Traffic Splitting Policies in Parallel Queues with Concurrent Access: a Comparison

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

[1]Innovation Research & Technology, Thales Nederland B.V., Huizen, The Netherlands
[2]CWI, Department of Stochastics, Amsterdam, The Netherlands
[3]VU University Amsterdam, Department of Mathematics, The Netherlands

***Abstract*—Surrounded by a multitude of wireless networks, users can nowadays experience significant performance improvements when smartly combining multiple networks concurrently (e.g., for transferring files). This phenomenon is called Concurrent Access (CA). Some users, which are referred to as foreground (FG) users, are able to access and utilize multiple networks simultaneously. The traffic streams from the FG users are optimized over multiple networks, in the presence of background (BG) users that can use only one network. In the literature a variety of traffic splitting algorithms have been proposed, with a focus on improving the performance of the FG users, whereas the influence of smart traffic splitting on the performance experienced by the BG users, as well as the resulting splitting ratios over the different networks, have received hardly attention. In this paper, we evaluate and compare the performance of these algorithms with respect to three quality metrics: (1) the file transfer performance of the FG traffic, (2) the file transfer performance of the BG traffic, and (3) the traffic splitting ratios, i.e. the fractions of traffic that is sent over each of the access networks. Our simulations-based results provide a number of valuable insights in the pros and cons of the different job splitting and assignment algorithms.**

Keywords*—*Traffic Splitting, Processor Sharing, Concurrent Access, Flow-level Performance, File Splitting.

I. Introduction

Over the past two decades the use of wireless communication networks has grown at an unprecedented rate, and this growth is not likely to come to an end in the near future. Normal cellphones are replaced by smartphones that generate and consume vast amounts of data. The main barrier to the sustained growth is the fundamental limitation on increasing the data rate. In practical deployments scarce resources have to be shared among a multitude of users, each of which shows random behavior in terms of application usage and mobility. In the competitive markets of wireless communication services it is essential for network operators to deliver high-quality services at sharp prices. This raises the need for smart and differentiating methods to utilize, control and further optimize the scarce wireless network resources in a cost-efficient manner.

A promising solution to satisfy the increasing demand for high application data rates is the concurrent use of multiple wireless networks. On multi-homed devices, for example, the data rate available to the applications may strongly benefit from the overlapping coverage of a wide range of wireless access technologies that operate in different frequency bands and already achieve very high spectral efficiencies. The approach to benefit from accessing multiple networks simultaneously, called concurrent access (CA), potentially delivers significant advantages, including performance improvement and increased robustness. Before carrying out this research, however, little was known about how to effectively exploit this enormous potential by smartly splitting traffic streams over multiple concurrent wireless networks. Currently, the efficient use of multiple networks concurrently is an active area of both research [1] and standardisation efforts [2]. This is the motivation for the research pursued in this paper.

In the context of communication 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. Many different forms of parallelism occur throughout different protocol layers in communication systems to enhance reliability, e.g., protecting channels that transfer voice signaling using the Stream Control Transmission Protocol (SCTP) [4], or to increase network performance for Wireless LANs (WLANs) using multiple antennas in the IEEE 802.11n and IEEE 802.11ac standard [5], [6].

Many research efforts that are focused on combining the capacity of multiple networks concentrate on the link layer, the transport layer and on the application layer of communication systems. At the *data link layer*, methods have been proposed for switching between several homogeneous networks [7] and scheduling over heterogeneous networks [8], [9]. However, methods at this layer require modifications for each different network interface that needs to be supported and, moreover, switching the data segments of the same Transport Control Protocol (TCP) session over different network links, even in homogeneous networks, adversely affects TCP performance [7], [10]. At the *transport layer*, two main areas can be distinguished: one concentrating on the use of SCTP (e.g., see [11], [12], [13]) and the other on using or modifying TCP. Within the IETF, SCTP has evolved from a transport protocol for voice-signaling traffic into one that allows various types of information to be transported over different network paths. It needs to be pointed out that the functionality for efficiently using concurrent paths is not considered by the standard, meaning that distributing and re-sequencing the data should

978-0-9836283-9-2

be implemented separately, and that the flow- and congestion control mechanism is the same for the possibly different networks used in parallel, which is not in the interest of overall efficient link utilization nor application performance [10], [14]. Several proposals to modify, use and extend TCP for exploiting multiple networks have appeared. The most prominent being a modification of TCP, called pTCP [10], later followed by mTCP [14] and MPTCP [1] that aimed at enhancing TCP to be capable of using multiple networks concurrently. In [15] a simple, yet efficient and easily deployable traffic-splitting algorithm for TCP-based networks is presented and evaluated on its FG performance, i.e., the performance of the traffic streams that are subjected to the traffic splitting policies.

On the theoretical side, there is a wealth of literature on performance models and analysis. Processor Sharing (PS) models have been successfully applied to model TCP-based file transfers over a wide range of different network standards that are commonly used, from cellular [16], [17] to WLANs [18], [19], [20]. The complex dynamics of multiple protocol layers can be modelled to obtain very accurate download response time predictions over a single network [19]. Despite the applicability of PS-based models to real communication networks, little is known on PS-based models suitable for modeling the use of multiple networks concurrently. In a queueing-theoretical context, the distribution and re-assembly of tasks into subtasks are typically modeled by fork-join constructions [21]. In cases where the processing times of the subtasks are independent, exact or numerical analysis is relatively simple (e.g., [22]), whereas the inclusion of dependent processing times (e.g., due to queueing or job splitting) typically leads to very complex analysis (e.g., [23], [24]) and no closed-form solution exists. In [25] and [26], the author analyzes a similar model but with FCFS queues and with probabilistic splitting. We further refer to Altman et al. [27], who consider routing policies in a distributed versus centralized environment. In general our queueing model falls within the framework of fork-join queueing networks, see [28] for an extensive overview.

