Load Balancing of Elastic Data Traffic in Heterogeneous Wireless Networks

Abdulfetah Khalid
School of Electrical Engineering
Aalto University, Finland
Email: abdulfetah.khalid@aalto.fi

Pasi Lassila
School of Electrical Engineering
Aalto University, Finland
Email: pasi.lassila@aalto.fi

Samuli Aalto
School of Electrical Engineering
Aalto University, Finland
Email: samuli.aalto@aalto.fi

***Abstract*—Heterogeneous wireless networks have been introduced to enable cellular networks to handle the continuously increasing amount of mobile data traffic. In heterogeneous networks, the deployment of macrocells is accompanied by the use of low power pico and femtocells (referred to as microcells) in hotspot areas inside the macrocell, which increases the data rate per unit area. In this paper, we study the load balancing problem of elastic data traffic in heterogeneous wireless networks focusing on a single macrocell that is supported by a number of microcells. This results in a parallel queueing system where individual cells are modelled as single server processor sharing queues. Both static and dynamic load balancing schemes are developed to balance the data flows between the macrocell and the microcells so that the mean flow-level delay is minimized. The performance of static policies is evaluated by analytical and numerical methods, whereas for the dynamic policies, we need to resort to simulations. Our numerical experiments indicate that all dynamic policies can significantly improve the flow-level delay performance compared to the optimal static policy. Among the implemented policies, the so-called myopic policy appears to be systematically the best. However, also a much simpler policy with respect to the required state information, the modified join-the-shortest-queue policy, is able to achieve almost the same performance. In addition, we observe that the performance gain of most of the dynamic polices (including the myopic one) is at least approximately insensitive with respect to the flow size distribution.**

I. Introduction

Mobile data communications has undergone significant evolution in recent years. The introduction of smart phones has resulted in a drastic increase in the data demand from the mobile users. To address this demand, operators are trying to increase the capacity of current mobile networks by enhancing the radio links and introducing so-called *heterogeneous networks* [1]. These are network architectures with microcells (i.e., pico and femtocells) overlaying the macrocell network. The macrocells are similar to the conventional base stations that we use in today's networks providing the basic coverage to the whole cell area, while the microcells are low power base stations used for hotspot areas within a macrocell to improve spectral efficiency per unit area or for areas that the macrocell cannot cover efficiently.

In a cellular network, efficient allocation of resources (channels) to cells is needed due to limited bandwidth. The problem becomes even worse when some cells are congested while others are not, which results in a hotspot problem in which the quality of service in congested cells is considerably degraded. In order to increase the overall system utilization, the load of the overloaded cells should be distributed to less congested cells. In heterogeneous networks, when macrocells are overloaded, part of arriving traffic can be transferred to less loaded overlapping microcells, or vice versa, by a suitable load balancing algorithm.

This paper addresses the load balancing problem for heterogeneous wireless networks. We focus on a single macrocell that is supported by different types of microcells having a wired backhaul connection to the Internet. These microcells operate on a different frequency than the macrocell and have their own characteristics with respect to handling data traffic. From the traffic point of view, each cell can be considered, whether it is the macrocell or a microcell, as a server which has its own service rate. We assume that the traffic consists of elastic downlink data flows. By further assuming that the resources of a cell are time-shared uniformly between the active users and the scheduler is not able to utilize the instantaneous rate variations across the users (non-channel-aware scheduling), individual cells can be modelled using the classical single server processor sharing (PS) queue [2]. Since there are different types of cells, we have a system of heterogeneous parallel processor sharing queues. Our problem is related to the well-known dispatching problem but it is more complex, as discussed later.

Our target is to develop both static and dynamic load balancing schemes that balance the elastic data flows between the macrocell and microcells. In the static case, the load balancing problem is formulated as an optimization problem in which the mean flow delay is minimized. The performance of static policies is evaluated by analytical and numerical methods. For the dynamic case, we consider several policies and investigate the delay performance by simulations. Our extensive numerical experiments indicate that all dynamic policies can significantly improve the flow-level delay performance compared to the optimal static policy. Among the implemented dynamic policies, the classical Join the Shortest Queue (JSQ) policy typically performs worst. On the other hand, the so-called myopic policy, which relies on detailed knowledge of remaining service times and is optimal when there are no more future arrivals, appears to be systematically the best. However, a much simpler heuristic policy, the modified JSQ (MJSQ), is able to achieve almost the same performance but with knowledge of number of flows and average service rates only. Experiments with optimized policies based on the first policy iteration of the MDP theory are not able to give any essential improvements over MJSQ either. These lead us to

believe that the myopic policy can be very close to optimal in minimizing the mean flow delay. Finally, we observe that the performance gain of most of the dynamic polices (including the myopic one) is at least approximately insensitive with respect to the flow size distribution.

Load balancing in parallel server systems has been widely investigated in the context of distributed server systems consisting of a single dispatcher and a group of parallel servers [3]. In this classical dispatching problem, the target is to determine an optimal dispatching policy that will, e.g., minimize the mean response time. For static policies (see, e.g., [4], [5]) the decisions do not depend on the states of the queues, while dynamic policies assign arriving jobs depending on some kind of information on the state of the system (see, e.g., [6], [7]). JSQ and Least Work Left (LWL) are well-known dynamic policies where an arriving job is dispatched to the least loaded server (measured in the number of jobs for JSQ and in the amount of unfinished work for LWL). Winston [6] showed the optimality of JSQ for homogeneous servers with the FIFO service discipline when the job sizes are exponential (but unknown for the dispatcher), whereas LWL is the optimal dynamic dispatching policy when the job sizes are deterministic [8]. A simple static rule is based on a probabilistic approach, while the Size Interval Task Assignment (SITA) policy represents a more advanced static method where all jobs within a given size range are dispatched to a particular server [9], [10]. Feng et al. [11] have shown that SITA with suitably chosen size ranges is the optimal static dispatching policy for homogeneous servers with the FIFO service discipline. While the optimal static dispatching policy for parallel FIFO queues is size-based (SITA for homogeneous servers and more complicated for heterogeneous servers), the optimal static dispatching policy for parallel PS queues does not need to depend on the size of the arriving job but can be purely probabilistic (both for homogeneous and heterogeneous servers), see [12].

In the classical dispatching problem described above, there is a single decision point, i.e., the dispatcher. However, in our heterogeneous network scenario, there are n parallel decision points corresponding to n separate microcells, which makes the problem essentially more complex. The load balancing problem in this setting has not been studied much. In [13], a load balancing algorithm is introduced that improves the call blocking probability in the heterogeneous wireless networks.

