0.21%	0.27%	0.34%	0.47%	0.68%	–	–	–
0.5	0.14%	0.21%	0.30%	0.39%	0.47%	0.58%	–	–	–	–
0.6	0.15%	0.26%	0.40%	0.55%	0.68%	–	–	–	–	–
0.7	0.19%	0.32%	0.50%	0.75%	–	–	–	–	–	–
0.8	0.21%	0.34%	0.79%	–	–	–	–	–	–	–
0.9	0.24%	0.85%	–	–	–	–	–	–	–	–

$\rho_0 = 0.5$ and $\beta_i = 1$ for $i = 0, 1, 2$

ρ_1/ρ_2	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.0	0.00%	0.02%	0.08%	0.22%	0.47%	0.87%	1.54%	2.73%	5.18%	11.87%
0.1	0.02%	0.04%	0.11%	0.25%	0.53%	1.05%	2.03%	4.15%	10.36%	28.39%
0.2	0.08%	0.11%	0.18%	0.32%	0.61%	1.25%	2.75%	7.46%	25.02%	–
0.3	0.22%	0.25%	0.32%	0.43%	0.70%	1.49%	4.51%	19.51%	–	–
0.4	0.47%	0.53%	0.61%	0.70%	0.93%	2.44%	14.81%	–	–	–
0.5	0.87%	1.05%	1.25%	1.49%	2.44%	12.77%	–	–	–	–
0.6	1.54%	2.03%	2.75%	4.51%	14.81%	–	–	–	–	–
0.7	2.73%	4.15%	7.46%	19.51%	–	–	–	–	–	–
0.8	5.18%	10.36%	25.02%	–	–	–	–	–	–	–
0.9	11.87%	28.39%	–	–	–	–	–	–	–	–

$\rho_0 = 0.9$ and $\beta_i = 1$ for $i = 0, 1, 2$

dependence of the networks as a result of splitting a job. It is reasonable to think that a mixture of a light-traffic and a heavy-traffic approximation could lead to good results. Second, we assume that the traffic streams are stationary. However, it is more realistic to assume that they are not and are governed by a Markov modulated Poisson process (of which the parameters are generally not known). In combination with the statistical learning algorithm this addition could create an automated system in which the decision making is done autonomously.

VII. 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] D. Cox, "Fundamental limitations on the data rate in wireless systems," *IEEE Communications Magazine*, vol. 46, no. 12, pp. 16–17, 2008.

[2] Federal Communications Commission Spectrum Policy Task Force, "Report of the spectrum efficiency working group," FCC - Federal Communications Commission, Tech. Rep., November 2002.

[3] J. Duncanson, "Inverse multiplexing," *IEEE Communications Magazine*, vol. 32, no. 4, pp. 34–41, 1994.

[4] IEEE Standard 802.11n, "Part 11: Wireless LAN medium access control (MAC) and physical layer specifications enhancements for higher throughput," October 2009.

[5] R. Chandra, P. Bahl, and P. Bahl, "Connecting to multiple IEEE 802.11 networks using a single wireless card," in *Proceedings of IEEE INFOCOM*, 2004.

[6] G. Koudouris, R. Agüero, E. Alexandri, J. Choque, K. Dimou, H. 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.

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

[8] 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*. Washington, DC, USA: IEEE Computer Society, 2003, p. 79.

[9] Y. Hasegawa, I. Yamaguchi, T. Hama, H. Shimonishi, and T. Murase, "Deployable multi-path communication schemes with sufficient performance data distribution methods," *Computer Communications*, vol. 30, no. 17, pp. 3285–3292, 2007.

[10] G. Hoekstra and F. 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.

[11] H. Hsieh and R. Sivakumar, "A transport layer approach for achieving aggregate bandwidths on multi-homed hosts," in *MobiCom '02: Proceedings of the 8th annual international conference on mobile computing and networking*. New York, NY, USA: ACM, 2002, pp. 83–94.

[12] S. Borst, O. Boxma, and N. Hegde, "Sojourn times in finite-capacity processor-sharing queues," in *Proceedings NGI 2005 Conference*, 2005.