For PS-queues that process the tasks of a job in parallel, the complex correlation structure between the sojourn times at the PS-queues makes an exact detailed mathematical analysis of the model impossible. As a result, the available literature on queueing models with regard to traffic-splitting is not widely adopted and hence leaves a gap between theory and practice. As an exception, Key et al. investigate the efficiency of combining multipath routing and congestion control in TCP-based networks. In [29] they show that under certain conditions the allocation of flows to paths is optimal and independent of the flow control algorithm used. In [30] it is shown that with RTT bias uncoordinated control can lead to inefficient equilibria, while without RTT bias, both coordinated and uncoordinated Nash equilibria correspond to desirable welfare maximizing states.

The literature leaves a clear gap between theory and practice: the existing theory provides important fundamental insight, but the models often rely on simplifying assumptions that cannot be met by practical deployments. On the practical side, the research efforts on splitting algorithms and implementations were concentrated at increasing the throughput on multiple networks (often in the absence of BG traffic) without having a notion of the user-level performance in the presence of BG traffic. In fact, the impact that CA methods in networks have on the performance of other traffic is a subject that is not well-known in the literature. To this end, in this paper we concentrate on the *overall* performance impact (i.e., the impact on *all* traffic streams) of various CA methods that can be applied to distributing application data among parallel networks.

The remainder of the paper is organized as follows. In Section II we describe the model and introduce the notation. More specifically, we consider a CA network that consists of N parallel communication networks, each of which is modeled as a PS-queue that handles two types of traffic: FG and BG streams. In Section III we review the CA methods for applying static/dynamic splitting and assignment of jobs in a queueing network of parallel queues. In Section IV we discuss the results of extensive simulations conducted with the presented methods and compare the performance of the overall set of traffic streams. Finally, in Section V we summarize our observations.

II. The Concurrent Access Network Model

In [19], [20] a new concept is introduced for modeling data traffic flows in a communication network by jobs that are processed by a Processor Sharing (PS) queue. The influence of complex combined dynamics and protocol overhead of multiple communication layers on data traffic flows in real networks can be accounted for in an explicit expression for a single parameter which will be called the *effective service time*. Based on the effective service time, the *effective load* is defined to describe the performance of data flows in a network with an $M/G/1$ PS model. Extensive validation by means of simulations and experiments conducted with network equipment reveals that accurate predictions for the mean download time [19] and its distribution [20] can be obtained for a wide range of parameter combinations. Additionally, this model can be used to parameterize a PS-based queueing network, such that the outcomes correspond to the performance of real networks or vice versa. Using the properties of the model for effective service time we analyze the concurrent access network model with a queueing network that consists of N parallel PS nodes (Figure 1). There are $N+1$ traffic streams: a single stream of FG jobs (called class-0 jobs) and N streams of BG jobs (called class-i jobs, for $i = 1, \ldots, N$). Class-i jobs arrive according to independent Poisson processes with rates λ_i, the service times B for all classes are independent samples from a general distribution with finite k-moment $\beta^{(k)} = \mathbb{E}[B^k]$, for $k = 1, 2, \ldots$, and the corresponding load offered to the system is $\rho_i = \lambda_i \beta^{(1)}$, $i = 0, 1, \ldots, N$. It is assumed that PS-queue i operates at rate $c_i = 1$. Based on the model for effective service time the network may be represented by PS-queues and the files by jobs to study traffic splitting methods in a queueing-theoretical context.

Let S_i denote the sojourn time of an arbitrary class-i job in the system, for $i = 0, 1, \ldots, N$. In the next section we discuss a number of traffic splitting methods, which are aimed to minimize the expected value of the sojourn time of FG jobs (i.e., $E[S_0]$), with respect to the choice of the splitting policy.

978-0-9836283-9-2

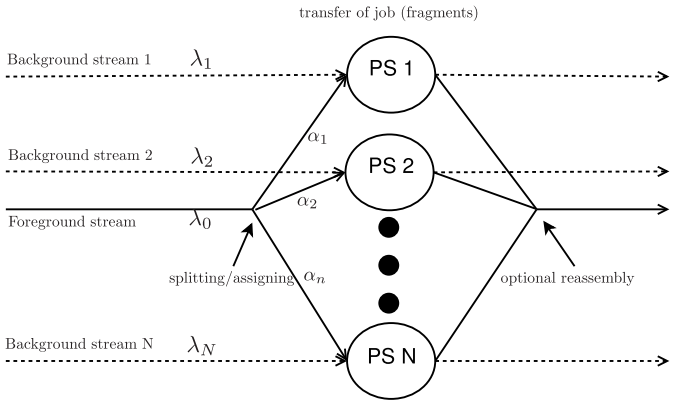

Fig. 1: The concurrent access network model.

III. The Concurrent Access Methods

In this section we briefly discuss the different traffic splitting methods under consideration.

A. Static splitting

In the case of having static splitting, each FG job is fragmented (split) upon arrival according to a static, splitting rule $\underline{\alpha} = (\alpha_1, \ldots, \alpha_N)$ where $\sum_{i=1}^N \alpha_i = 1$ and $\alpha_i \geq 0$, $i = 1, \ldots, N$. After splitting, the fragments are routed to their corresponding queues. Thus, when a job of size B arrives at stream 0, a fragment of size $\alpha_i B$ is directed to each queue i. Once all fragments complete their service, the fragments must be reunited (indicated in Figure 1 as the optional reassembly), and this completes the processing of the job that models the file transfer through the N communication networks. When we consider a tagged FG job J that arrives to a network in steady-state, we denote the sojourn time of its i'th fragment operating under the splitting rule $\underline{\alpha}$ by $S_i^{\underline{\alpha}}$; this is the time it takes the fragment to complete service at queue i. Denote $\underline{S}_{\underline{\alpha}} = (S_1^{\underline{\alpha}}, \ldots, S_N^{\underline{\alpha}})$. The sojourn time of J through the network is defined as $S_0^{\underline{\alpha}} := \max \underline{S}_{\underline{\alpha}}$.