The rest of this paper is organized as follows. In Section II, we introduce the system model and formulate the problem. The optimal probabilistic dispatching policy is analyzed in Section III. Dynamic policies are introduced in Section IV, and their performance benefits are investigated numerically in Section V. Conclusions are in Section VI.

II. System Model and Problem Formulation

We consider a heterogeneous wireless system consisting of a single macrocell and n separate microcells located inside the coverage area of the macrocell, where each cell has its own dedicated wireline connection to the Internet. Traffic consists of elastic downlink data flows (such as TCP file transfers). Let λ_i denote the arrival rate of new flows within the area of microcell $i = 1, \ldots, n$. Each such a flow can be carried either

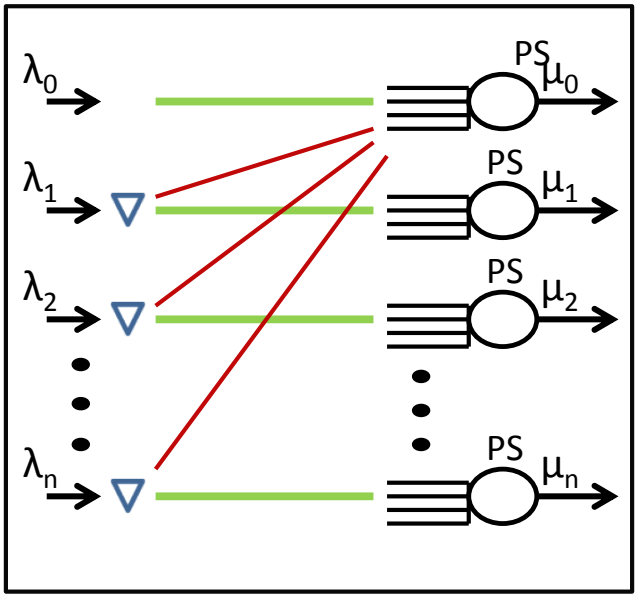

Fig. 1. Load balancing model for heterogeneous networks

by the microcell itself or the macrocell (but not by the other microcells). So upon the arrival, a dispatching decision must be made whether the flow is carried by the microcell or the macrocell. In addition, let λ_0 denote the arrival rate of those flows (outside the "hotspot" areas covered by the microcells) that can only be carried by the macrocell. All these separate flow arrival processes are assumed to be independent Poisson processes.

We assume that the macrocell and the microcells are operating in a different frequency band, i.e., out-band, which is reasonable in order to minimize interference between macro- and the microcells [1]. On the other hand, the microcells can be assumed to be far enough from each other so that they do not interfere each other. Hence, as also motivated in Section I, each cell is modelled by a single server PS queue. Let μ_i denote the service rate of microcell i. Thus, $1/\mu_i$, the mean service time, is the average time that is needed to complete the transfer of a random flow by microcell i if there were no other flows to be carried by the same cell. We note that the service time is affected at least by the size of the original flow, the location of the corresponding terminal (within the microcell), and the radio channel conditions during the flow transfer. However, since we assume that the scheduler of the microcell does not utilize these features, we do not model them separately. Finally, let μ_0 denote the service rate of the macrocell. See Figure 1 for an illustration of our model.

In this paper, we assume that, for all i,

$$\mu_i \geq \mu_0. \tag{1}$$

This is motivated by the fact that in the coverage area of a femtocell the achievable rate is typically higher than in the whole macrocell.

In an *unbalanced system*, each arriving flow within a specific microcell is served by that cell independent of whether the cell is congested or not. However, in a *balanced system*

flows arriving in any microcell i may be served either by the local microcell i or the macrocell, which can be utilized to optimize the system performance. We investigate this load balancing problem in two stages. First, we seek to find the optimal probabilistic dispatching policy that minimizes the mean sojourn time, i.e., the average total time needed to carry the flow. Here we determine the optimal splitting probabilities when the load parameters (arrival rates and mean service times) are known. Then, in the second stage, we consider dynamic load balancing polices.

However, before handling the optimization problem, we consider the stability of such a parallel queueing system. Since the macrocell must be able to carry both its own dedicated arrivals and the excess traffic from the microcells, we have the following *necessary* stability condition:

$$\lambda_0 + \sum_{i=1}^{n} (\lambda_i - \mu_i)^+ < \mu_0, \tag{2}$$

where $(x)^+ = \max\{x, 0\}$.

III. Optimal probabilistic dispatching policy

In this section, we consider static probabilistic dispatching policies. Such a policy is described by a vector $\mathbf{p} = (p_1, p_2, ..., p_n)$ where p_i refers to the probability that a flow belonging to arrival stream i is assigned to the corresponding microcell i. All dispatching decisions are assumed to be independent of each other. Figure 2 illustrates this kind of probabilistic traffic allocation.

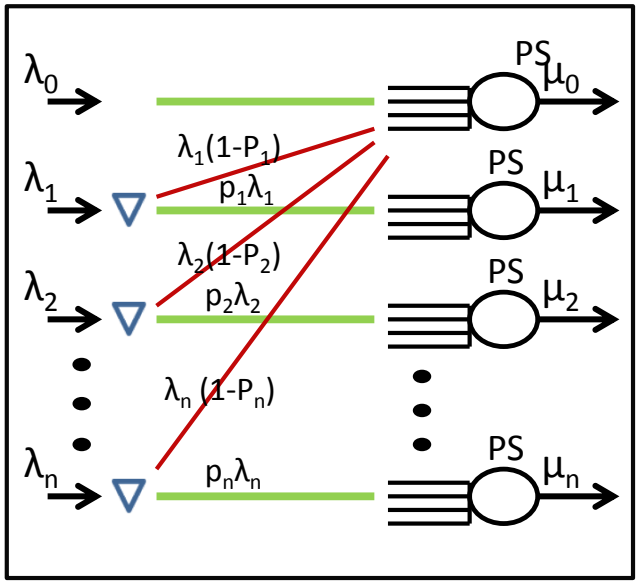

Fig. 2. Probabilistic traffic allocation in heterogeneous networks

Due to the splitting property of the Poisson process, the arrival process to each of the queues is an independent Poisson process for any probabilistic dispatching policy. Thus, the system is composed of $n+1$ parallel M/G/1-PS queues implying that the necessary and sufficient stability condition is clearly as follows:

$$\lambda_0 + \sum_{i=1}^{n} \lambda_i (1 - p_i) < \mu_0 \quad \text{and} \quad \lambda_i p_i < \mu_i \text{ for all } i. \tag{3}$$