[13] Y. Wu, C. Williamson, and J. Luo, "On processor -sharing and its appli-

TABLE IV
COMPARISON: GAIN IN BAYESIAN MODEL VERSUS DYNAMIC SERVER SELECTION

ρ_1/ρ_2	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.0	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
0.1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.01%	0.01%	0.02%	0.05%
0.2	0.00%	0.00%	0.00%	0.01%	0.01%	0.02%	0.03%	0.05%	0.07%	–
0.3	0.00%	0.00%	0.01%	0.02%	0.03%	0.05%	0.06%	0.09%	–	–
0.4	0.00%	0.00%	0.01%	0.03%	0.04%	0.05%	0.07%	–	–	–
0.5	0.00%	0.00%	0.02%	0.05%	0.05%	0.07%	–	–	–	–
0.6	0.00%	0.01%	0.03%	0.06%	0.07%	–	–	–	–	–
0.7	0.00%	0.01%	0.05%	0.09%	–	–	–	–	–	–
0.8	0.00%	0.02%	0.07%	–	–	–	–	–	–	–
0.9	0.00%	0.05%	–	–	–	–	–	–	–	–

$\rho_0 = 0.1$ and $\beta_i = 1$ for $i = 0, 1, 2$

ρ_1/ρ_2	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.0	0.00%	0.00%	0.00%	0.01%	0.02%	0.04%	0.07%	0.08%	0.10%	0.11%
0.1	0.00%	0.00%	0.01%	0.03%	0.05%	0.09%	0.11%	0.13%	0.16%	0.21%
0.2	0.00%	0.01%	0.03%	0.06%	0.09%	0.13%	0.17%	0.22%	0.29%	–
0.3	0.01%	0.03%	0.06%	0.08%	0.12%	0.17%	0.21%	0.28%	–	–
0.4	0.02%	0.05%	0.09%	0.12%	0.17%	0.23%	0.30%	–	–	–
0.5	0.04%	0.09%	0.13%	0.17%	0.23%	0.32%	–	–	–	–
0.6	0.07%	0.11%	0.17%	0.21%	0.30%	–	–	–	–	–
0.7	0.08%	0.13%	0.22%	0.28%	–	–	–	–	–	–
0.8	0.10%	0.16%	0.29%	–	–	–	–	–	–	–
0.9	0.11%	0.21%	–	–	–	–	–	–	–	–

$\rho_0 = 0.5$ and $\beta_i = 1$ for $i = 0, 1, 2$

ρ_1/ρ_2	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.0	0.00%	0.00%	0.02%	0.06%	0.12%	0.21%	0.38%	0.55%	1.59%	3.73%
0.1	0.00%	0.02%	0.04%	0.08%	0.17%	0.31%	0.57%	1.15%	2.99%	7.23%
0.2	0.02%	0.04%	0.05%	0.09%	0.13%	0.39%	0.62%	1.51%	6.02%	–
0.3	0.06%	0.08%	0.09%	0.12%	0.18%	0.40%	1.20%	4.51%	–	–
0.4	0.12%	0.17%	0.13%	0.18%	0.21%	0.64%	3.01%	–	–	–
0.5	0.21%	0.31%	0.39%	0.40%	0.64%	4.07%	–	–	–	–
0.6	0.38%	0.57%	0.62%	1.20%	3.01%	–	–	–	–	–
0.7	0.55%	1.15%	1.51%	4.51%	–	–	–	–	–	–
0.8	1.59%	2.99%	6.02%	–	–	–	–	–	–	–
0.9	3.73%	7.23%	–	–	–	–	–	–	–	–

$\rho_0 = 0.9$ and $\beta_i = 1$ for $i = 0, 1, 2$

cations to cellular data network provisioning," *Performance Evaluation*, vol. 64, no. 9-12, pp. 892–908, 2007.

[14] R. Litjens, F. Roijers, J. van der Berg, R. Boucherie, and M. Fleuren, "Performance analysis of wireless LANs: an integrated packet/flow level approach," in *Proceedings of the 18th International Teletraffic Congress - ITC18*, Berlin, Germany, 2003, pp. 931–940.

[15] G. Hoekstra and R. van der Mei, "On the processor sharing of file transfers in wireless LANs," in *Proceedings of the 69th IEEE Vehicular Technology Conference, VTC*, Barcelona, Spain, April 2009, pp. 26–29.

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

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

[18] R. Schassberger, *Warteschlangen*. Springer-Verlag, 1973.