When optimizing static splitting, the objective is to choose a splitting rule $\underline{\alpha}^*$ such that $E[S_0^{\underline{\alpha}}]$ is minimal over all feasible values of $\underline{\alpha}$. In this context, note that in general the sojourn times of a job's fragments in the different PS-queues $S_1^{\underline{\alpha}}, \ldots, S_N^{\underline{\alpha}}$ are generally correlated, and do not allow for an exact detailed analysis. For the case of light foreground traffic and regularly varying foreground job-size distributions, a Reduced Load Approximation (RLA) can be obtained for the sojourn times, similar to that of a single PS-queue. It is proven in [31] that the following simple static splitting rule provides tail-optimal performance with respect to the foreground sojourn time:

$$\alpha_i^* := \frac{1-\rho_i}{\sum_{j=1}^N (1-\rho_j)} \quad (i = 1, \ldots, N), \tag{1}$$

to obtain a splitting rule for all queues denoted by:

$$\underline{\alpha}^* := (\alpha_1^*, \ldots, \alpha_N^*). \tag{2}$$

This rule divides FG jobs in parts that are proportional to the remaining capacity in each queue. Note that $1 - \rho_i$ is the unutilized capacity of queue i due to BG traffic and $\sum_{j=1}^N (1-\rho_j)$ is the total unutilized capacity due to BG traffic. Note that $\underline{\alpha}^*$ does not depend on ρ_0. We refer to [31] for a more detailed discussion about static splitting.

In [32] an approximation for a fixed splitting rule is presented that minimizes the transfer time with respect to the mean foreground response time. The RLA leads to highly accurate approximations of $\underline{\alpha}^*$ if either $\rho_0 \approx 0$ or if the queues are fairly equally loaded, but that it may become inaccurate when one or more queues are heavily loaded while others are not. This requires a refinement of the RLA to improve its accuracy for asymmetric background-load scenarios, which may be realized by combining the tail-optimal splitting rule with known heavy-traffic asymptotics (HTA) for multi-class PS models. For a given splitting rule α the utilization of queue i is denoted by $\xi_i := \rho_i + \alpha_i \rho_0$ and the following approximation for the expected foreground sojourn time, $\mathbb{E}[S_{HTA,0}^{\underline{\alpha}}]$, can be used:

$$\mathbb{E}[S_{HTA,0}^{\underline{\alpha}}] \approx \beta^{(1)} \sum_{k=1}^{N} (-1)^{k+1} \sum_{(i_1,\ldots,i_k)\in S_k} \frac{1}{\frac{1-\xi_{i_1}}{\alpha_{i_1}} + \cdots + \frac{1-\xi_{i_N}}{\alpha_{i_N}}}, \tag{3}$$

where

$$S_k := \{(i_1, \ldots, i_k) : i_1, \ldots, i_k \in \{1, \ldots, N\}, i_1 < i_2 < \cdots < i_k\}. \tag{4}$$

Denote by $\underline{\alpha}_{HTA}^* = (\alpha_{HTA,1}^*, \ldots, \alpha_{HTA,N}^*)$ the splitting rule that minimizes $\mathbb{E}[S_{HTA,0}^{\underline{\alpha}}]$ among all $\underline{\alpha} \in A$, i.e.,

$$\mathbb{E}[S_{HTA,0}^{\underline{\alpha}^*}] = \min_{\underline{\alpha}\in A} \mathbb{E}[S_{HTA,0}^{\underline{\alpha}}]. \tag{5}$$

The optimal split $\underline{\alpha}_{HTA}^*$ can then be approximated by evaluating (3) over all $\underline{\alpha} \in A$, or by some non-linear optimization method. In practice, this causes no problem as N is not too large. Note that in the context of concurrent access for wireless networks, which was the main motivation for this study, N is indeed small, say 2 or 3. The RLA and HTA approaches can be combined with a Composed-Split Approximation (CSA), denoted $\underline{\alpha}_{CSA}^* = (\alpha_{CSA,1}^*, \ldots, \alpha_{CSA,N}^*)$, where for $i = 1, \ldots, N$,

$$\alpha_{CSA,i}^* = (1 - \kappa_i)\alpha_{RLA,i}^* + \kappa_i \alpha_{HTA,i}^*, \tag{6}$$

$$\text{with } \kappa_i := \max\{\rho_1, \ldots, \rho_N\}, \tag{7}$$

and where $\alpha_{RLA,i}^*$ is given in (1), and where $\alpha_{HTA,i}^*$ can be obtained from (3)-(5). The interpolation factor κ_i is taken such that κ_i is independent of $\underline{\alpha}$ (and of i), while for light-traffic scenarios the RLA is dominant and for heavy-load scenarios the HTA is dominant. We refer to [32] for a more detailed discussion about this approximation for minimizing the mean sojourn time with a static splitting policy.

B. Dynamic assignment

For the case of dynamic assignment, FG jobs are assigned to one of the available networks, in the presence of BG traffic streams. The problem is to develop a dynamic policy that minimizes the mean sojourn time of the FG traffic. The optimal policy generally depends on the information that is available to the decision maker. For example, in the case where the decision

maker has information about the numbers of FG and BG jobs at each of the access nodes, the optimal policy can be developed by solving a Markov decision problem. As another example, in some cases the decision maker does not have full information about the numbers of FG and BG jobs in each of the networks, but instead *only* has partial information on the *total* number of jobs at each of the networks. In [33], a Bayesian learning algorithm was proposed that splits a stream by optimally assigning entire jobs to different nodes. The optimality of the partial information algorithm is evaluated by comparing the performance of the algorithm with the 'ideal" performance of the optimal policy using full state information about the numbers of FG and BG jobs at the nodes. Experimental results show that the dynamic assignment, which is also referred to as the *Bayesian* assignment a delivers close to optimal performance over a wide range of parameter values. We refer to [33] for a more detailed discussion of dynamic assignment policies.