In addition, for a stable system, the mean sojourn time is

$$E[T] = \frac{\lambda_0 + \sum_{i=1}^{n} \lambda_i (1 - p_i)}{\left(\lambda_0 + \sum_{i=1}^{n} \lambda_i\right)\left(\mu_0 - \left(\lambda_0 + \sum_{i=1}^{n} \lambda_i (1 - p_i)\right)\right)} + \sum_{i=1}^{n} \frac{p_i \lambda_i}{\left(\lambda_0 + \sum_{i=1}^{n} \lambda_i\right)(\mu_i - p_i \lambda_i)}. \tag{4}$$

Our objective is to find the optimal probabilistic dispatching policy that minimizes the mean sojourn time $E[T]$. The optimization problem can be formulated in a standard form as follows:

$$\begin{aligned} \text{minimize} \quad & E[T] \\ \text{subject to} \quad & \lambda_i p_i - \mu_i \le 0, \qquad \text{for all } i \in \{1, ..., n\} \\ & \left(\lambda_0 + \sum_{i=1}^{n} \lambda_i (1 - p_i)\right) - \mu_0 \le 0, \\ & p_i - 1 \le 0, \qquad \text{for all } i \in \{1, ..., n\} \\ & p_i \ge 0, \qquad \text{for all } i \in \{1, ..., n\}. \end{aligned} \tag{5}$$

Since the objective function and the constraints are convex, the optimization problem can be solved using standard numerical techniques. As a result, we get the optimal policy defined by the probability vector $\mathbf{p}^*$ satisfying

$$\mathbf{p}^* = \underset{\mathbf{p}}{\operatorname{argmin}}\ E[T]. \tag{6}$$

Symmetric case

As explained above, we have to rely on numerical methods when determining the optimal probabilistic dispatching policy in the general case. However, when considering a *symmetric traffic scenario* defined by requiring that $\lambda_i = \lambda$ and $\mu_i = \mu$ for all $i = 1, \ldots, n$, we are able to derive the optimal policy and the resulting performance analytically. Due to the symmetry, we may restrict the study to *symmetric policies* for which $p_i = p$ for all i. The stability condition (3) for the symmetric policies simplifies to

$$\lambda_0 + n\lambda (1 - p) < \mu_0 \quad \text{and} \quad \lambda p < \mu \tag{7}$$

and the expression (5) for the mean sojourn time to

$$E[T] = \frac{\lambda_0 + n\lambda (1 - p)}{(\lambda_0 + n\lambda)(\mu_0 - (\lambda_0 + n\lambda (1 - p)))} + \frac{np\lambda}{(\lambda_0 + n\lambda)(\mu - p\lambda)}. \tag{8}$$

The optimization problem is to find $p^* \in [0, 1]$ that minimizes the mean delay, $E[T]$. This renders a unique solution due to the convexity of $E[T]$:

$$p^* = \min\{p, 1\}, \tag{9}$$

where

$$p = \frac{\sqrt{\mu_0}\mu + \sqrt{\mu}\left((\lambda_0 + n\lambda) - \mu_0\right)}{\sqrt{\mu_0}\lambda + \sqrt{\mu} n \lambda}. \tag{10}$$

Note that $p > 0$ since, due to assumption (1), we have

$$p > \frac{\sqrt{\mu_0}\mu - \sqrt{\mu}\mu_0}{\sqrt{\mu_0}\lambda + \sqrt{\mu}n\lambda} = \frac{\sqrt{\mu_0}\sqrt{\mu}\left(\sqrt{\mu} - \sqrt{\mu_0}\right)}{\sqrt{\mu_0}\lambda + \sqrt{\mu}n\lambda} \geq 0. \quad (11)$$

IV. Dynamic policies

In this section, we consider *dynamic policies* for which the dispatching decisions depend on the states of the queues. The description of the state of the system, $\mathbf{s}$, depends on the chosen policy. It may, for example, be the number of flows in the system or the remaining workload of the flows. When a flow arrives to a specific microcell i, the dispatcher assigns it to the microcell i or to the macrocell (indexed by 0 from this on) based on the chosen policy. Let $\Delta_i(\mathbf{s}) \in \{0, i\}$ denote the action that the dispatcher in microcell i takes when the system is in state $\mathbf{s}$.

A. Join the shortest queue

In the *Join the Shortest Queue* policy (JSQ), the dispatching decision is determined by comparing the number of flows in the two cells. So the system state from the microcell i point of view is $\mathbf{s} = (n_0, n_i)$, where n_i [n_0] denotes the number of flows in cell i [0]. The JSQ policy is defined by

$$\Delta_i(\mathbf{s}) = \operatorname{argmin}\{n_0, n_i\}. \quad (12)$$

When the number of flows in both the macrocell and the microcell are equal, the policy is implemented to dispatch the arriving flow to the microcell i. We recall that, for the original dispatching problem (with a single decision point), JSQ is the optimal policy for exponentially distributed job sizes and homogeneous servers [6].

B. Modified join the shortest queue

The *Modified JSQ* policy (MJSQ) is a modified version of the basic JSQ policy where the number of flows in the cell is scaled with the service rate of the cell. So the system state remains the same $\mathbf{s} = (n_0, n_i)$, and MJSQ is defined by

$$\Delta_i(\mathbf{s}) = \operatorname{argmin}\{n_0/\mu_0, n_i/\mu_i\}. \quad (13)$$

MJSQ is close to the minimum expected delay (MED) dispatching policy, introduced in [14]. However, MJSQ does not take into account the size of the arriving flow. Although not optimal, the MED policy can provide good performance in heterogeneous multi-server queueing systems.

C. Myopic policy

The *Myopic* policy (MP) assigns the arriving flow based on the additional delay cost introduced by the new flow. The policy assumes that the remaining service requirements of the flows and the service requirement of the arriving flow are known. The policy stems from the idea to minimize the total delay assuming that no further flows arrive after the flow that has just arrived.

Consider first microcell i. As before, let n_i denote the number of flows, and define $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,n_i})$, where x_{ij} denotes the remaining service time of flow j and the flows are indexed in the decreasing order of the remaining service times so that $x_{i,1} \geq \ldots \geq x_{i,n_i}$. Bonomi [7] was the first to determine an explicit formula for the additional delay cost:

$$\sigma_i = x(2k_i + 1) + 2 \sum_{j=k_i+1}^{n_i} x_{ij}, \quad (14)$$

where x denotes the service time of the arriving flow and k_i refers to the number of flows "longer" than x, $k_i = \max\{j : x_{ij} > x\}$.