C. Dynamic splitting of FG jobs

In the case of dynamic splitting, FG jobs use the capacity of all nodes *simultaneously* in a fluid-like manner, using the (instantaneously) available capacity at all the N PS nodes; at any moment in time the capacity available at node i is equally shared amongst all FG jobs in the system and with the BG jobs at node i. The splitting operates without delay with infinitely small granularity and has perfect information about the number of FG and per-class BG jobs in the system. If upon arrival of a tagged FG job F there are k_0 other FG jobs in the system and k_i BG jobs at node i, then F obtains a fraction

$$f_i := \frac{1}{k_0 + 1 + k_i} \tag{8}$$

of the capacity of node i, for $i = 1, \ldots, N$. Note that in this way, the instantaneous total transmission speed that F receives equals $\sum_{i=1}^{N} f_i$, and that this speed changes during the course of the sojourn time of F in the system, as other jobs may come and go.

The model described above uses fluid-like splitting of FG jobs at infinitely fine time granularity. Therefore, the performance of the dynamic splitting model is optimal in the sense that the *synchronization delay* in the reassembly phase (which is usually encountered when splitting is done at coarse-grained granularity or with non-perfect or delayed information) is *zero*, while the FG jobs receive no more than their fair share of capacity at each of the nodes. It is evident that from the viewpoint of the FG traffic even better performance for FG jobs can be obtained by allowing unfair capacity sharing at the PS nodes in favor of FG traffic. We refer to [34] for a more detailed discussion on the dynamic splitting model.

D. Static assignment

The static assignment policy simply assigns *all* foreground traffic to the node with the lower background load (or to node one when the loads are equal).

E. Join-Shortest-Queue

The Join-Shortest-Queue (JSQ) policy assigns a FG job to the node with the smallest number of jobs present (ties are broken evenly).

IV. Comparison

In this section we compare the outcomes of the five CA strategies discussed above by simulations. For these strategies, we focus on the following three quality metrics: (1) mean sojourn time of FG jobs, (2) mean sojourn time of BG jobs, and (3) the traffic splitting ratios. In the following subsections the results of simulations are shown. To this end, we consider a network, where $N = 2$ and $\beta^{(1)} = 1$. The load of the FG traffic ρ_0 was varied from light ($\rho_0 = 0.1$) to mild ($\rho_0 = 0.5$), moderate ($\rho_0 = 0.9$) and heavy ($\rho_0 = 1.8$), and the BG loads ρ_1 and ρ_2 were varied as $0.1, 0.2, \ldots, 0.9$. Each simulation run is based on averages from 10^8 FG observations. For convenience, the BG traffic performance in both queues is consolidated into the following notion of the BG traffic performance:

$$\gamma = \frac{\mathbb{E}[S_1]\,\rho_1 + \mathbb{E}[S_2]\,\rho_2}{\rho_1 + \rho_2} \tag{9}$$

A. Light FG traffic load

Figure 2 and Table I show that under light FG traffic load, the traffic splitting/assignment ratios demonstrate very typical behavior for each strategy. The static assignment strategy simply selects the queue with the smallest BG traffic load and is a trivial example. When performing static job-splitting, the ratio remains close to half when the BG traffic loads become unequal and cause a very steep increase in the splitting ratio towards a highly unbalanced system. The ratio in which the jobs are assigned by the Join-the-Shortest Queue (JSQ) algorithm differ much from the other in that there seems to be a linear increase as the systems become more and more unequally loaded. Different from the other strategies, both the dynamic Bayesian assignment and the dynamic splitting policies show a bended curve as the queues become more and more unequally loaded. However, the dynamic splitting ratio shows a much steeper increase and remains far below the other strategies. The sojourn times that match the splitting/assignment ratios from Figure 2 are shown in Table II. From the perspective of the FG sojourn time, the dynamic traffic splitting outperforms all other strategies. As long as the overall BG traffic load is not very high, static assignment performs fairly well considering its simplicity. If the BG load in both queues increases, static assignment may yield very high sojourn times or may cause one of the queues to become unstable where other strategies continue to perform well.

Static splitting of light FG traffic leads to fairly good performance as long as both queues are lightly loaded. As the BG load on one of the queues is gradually increased, the performance of static splitting is very similar to static assignment for low to moderate overall BG traffic loads. The results for Bayesian assignment and JSQ are so close that they can hardly be distinguished. In particular under higher overall BG traffic loads the performance of dynamic assignment and JSQ lead to fairly good results. Clearly, there is no strategy that outperforms dynamic splitting. The relative difference to static splitting is not very large under very low overall BG traffic loads. If, however, the BG traffic loads increase the difference between dynamic splitting and static splitting becomes very large, up to the point where dynamic splitting is more than twice as fast. The impact of the traffic distribution strategies on the BG traffic performance is shown in Table III. Here it

TABLE I: Splitting/assignment ratios of light foreground traffic ($\rho_0 = 0.1$) for various CA strategies as a function of the background load.

CA Method	ρ_1 \ ρ_2	0.2	0.4	0.6	0.8
JSQ	0.1	0.55	0.65	0.76	0.87
dynamic assignment	0.1	0.84	0.87	0.90	0.96
static assignment	0.1	1.00	1.00	1.00	1.00
static split	0.1	0.50	0.64	0.78	0.92
dynamic split	0.1	0.52	0.57	0.64	0.74
JSQ	0.3		0.55	0.68	0.82
dynamic assignment	0.3		0.74	0.80	0.91
static assignment	0.3		1.00	1.00	1.00
static split	0.3		0.54	0.71	0.89
dynamic split	0.3		0.53	0.60	0.71
JSQ	0.5			0.57	0.74
dynamic assignment	0.5			0.67	0.80
static assignment	0.5			1.00	1.00
static split	0.5			0.59	0.83
dynamic split	0.5			0.54	0.65
JSQ	0.7				0.61
dynamic assignment	0.7				0.65
static assignment	0.7				1.00
static split	0.7				0.66
dynamic split	0.7				0.57

TABLE II: Mean sojourn times of light foreground traffic ($\rho_0 = 0.1$) for various CA strategies as a function of the background load.

CA Method	ρ_1 \ ρ_2	0.2	0.4	0.6	0.8
JSQ	0.1	1.11	1.19	1.25	1.28
dynamic assignment	0.1	1.10	1.14	1.19	1.23
static assignment	0.1	1.25	1.25	1.25	1.25
static split	0.1	0.71	0.88	1.03	1.16
dynamic split	0.1	0.60	0.66	0.75	0.89
JSQ	0.3		1.35	1.49	1.62
dynamic assignment	0.3		1.33	1.45	1.58
static assignment	0.3		1.67	1.67	1.67
static split	0.3		1.05	1.29	1.52
dynamic split	0.3		0.75	0.87	1.07
JSQ	0.5			1.82	2.16
dynamic assignment	0.5			1.80	2.13
static assignment	0.5			2.50	2.50
static split	0.5			1.68	2.15
dynamic split	0.5			1.05	1.35
JSQ	0.7				3.16
dynamic assignment	0.7				3.15
static assignment	0.7				5.00
static split	0.7				3.50
dynamic split	0.7				1.90

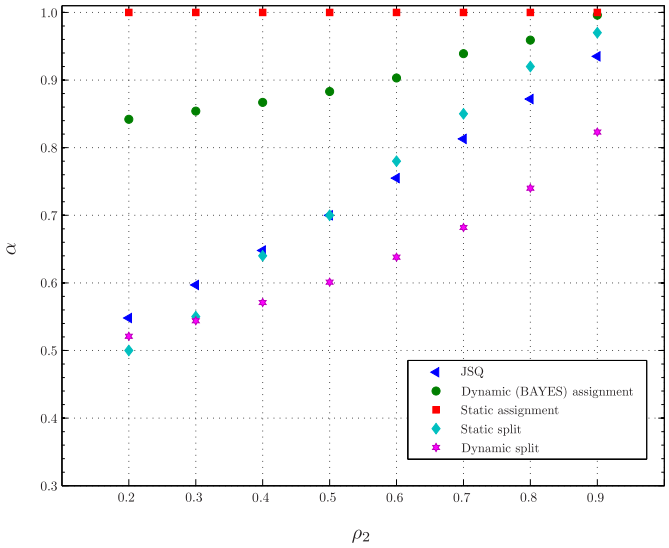

Fig. 2: Splitting/assignment ratios of light foreground traffic ($\rho_0 = 0.1$) and keeping the background traffic in network one constant ($\rho_1 = 0.1$) for various CA strategies as a function of the background load.

is shown that the relative differences between the BG traffic load is limited as long as the queues are not very unequally loaded with BG traffic and the overall BG traffic load is not too high. The overall BG traffic sojourn times is most favorable in the remaining cases for either static assignment low to medium BG traffic load) or in the case of dynamic assignment or JSQ. Under very high BG traffic load, dynamic splitting improves the FG traffic performance slightly at the expensive of the BG traffic compared to JSQ and dynamic (Bayesian) assignment. When considering the splitting ratios in Table I, dynamic splitting is directing much more service demand to the queue with the highest load. This consistent behavior can be observed throughout Table I. Static splitting performs worst at very high load, both with respect to the FG- and the BG performance. This may be explained by the high fluctuations in occupation level in both queues: splitting the FG traffic based on a long-term characteristic does not match the occupation levels at small time scales.

B. Mild FG traffic load

Table IV shows the results on the splitting/assignment ratios for a FG traffic load that is equal to $\rho_0 = 0.5$. As expected, the static assignment strategy has fewer combinations for which the system is stable. Considering the values of the other CA strategies it may seem to increase faster than for the light BG traffic load. This is not generally the case. For static splitting, the ratios are increasing much slower with an increasing BG load skewness when compared to light FG load. This effect is even more pronounced for Bayesian assignment.

The JSQ strategy increases its assignment ratios very modestly. An exception to this rule is the behavior of dynamic splitting, which is splitting the traffic such that, in comparison to light FG load, a higher portion of the FG traffic arrives at the queue that has a higher load. Due to the higher FG traffic load, the splitting ratios show smaller variations, in

TABLE III: Background traffic performance (γ) for various CA strategies with light foreground traffic ($\rho_0 = 0.1$) as a function of the background load.

CA Method	ρ_1 \ ρ_2	0.2	0.4	0.6	0.8
JSQ	0.1	1.24	1.60	2.36	4.64
dynamic assignment	0.1	1.24	1.59	2.34	4.60
static assignment	0.1	1.25	1.58	2.32	4.58
static split	0.1	1.28	1.66	2.44	4.77
dynamic split	0.1	1.26	1.64	2.44	4.91
JSQ	0.3		1.64	2.24	4.16
dynamic assignment	0.3		1.63	2.22	4.11
static assignment	0.3		1.67	2.22	4.09
static split	0.3		1.70	2.33	4.30
dynamic split	0.3		1.66	2.30	4.40
JSQ	0.5			2.43	4.09
dynamic assignment	0.5			2.42	4.07
static assignment	0.5			2.50	4.04
static split	0.5			2.55	4.29
dynamic split	0.5			2.48	4.31
JSQ	0.7				4.76
dynamic assignment	0.7				4.75
static assignment	0.7				5.00
static split	0.7				5.21
dynamic split	0.7				4.95

particular for higher BG load values. Again are the dynamic policies characterized by non-linear increases whereas the JSQ strategy is fairly linear. The impact on the FG sojourn times is visible in Table V, where the static assignment strategy is clearly performing much worse than others. Also the distance between these other strategies has increased in comparison to the case of light FG traffic load (in Table II). Table V also shows that for mild FG traffic load the difference between the JSQ strategy and dynamic Bayesian assignment becomes visible for higher BG load values. For the highest BG load depicted, the results are very similar to the ones illustrated in Table II. The results on the BG performance are shown in Table VI and demonstrate higher differences between the various strategies. It can be observed that static assignment of FG traffic does not lead to poor BG performance.