The additional cost σ_0 related to the macrocell is calculated correspondingly. The state of the system from the microcell i point of view is $\mathbf{s} = ((n_0, \mathbf{x}_0), (n_i, \mathbf{x}_i))$, and the MP policy is defined by

$$\Delta_i(\mathbf{s}) = \operatorname{argmin}\{\sigma_0, \sigma_i\}. \quad (15)$$

D. Improved Policies

Now we will use the *first policy iteration* (FPI) technique of the MDP theory [15] to improve the optimal probabilistic policy discussed earlier. The key observation is that the behavior of the system with the optimal probabilistic policy can still be characterized by parallel independent M/G/1 PS queues. Thus, in the first policy iteration we can utilize the known relative values of states (with respect to the sojourn time) for a single M/M/1-PS queue [16], [17].

We consider below two FPI based policies. The first one is the *Size-unaware FPI* policy (FPI-U) where the dispatcher is only aware of the number of flows in both cells, while in the second one, which is called the *Size-aware FPI* policy (FPI-A), the dispatcher is even aware of the remaining service times of all flows in both cells. For both of these policies, we use the optimal probabilistic dispatching policy as the basic policy that is needed to determine the relative values of states. Similarly as the MP policy, the FPI policies assign the arriving flow based on the additional delay cost introduced by the new flow. The difference comes from the way the additional cost is evaluated, as explained below.

1) Size-unaware FPI policy: Consider first microcell i, and recall that n_i refers to the number of flows in cell i. The additional delay cost, σ_i, of one additional flow is determined as the following difference of the relative values of states [16], [17]:

$$\sigma_i = v_i(n_i + 1) - v_i(n_i) = \frac{n_i + 1}{\mu_i - p_i^*\lambda_i}, \quad (16)$$

Correspondingly, for the macrocell, we get the additional delay cost

$$\sigma_0 = \frac{n_0 + 1}{\mu_0 - \lambda_0 - \sum_{j=1}^{n} \lambda_j(1 - p_j^*)}, \quad (17)$$

where n_0 refers to the number of flows in the macrocell.

The state of the system from the microcell i point of view is $\mathbf{s} = (n_0, n_i)$, and the FPI-U policy is defined by

$$\Delta_i(\mathbf{s}) = \operatorname{argmin}\{\sigma_0, \sigma_i\}. \quad (18)$$

2) Size-aware FPI policy (FPI-A): Consider again first microcell i. As before, let n_i denote the number of flows and define $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,n_i})$, where x_{ij} denotes the remaining service time of flow j and the flows are indexed in the decreasing order of the remaining service times so that

$x_{i,1} \geq \ldots \geq x_{i,n_i}$. The additional delay cost, σ_i, of one additional flow with service time x is now determined as the following difference of the relative values of states [17]:

$$\sigma_i = v_i(n_i + 1, \mathbf{x}_i \oplus x) - v_i(n_i, \mathbf{x}_i), \tag{19}$$

where $\mathbf{x}_i \oplus x = (x_{i,1}, \ldots, x_{i,k_i}, x, x_{i,k_i+1}, \ldots, x_{i,n_i})$ with $k_i = \max\{j : x_{ij} > x\}$ and

$$v_i(n_i, \mathbf{x}_i) = v_i(n_i) + \frac{1}{(1-\rho_i^*)^2} \sum_{j=1}^{n_i} (2j-1)\, x_{ij} + \frac{2-\rho_i^*}{\mu_i (1-\rho_i^*)^2} \sum_{j=1}^{n_i} \left(-\frac{j\rho_i^*}{1-\rho_i^*} \right) \left(\sum_{k=1}^{j} e^{-\mu_i(1-\rho_i^*)(x_{ik}-x_{ij})} \right) \times \left(1 - e^{-\mu_i(1-\rho_i^*)(x_{ij}-x_{i,j+1})} \right). \tag{20}$$

with $\rho_i^* = p_i^* \lambda_i / \mu_i$.

The additional cost σ_0 related to the macrocell is calculated correspondingly. The state of the system from the microcell i point of view is $\mathbf{s} = ((n_0, \mathbf{x}_0), (n_i, \mathbf{x}_i)$, and the FPI-A policy is defined by

$$\Delta_i(\mathbf{s}) = \operatorname{argmin} \{\sigma_0, \sigma_i\}. \tag{21}$$

V. Numerical results

In this section, we examine the performance of the derived load balancing policies. We present numerical examples for both symmetric and asymmetric traffic scenarios and illustrate the performance gain of the implemented dynamic policies compared to the optimal static one. Our basic system consists of two microcells, and initially we assume that the flow sizes follow an exponential distribution. Later on, we also study the effect of more variable service time distributions, a greater number of microcells, and faster service rates of the microcells.

The performance of the dynamic policies is studied by simulations. The system is typically simulated as a function of the load for each policy, and the mean number of flows in the system is considered as the performance metric. To estimate the mean number of customers, each simulation run consisted of $5 \cdot 10^6$ arrivals for the FPI-A policy and $5 \cdot 10^7$ arrivals for all other policies. The reason for the lower number of arrivals for the FPI-A policy is the higher computational complexity of the policy, especially at higher loads, see Section IV.D.2.

A. Basic traffic scenarios

In this section, we examine four different traffic scenarios. In all these scenarios the service rates of the cells remain the same, $\mu_0 = 1$ and $\mu_1 = \mu_2 = 2$, so that the difference comes from the arrivals rates.

First, we will go through results obtained from two symmetric traffic scenarios where the arrival rate to the two microcells are equal. In Traffic scenario 1, there are no arrivals in the macrocell, $\lambda_0 = 0$, and the arrival rates to the microcells are equal, $\lambda_1 = \lambda_2 = \lambda$, where λ is varied. In Traffic scenario 2, the arrival rates to the microcells are constant, $\lambda_1 = \lambda_2 = 2$, while the arrival rate to the macrocell, λ_0, is varied.

TABLE I. Summary of traffic scenarios

Traffic scenarios	fixed parameters	varied parameters
Traffic scenario 1	$\lambda_0 = 0$	$\lambda_1 = \lambda_2 = \lambda$
Traffic scenario 2	$\lambda_1 = \lambda_2 = 2$	$\lambda_0 = \lambda$
Traffic scenario 3	$\lambda_0 = 0, \lambda_2 = 2$	$\lambda_1 = \lambda$
Traffic scenario 4	$\lambda_1 = 1, \lambda_2 = 2$	$\lambda_0 = \lambda$

Additionally, we study two asymmetric traffic scenarios. In Traffic scenario 3, we assume that $\lambda_0 = 0$, $\lambda_1 = \lambda$, and $\lambda_2 = 2$. So λ_1 is the free parameter to be changed in this case. Traffic scenario 4 is quite similar to Traffic scenario 2 with the only exception that the arrival rates to the microcells are asymmetric, $\lambda_0 = \lambda$, $\lambda_1 = 1$, and $\lambda_2 = 2$. So in this case λ_0 is the free parameter to be varied. Table I summarizes the four traffic scenarios.