In fact, for low overall BG traffic loads, static assignment leads to the lowest BG sojourn times. Dynamic assignment using JSQ or Bayesian strategies leads to slightly higher sojourn times than those for static assignment. For higher overall BG loads the difference between Bayesian assignment and JSQ becomes visible and clearly Bayesian assignment delivers lower FG response times at the expense of the BG traffic performance. Dynamic splitting leads to relatively high BG sojourn times when both queues are exposed to very different BG traffic loads. In those cases dynamic splitting delivers higher BG sojourn times than static splitting. The latter strategy adversely affects the BG performance most prominently when the overall BG traffic load is high, an effect that is similar to the FG sojourn times in Table V.

C. Moderate FG traffic load

For a moderate FG traffic load, $\rho_0 = 0.9$ the splitting/assignment ratios are as expected: the differences between the values are again smaller in comparison to lower FG traffic

TABLE IV: Splitting/assignment ratios of mild foreground traffic ($\rho_0 = 0.5$) for various CA strategies as a function of the background load.

CA Method	ρ_1 \ ρ_2	0.2	0.4	0.6	0.8
JSQ	0.1	0.54	0.62	0.72	0.85
dynamic assignment	0.1	0.65	0.69	0.75	0.87
static assignment	0.1	1.00	1.00	1.00	1.00
static split	0.1	0.51	0.65	0.74	0.88
dynamic split	0.1	0.52	0.58	0.65	0.76
JSQ	0.3		0.546	0.65	0.79
dynamic assignment	0.3		0.60	0.67	0.80
static assignment	0.3		1.00	1.00	1.00
static split	0.3		0.54	0.67	0.83
dynamic split	0.3		0.53	0.61	0.73
JSQ	0.5			0.560	0.721
dynamic assignment	0.5			0.57	0.73
static assignment	0.5			1.00	1.00
static split	0.5			0.57	0.74
dynamic split	0.5			0.54	0.68

TABLE V: Mean sojourn times of mild foreground traffic ($\rho_0 = 0.5$) for various CA strategies as a function of the background load.

CA Method	ρ_1 \ ρ_2	0.2	0.4	0.6	0.8
JSQ	0.1	1.30	1.48	1.71	2.03
dynamic assignment	0.1	1.29	1.45	1.69	2.00
static assignment	0.1	2.50	2.50	2.50	2.50
static split	0.1	0.98	1.25	1.60	2.04
dynamic split	0.1	0.79	0.91	1.09	1.40
JSQ	0.3		1.78	2.22	3.00
dynamic assignment	0.3		1.77	2.20	2.96
static assignment	0.3		5.00	5.00	5.00
static split	0.3		1.62	2.21	3.234
dynamic split	0.3		1.09	1.38	1.97
JSQ	0.5			3.17	5.74
dynamic assignment	0.5			3.17	5.59
static assignment	0.5			unstable	unstable
static split	0.5			3.46	6.90
dynamic split	0.5			1.94	3.53

TABLE VI: Background traffic performance (γ) for various CA strategies with mild foreground traffic ($\rho_0 = 0.5$) as a function of the background load.

CA Method	ρ_1 \ ρ_2	0.2	0.4	0.6	0.8
JSQ	0.1	1.47	1.90	2.77	5.26
dynamic assignment	0.1	1.48	1.93	2.88	5.34
static assignment	0.1	1.67	1.83	2.50	4.72
static split	0.1	1.72	2.29	3.45	6.59
dynamic split	0.1	1.58	2.13	3.34	7.36
JSQ	0.3		2.09	2.94	5.41
dynamic assignment	0.3		2.12	3.04	5.74
static assignment	0.3		3.10	3.33	5.00
static split	0.3		2.54	3.75	7.28
dynamic split	0.3		2.28	3.41	7.59
JSQ	0.5			3.75	7.52
dynamic assignment	0.5			3.85	7.93
static assignment	0.5			unstable	unstable
static split	0.5			5.06	11.76
dynamic split	0.5			4.24	10.17

TABLE VII: Splitting/assignment ratios of moderate foreground traffic ($\rho_0 = 0.9$) for various CA strategies as a function of the background load.

CA Method	ρ_1 \ ρ_2	0.2	0.4	0.6	0.8
JSQ	0.1	0.53	0.61	0.70	0.82
dynamic assignment	0.1	0.56	0.61	0.70	0.84
static split	0.1	0.53	0.61	0.71	0.84
dynamic split	0.1	0.52	0.58	0.66	0.79
JSQ	0.3		0.54	0.64	
dynamic assignment	0.3		0.54	0.64	
static split	0.3		0.54	0.65	
dynamic split	0.3		0.53	0.62	

TABLE VIII: Mean sojourn times of moderate foreground traffic ($\rho_0 = 0.9$) for various CA strategies as a function of the background load.

CA Method	ρ_1 \ ρ_2	0.2	0.4	0.6	0.8
JSQ	0.1	1.72	2.18	3.04	5.54
dynamic assignment	0.1	1.74	2.19	3.00	5.20
static split	0.1	1.50	2.14	3.31	6.53
dynamic split	0.1	1.17	1.47	2.02	3.61
JSQ	0.3		3.09	5.68	
dynamic assignment	0.3		3.14	5.56	
static split	0.3		3.33	6.91	
dynamic split	0.3		2.07	3.69	

loads and the splitting ratios of dynamic splitting are lagging behind the other strategies. In the case of moderate FG traffic load static assignment does not lead to a stable system in the case of having a BG load equal to 0.1. The differences between static splitting and dynamic assignment using JSQ or Bayes are very small. Similar to what was observed for an increasing FG traffic load from light to mild intensities, dynamic splitting is the only strategy that is directing again more traffic towards the queue that has the highest BG load. In contrast, Bayesian assignment and static splitting send a smaller portion of the FG traffic to the queue with the highest BG load.

The JSQ strategy has the smallest sensitivity to the FG traffic intensity; slightly smaller amounts of jobs are assigned to the queue with the highest BG load. The resulting impact on the FG sojourn times are shown in Table V where it is clearly shown that Bayesian assignment outperforms the JSQ strategy most notably for unbalanced systems. Again, the sojourn times of the FG traffic are significantly smaller than for all other strategies. The resulting BG traffic sojourn times demonstrate that dynamic splitting optimizes the FG traffic at the expense of the BG performance, in particular BG jobs in highly loaded queues in unbalanced systems experience very high mean sojourn times. Note that the static assignment strategy does not lead to a stable system for moderate FG load.

D. Heavy FG traffic load

When exposing the $N = 2$ queueing network to a high FG traffic load of $\rho_0 = 1.8$ the four strategies shown in Table X may lead to a stable system. Dynamic splitting sends the most traffic over the queue that also carries BG traffic. Bayesian

TABLE IX: Background traffic performance (γ) for various CA strategies with moderate foreground traffic ($\rho_0 = 0.9$) as a function of the background load.

CA Method	ρ_1 \ ρ_2	0.2	0.4	0.6	0.8
JSQ	0.1	1.91	2.61	4.08	8.69
dynamic assignment	0.1	2.03	2.92	4.84	9.34
static split	0.1	2.56	3.78	6.71	16.65
dynamic split	0.1	2.24	3.27	5.91	18.06
JSQ	0.3		3.39	6.35	
dynamic assignment	0.3		3.60	7.31	
static split	0.3		5.08	10.76	
dynamic split	0.3		4.13	8.66	

TABLE X: Simulation results for heavy FG load ($\rho_0 = 1.8$), with $\rho_1 = 0$ and $\rho_2 = 0.1$.

	α^*	$\mathbb{E}[S_0^{\alpha^*}]$	γ
JSQ	0.527	10.532	10.682
Dynamic assignment	0.533	10.370	15.391
Static split	0.527	11.723	20.555
Dynamic split	0.525	9.510	19.207

assignment performs slightly better with respect to FG traffic performance than JSQ. In this context the relative difference between static splitting and the other methods is fairly limited when considering its simplicity. The FG traffic performance of dynamic splitting is 8.3% better than the second-best strategy of Bayesian assignment. When taking an overall performance metric $\psi = \rho_0 \cdot \mathbb{E}[S_0] + \rho_2 \cdot \gamma$ dynamic splitting also performs best, improving the performance with 4.9% in comparison to JSQ.

V. Summary

In this paper the peformance impact of splitting or assigning foreground (FG) traffic jobs over multiple PS-queues is considered. To this end, we compare the following five splitting and assignment policies: JSQ, static splitting, static assignment, dynamic splitting and dynamic assignment. The comparison is based on the following quality metrics: the mean sojourn time of both FG and background (BG) traffic streams, and the splitting ratios. Based on extensive simulation experimentation, we come to the following observations.

- Dynamic splitting takes advantage of the presence of multiple networks by optimizing the FG traffic streams very efficiently, leading to the fastest FG transfer times of all strategies. Due to its dynamic behavior it takes advantage of any possible capacity a network may deliver, without risking high transfer times due to sudden network load fluctuations. The background traffic may suffer from higher delays, especially in the highly loaded network of an unbalanced system. However, the background traffic does not suffer from degraded performance under all circumstances. Dynamic splitting may take advantage of short-term underutilizations in networks in a much more adequate way than with dynamic assignment.

- Dynamic assignment is the best solution if one wishes to distribute traffic over multiple networks, but does not want to make the total capacity of the FG traffic exceed the capacity of a single network (which is also an assumption of multi-path TCP).
- Dynamic assignment improves the FG performance, and at the same time is mild to the BG traffic. By dynamic assignment one may decide in exceptional cases to select another queue.
- Static splitting may improve the FG performance, but not under all circumstances. As the loads in the network may differ and vary different from the mean values, static splitting takes the risk of re-sequencing delays and losses may have serious negative impact (the highest of all methods compared) on the BG performance.

VI. Acknowledgments

This work was performed within the project RRR (**R**ealisation of **R**eliable and Secure **R**esidential Sensor Platforms) of the Dutch program *IOP Generieke Communicatie*, number IGC1020, supported by the *Subsidieregeling Sterktes in Innovatie*.

References

[1] J. W. Bosman, G. J. Hoekstra, S. Bhulai, G. J. Hoekstra, and R. D. van der Mei, "A Simple Index Rule For Efficient Traffic Splitting Over Parallel Wireless Networks With Partial Information," *Performance Evaluation*, vol. 70, pp. 889 – 899, 2013.

[2] "Multipath tcp (mptcp) charter," February 2014, http://datatracker.ietf.org/wg/mptcp/charter/.

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

[4] R. Stewart, "Stream control transmission protocol," Internet Engineering Task Force, RFC 2960, October 2000.

[5] I. S. 802.11n, "Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer specifications Enhancements for Higher Throughput," October 2009.