Figure 3 illustrates the performance of the dynamic policies in Traffic scenarios 1-4. More precisely, we present the ratio between the total average number of flows of the dynamic policies and that of the optimal static policy as a function of the free traffic parameter for each traffic scenario separately. Note that, due to Little's result, the ratio is exactly the same as the ratio between the average sojourn times of the dynamic policies and that of the optimal static policy.

As expected, all implemented dynamic policies seem to improve the performance compared to the optimal static policy in all examined traffic scenarios. The results also show that MP is systematically the best among them. On the other hand, it can be observed that JSQ performs uniformly worse than the other dynamic policies. It can also be observed that although FPI based policies show a significant improvement over the optimal static policy, they cannot compete with the MP policy. Among the two FPI based policies, FPI-A appears to be slightly better than FPI-U. The MJSQ policy, which requires significantly less information than FPI-A, seems, however, to be better than the two FPI policies in these basic traffic scenarios.

As mentioned above, the MP policy gives uniformly the best results. In fact, it continues to be the best performing policy in all the other tests that we have conducted, as will be discussed in Sections V.B-V.D. On the other hand, MJSQ and the FPI-based improved policies are not performing much worse than MP. Thus, it is plausible that significant improvements in the performance over the MP policy can not be achieved and, indeed, the MP policy is close to optimal in minimizing the mean delay.

It can also be observed that the percentage decrease in the mean number of flows (the gain of the dynamic load balancing policies) is usually quite clear, such as 20% – 40%, but there is one exception. In Traffic scenario 1, the gain almost disappears with small values of λ. This is due to the fact that the optimal splitting probability p^* of the (symmetric) basic policy is equal to 1 for small values of λ indicating that there is not much help from the macrocell in this case.

B. Effect of the flow size variation

The purpose of this section is to investigate how the performance of the *dynamic* load balancing policies changes when we increase the variation of the flow size distribution.

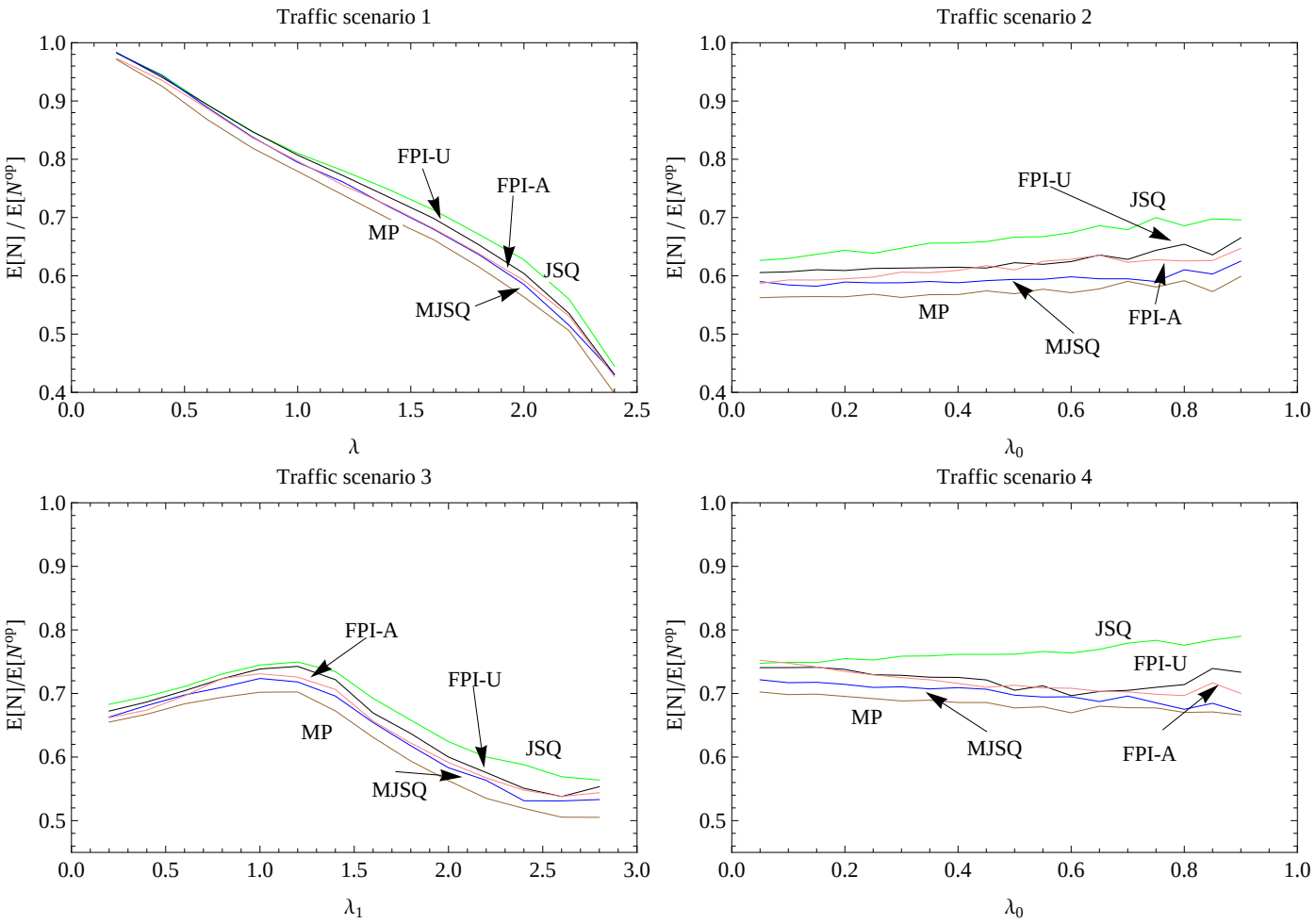

Fig. 3. Ratio of the mean number of flows in the system between the dynamic policies and the base line optimal static policy for Traffic scenarios 1-4

We note that the mean delay performance of all *static* policies is insensitive to the flow size distribution (as long as the mean value remains the same), since all parallel queues are of the M/G/1-PS type in this case. Besides the dynamic policies discussed so far, in this section we consider the *Least Work Left* policy (LWL) which chooses the cell with the least amount of remaining work (in time units) when dispatching an arriving flow.

Traffic scenario 2 is simulated for the bounded Pareto flow size distribution with different values for the shape parameter α (however, keeping the mean value fixed). The CDF of the bounded Pareto distribution is defined by

$$F(x) = \begin{cases} 0, & x < p, \\ \dfrac{1-(k/x)^{\alpha}}{1-(k/p)^{\alpha}}, & k \le x \le p, \\ 1, & x > p. \end{cases}$$

Parameters k and p define the support of the distribution, while α describes its variation. The lower the α, the more variable the flow size distribution will be. The results of the simulations are presented in Figure 4.

It can be seen from the figure that MP is still systematically the best dynamic policy, and MJSQ is the second best in almost all simulated cases. Another observation is that the performance of almost all dynamic policies, except LWL, is at least approximately insensitive with respect to the flow size distribution. When the flow size variation is high, there is a considerable performance degradation in the LWL policy. Although LWL requires more information, it performs even worse than JSQ for the shape parameter $\alpha = 1.5$ and $\alpha = 2.0$. However, when $\alpha = 3.0$, the performance of the LWL policy resembles that of LWL with exponential flow size distribution.

Two major reasons were mentioned in [18] for the performance degradation of LWL in the classical dispatching problem with highly varying distributions. The first one is related to the scheduling policy used, i.e, the PS discipline. For PS systems, the unfinished work does not capture the additional delay cost due to the arriving flow. The other one is related to the service rate of the servers. The LWL policy does not take into account the differences in these service rates as regards the arriving flow.

C. Effect of the number microcells

The effect of increasing the number of microcells, n, within a single macrocell is studied in this section. We consider a generalization of Traffic scenario 2 where $\lambda_0 = \lambda$, $\lambda_1 = \ldots = \lambda_n = 2$, $\mu_0 = 1$, and $\mu_1 = \ldots = \mu_n = 2$.

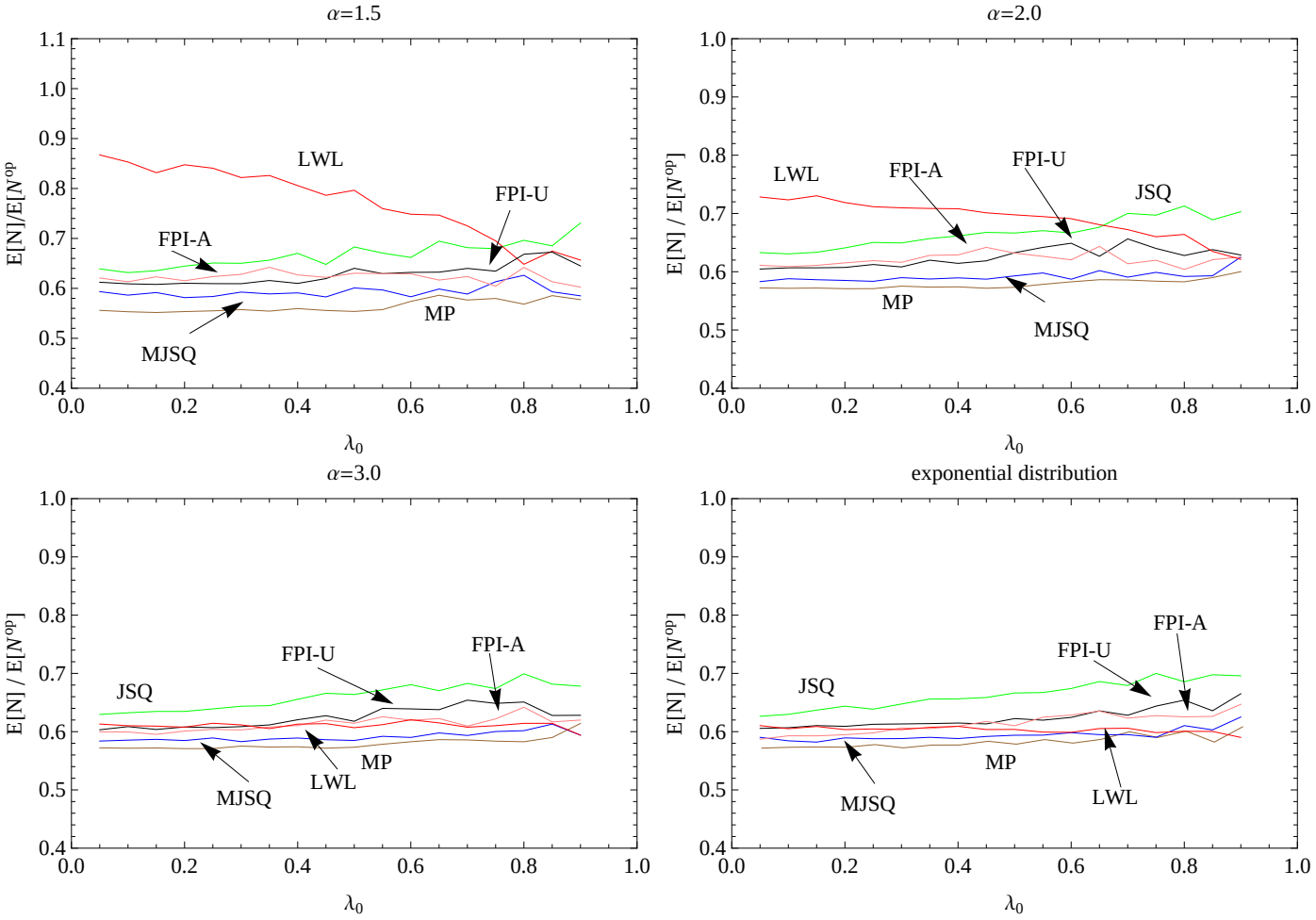

Fig. 4. Illustration of effect of the flow size variation: (a) bounded Pareto flow size distribution: $\alpha = 1.5$ (top left), (b) bounded Pareto flow size distribution: $\alpha = 2.0$. (top right), (c) bounded Pareto flow size distribution: $\alpha = 3.0$ (bottom left) and (d) exponential flow size distribution (bottom right)