[6] I. S. 802.11ac, "Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications–Amendment 4: Enhancements for Very High Throughput for Operation in Bands below 6 GHz." 2013.

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

[8] F. Berggren and R. Litjens, "Performance analysis of access selection and transmit diversity in multi-access networks," in *MobiCom '06: Proceedings of the* 12^{th} *Annual International Conference on Mobile Computing and Networking*, New York, NY, U.S.A., 2006, pp. 251–261.

[9] 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* 14^{th} *IST Mobile and Wireless Communications Summit*, 2005.

[10] H. Hsieh and R. Sivakumar, "A transport layer approach for achieving aggregate bandwidths on multi-homed mobile hosts," *Wireless Networks*, vol. 11, no. 1, pp. 99–114, 2005. [Online]. Available: http://dx.doi.org/10.1007/s11276-004-4749-6

[11] C. Huang and C. Tsai, "The handover control mechanism for multipath transmission using stream control transmission protocol (SCTP)," *Computer Communications*, vol. 30, no. 17, pp. 3239–3256, 2007.

[12] J. Iyengar, "End-to-end concurrent multipath transfer using transport layer multihoming," Ph.D. dissertation, University of Delaware, 2006.

[13] J. Iyengar, P. Amer, and R. Stewart, "Concurrent multipath transfer using sctp multihoming over independent end-to-end paths," *IEEE/ACM Transactions on Networking*, vol. 14, no. 5, pp. 951–964, 2006.

[14] M. Zhang, J. Lai, A. Krishnamurthy, L. Peterson, and R. Wang, "A transport layer approach for improving end-to-end performance and robustness using redundant paths," in *ATEC '04: Proceedings of the annual conference on USENIX Annual Technical Conference*, Berkeley, CA, U.S.A., 2004, pp. 99–112.

[15] R. D. van der Mei, G. J. Hoekstra, and J. W. Bosman, "Efficient Traffic Splitting In Parellel TCP-Based Networks: Modeling And Experimental Validation," in *Proceedings of the International Teletraffic Congress (ITC, 2013)*, 2013, pp. –.

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

[17] Y. Wu, C. Williamson, and J. Luo, "On processor sharing and its applications to cellular data network provisioning," *Performance Evaluation*, vol. 64, no. 9-12, pp. 892–908, 2007.

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

[19] G. Hoekstra and R. van der Mei, "Effective load for flow-level performance modelling of file transfers in wireless LANs," *Computer Communications*, vol. 33, no. 16, pp. 1972–1981, 2010.

[20] G. J. Hoekstra, R. D. van der Mei, and N. Diaz Feraren, "Engineering Elastic Traffic In TCP-Based Networks: Processor Sharing And Effective Services Times," in *Proceedings of International Teletraffic Congress 2013 (ITC 25)*, 2013.

[21] S. Ko and R. Serfozo, "Response times in M/M/s fork-join networks," *Advances in Applied Probability*, vol. 36, no. 3, pp. 854–871, 2004.

[22] G. Choudhury and D. Houck, "Combined queuing and activity network based modeling of sojourn time distributions in distributed telecommunication systems," in *Proceedings of the* 14^{th} *International Teletraffic Congress - ITC14*, Antibes, France, 1994, pp. 525–534.

[23] L. Flatto and S. Hahn, "Two parallel queues created by arrivals with two demands," *SIAM Journal on Applied Mathematics*, vol. 44, no. 5, pp. 1041–1053, 1984. [Online]. Available: http://link.aip.org/link/?SMM/44/1041/1

[24] J. Lui, R. Muntz, and D. Towsley, "Computing performance bounds for fork-join queueing models," The Chinese University of Hong Kong, Tech. Rep., 1994.

[25] M. Lelarge, "Tail asymptotics for discrete event systems," in *Proceedings Valuetools '06:* 1^{st} *international conference on Performance evaluation methodolgies and tools*, 2006, pp. 563–584.

[26] ——, "Packet reordering in networks with heavy-tailed delays," *Mathematical Methods of Operations Research*, vol. 67, no. 2, pp. 341–371, 2008.

[27] E. Altman, U. Ayesta, and B. Prabhu, "Load balancing in processor sharing systems," in *Proceedings of the* 3^{rd} *International Conference on Performance Evaluation Methodologies and Tools*, ser. ValueTools '08, 2008, pp. 12:1–12:10.

[28] F. Baccelli, W. Massey, and D. Towsley, "Acyclic fork-join queuing networks," *Journal of the ACM*, vol. 36, no. 3, pp. 615–642, 1989.

[29] P. Key, L. Massoulié, and D. Towsley, "Combining multipath routing and congestion control for robustness," in *Proceedings Conference on Information Sciences and Systems*, 2006.

[30] ——, "Path selection and multipath congestion control," in *Proceedings IEEE INFOCOM 2007*, 2007, pp. 143–151.

[31] G. Hoekstra, R. van der Mei, Y. Nazarathy, and A. Zwart, "Optimal file splitting for wireless networks with concurrent access," *Lecture Notes in Computer Science*, vol. 5894, pp. 189–203, 2009.

[32] G. J. Hoekstra, R. D. van der Mei, and S. Bhulai, "Optimal Job Splitting In Parallel Processor Sharing Queues," *Stochastic Models*, vol. 28, 2012.

[33] S. Bhulai, G. J. Hoekstra, J. W. Bosman, and R. D. van der Mei, "Dynamic Traffic Splitting To Parallel Wireless Networks With Partial Information: A Bayesian Approach." *Performance Evaluation*, vol. 69, pp. 41 – 52, 2012.

[34] R. D. van der Mei, G. J. Hoekstra, and J. W. Bosman, "Efficient Traffic Splitting In Parellel TCP-Based Networks: Modeling And Experimental

Validation," in *Proceedings of the International Teletraffic Congress (ITC, 2013)*, 2013.