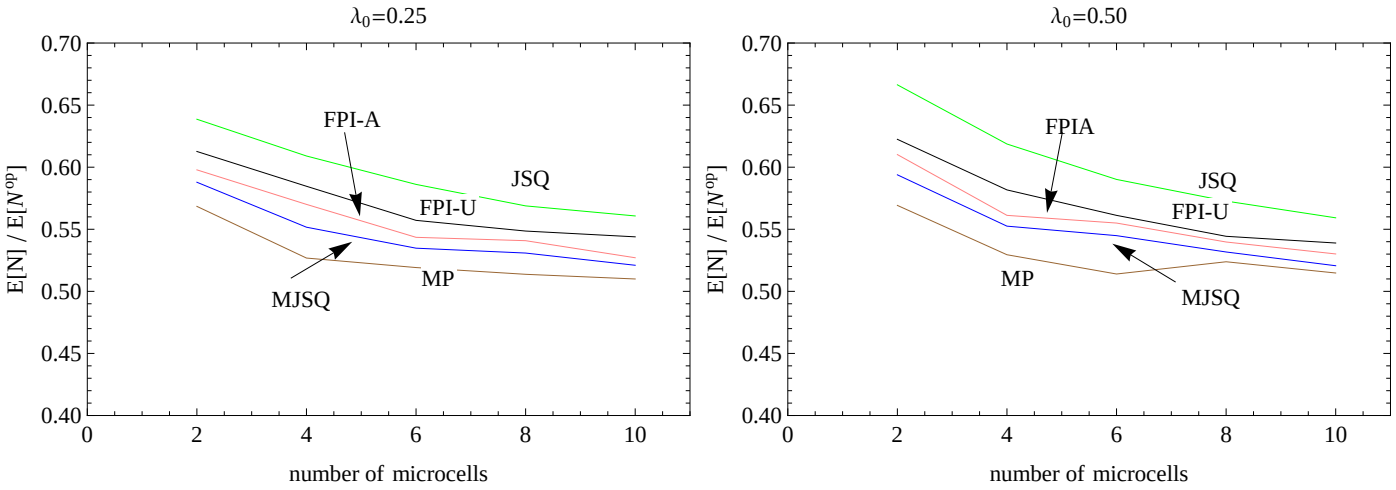

Fig. 5. Impact of the number of microcells on the performance gain of load balancing policies

Figure 5 shows the impact of the number of microcells on the performance gain of the dynamic load balancing policies. The y-axis represents the ratio in the mean number of flows in the system, as compared to the base line optimal static policy, and the x-axis shows the number of microcells in the system for two different values of the free parameter λ_0.

First we observe that, as before, MP is again systematically the best dynamic policy, and MJSQ is the second best. It can also be seen from the figure that there is around 40% performance gain, compared to the optimal static policiy, when we have two microcells. However, this percentage increases systematically when we increase the number of microcells, reaching around 45% when the number of microcells is 10. Thus, we conclude that increasing the number of microcells results in an increase in the performance gain of the system.

D. Impact of service rate difference

Finally, we study the impact of increasing the service rate difference between the macrocell and the microcells on the performance gain of the dynamic policies. We consider the same four different traffic scenarios as earlier (macrocell + two microcells), but the service rate of the microcells is doubled, i.e., we have $\mu_1 = \mu_2 = 4$. The performance ratio compared with the static optimal policy for each scenario as a function of the load is given in Figure 6.

Qualitatively, the results are quite similar to what we observed for the basic scenario in Section V.A. The JSQ policy performs worst and MP is performing uniformly the best, while the simple MJSQ gives roughly as good performance as MP. The FPI-based improvements are more or less as good as MJSQ. A slight difference occurs in Traffic scenario 1: in Figure 6 under light traffic both JSQ and MJSQ perform even worse than the static optimal. This can be explained by noting that for $\mu_1 = \mu_2 = 4$ the optimal static policy never uses the macrocell, i.e., $p_1^* = p_2^* = 1$, when $\lambda < 2$. However, JSQ and MJSQ are both dynamic policies and sometimes based on the state they dispatch flows to the very slow macrocell.

VI. Conclusions

In this paper, we have investigated the load balancing problem of elastic data traffic in heterogeneous networks. We have modelled a heterogeneous wireless network consisting of a single macrocell and a number of out-band microcells as a distributed server system with parallel heterogeneous M/G/1-PS queues. Each queue represents a cell with its own arrival rate of traffic. In the microcells, a decision can be made to serve the flow locally or whether to dispatch it to the macrocell. This gives rise to a generalized dispatching problem with multiple decision points.

The objective was to determine policies that are able to optimize the mean delay of the flows in the system. The static optimal policy was used as the baseline policy. In the symmetric case, an explicit solution was determined but in the general case the problem can be solved numerically. In our setting, the optimal dynamic policy minimizing the mean delay is not known. However, we presented several dynamic policies that are well motivated by literature and applied them in our case. The implemented policies include: classical JSQ (optimal in the classic dispatching problem with homogeneous servers), MJSQ (heuristic version of JSQ for heterogeneous servers), MP (optimal if no more future arrivals) and two optimized policies (FPI-U and FPI-A) that are based on the first policy iteration method of the MDP theory. Our main contribution is in studying experimentally by simulations these increasingly more complex dynamic policies.

To gain insight into the performance of the different policies, we conducted an extensive set of numerical experiments. The basic setting consisted of a system with one macrocell and two microcells with higher service rates than the macrocell. In this setting, we evaluated four different scenarios having both symmetric and asymmetric arrival rates to the cells. Other aspects that were studied include the impact of more variable service time distributions, increasing the number of microcells and higher degree of asymmetry in the service rates.

Our main findings were the following. All dynamic policies can significantly improve the flow-level delay performance compared to the optimal static policy. Among the implemented dynamic policies, the classical Join the Shortest Queue (JSQ) policy typically performs worst. On the other hand, the so-called MP policy, which relies on detailed knowledge of remaining service times and is optimal when there are no more future arrivals, appears to be systematically the best. However, the much simpler heuristic policy, MJSQ, is able to achieve almost the same performance but with knowledge of number of flows and average service rates only. Optimized policies (FPI-U and FPI-A) are not able to give any essential improvements over MJSQ either. Thus, we believe that the MP policy can be very close to optimal in minimizing the mean flow delay. Finally, it appears that the performance gain of most of the dynamic polices (except LWL) is at least approximately insensitive with respect to the flow size distribution.

Future work for the problem includes, for example adding spatial features in the traffic model. This would allow a more detailed representation of the actual service time that consists of a model for the achievable service rate of a user at a given location. This information could also be utilized in the load balancing policy to develop more efficient policies. Also, considering in-band configurations of the system is one direction, where one then needs to consider the interference between the cells. Of course, the overall question of optimal policies is still completely open, even for the basic model introduced in this paper.

Acknowledgment

This research has been partially supported by the HEWINETS (Dynamic Heterogeneous Wireless Access Networks) project, funded by Ericsson, Cassidian Systems and TEKES.

References

[1] Deployment strategies for Heterogeneous Networks, *White paper*, Nokia-Siemens Networks, May 2012. Available online: http://br.nokiasiemensnetworks.com/taxonomy/term/85

[2] T. Bonald and A. Proutiere, Wireless downlink data channels: user performance and cell dimensioning, in *Proc. of ACM MobiCom*, Sep. 2003

[3] M. Harchol-Balter, *Performance Modeling and Design of Computer Systems: Queueing Theory in Action*, Cambridge University Press, 2013

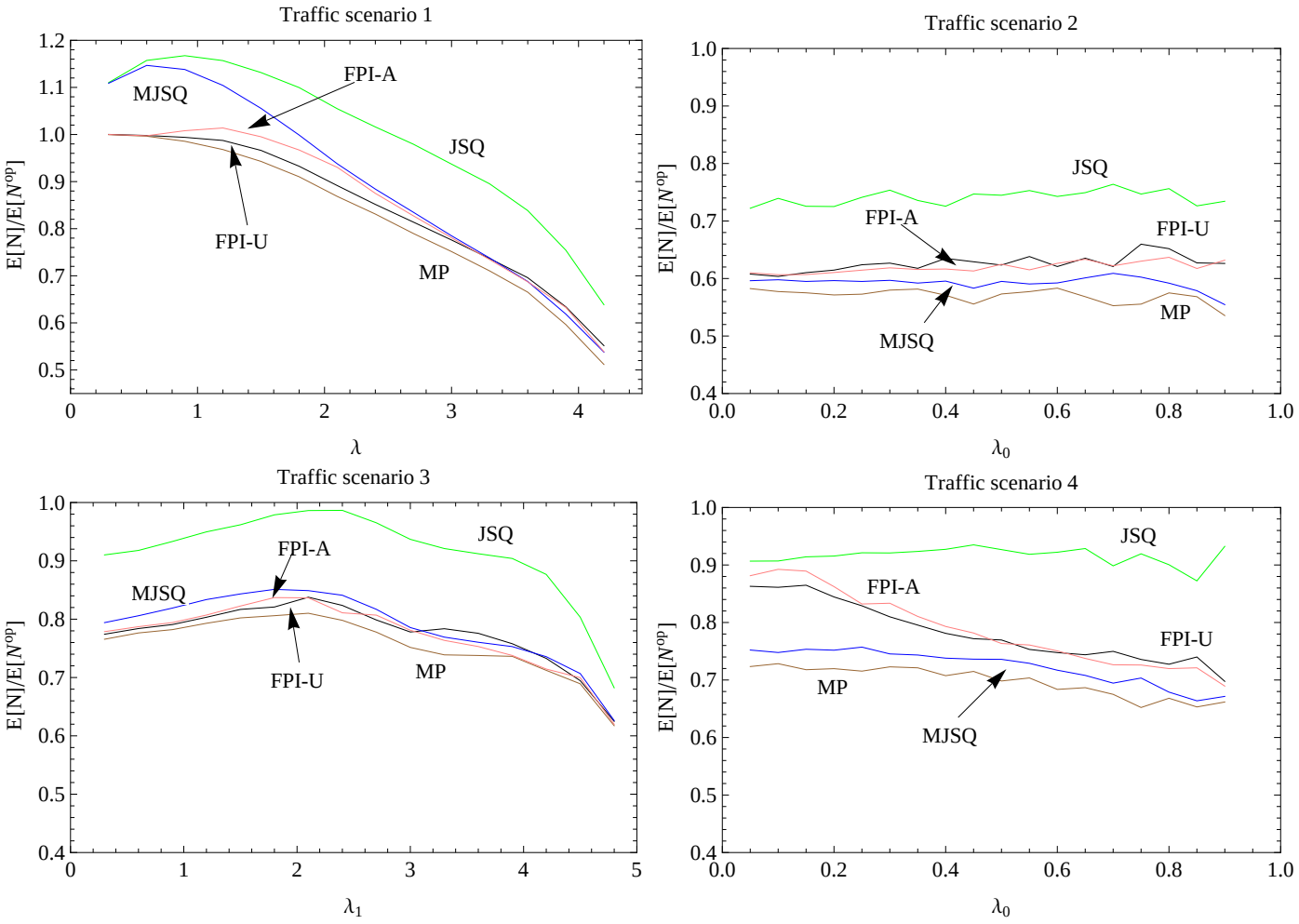

Fig. 6. Impact of service rate difference between macrocell and microcells for Traffic scenarios 1-4

[4] A. N. Tantawi and D. Towsley, Optimal static load balancing in distributed computer systems, *Journal of the ACM*, vol. 32, no. 2, Apr. 1985

[5] F. Bonomi and A. Kumar, Adaptive optimal load balancing in a nonhomogeneous multi-server system with a central job scheduler, *IEEE Transaction on Computers*, vol. 38, no. 11, Oct. 1990

[6] W. Winston, Optimality of the shortest line discipline, *Journal of Applied Probability*, vol. 14, no. 7, 1977

[7] F. Bonomi, On job assignment for a parallel system of processor sharing queues, *IEEE Transaction on Computers*, vol. 39, no. 7, Jul. 1990

[8] E. Hyytiä, A. Penttinen, S. Aalto, and J. Virtamo, Dispatching problem with fixed size jobs and processor sharing discipline, in *Proc. of ITC 23*, Sep. 1998

[9] M. E. Crovella, M. Harchol-Balter, and C. D. Murta, Task assignment in a distributed system: Improving the performance by unbalancing the load, in *Proc. of ACM SIGMETRICS*, Jun. 1998

[10] M. Harchol-Balter, M. E. Crovella, and C. D. Murta, On choosing a task assignment policy for a distributed server system, *Journal of Parallel and Distributed Computing*, vol. 59, no. 2, Nov. 1999

[11] H. Feng, V. Misra, and D. Rubenstein, Optimal state-free, size-aware dispatching for heterogeneous M/G/-type systems, *Performance Evaluation*, vol. 62, no. 1-4, 2005.

[12] E. Altman, U. Ayesta, and B. J. Prabhu, Load balancing in processor sharing systems, *Telecommunication Systems*, vol. 47, no. 1-2, 2012.

[13] G. Ning, G. Zhu, Q. Li, and R. Wu, Dynamic load balancing based on sojourn time in multitier cellular systems, in *Proc. of IEEE VTC 2006-Spring*, vol. 1, May 2006

[14] J. Lui, R. R. Muntz, and, D. Towsley, Bounding the mean response time of the minimum expected delay routing policy: An algorithmic approach, *IEEE Transaction on Computers*, vol. 44, no. 7, 1995

[15] S. M. Ross, *Applied Probability Models with Optimization Applications*, Holden-Day, 1970

[16] S. Aalto and J. Virtamo, Basic packet routing problem, in *Proc. of NTS-13*, Trondheim, Norway, Aug. 1996.

[17] E. Hyytiä, J. Virtamo, and S. Aalto, M/M/1-PS queue and size-aware task assignment, *Performance Evaluation*, vol. 68, no. 11, Nov. 2011

[18] Z. Tari, J. Broberg, A. Y Zomaya, and R. Baldoni, A least flow-time first load sharing approach for distributed server farm, *Journal of Parallel and Distributed Computing*, vol. 65, no. 7, Jul. 2005