Multi-Resource Fairness: Objectives, Algorithms and Performance

Thomas Bonald, James Roberts

Multi-Resource Fairness: Objectives, Algorithms and Performance

Thomas Bonald*
Télécom ParisTech, Paris, France
thomas.bonald@telecom-paristech.fr

James Roberts
IRT-SystemX, Paris-Saclay, France
james.roberts@irt-systemx.fr

ABSTRACT

Designing efficient and fair algorithms for sharing multiple resources between heterogeneous demands is becoming increasingly important. Applications include compute clusters shared by multi-task jobs and routers equipped with middleboxes shared by flows of different types. We show that the currently preferred objective of Dominant Resource Fairness (DRF) has a significantly less favorable efficiency-fairness tradeoff than alternatives like Proportional Fairness and our proposal, Bottleneck Max Fairness. We propose practical algorithms to realize these sharing objectives and evaluate their performance under a stochastic demand model. It is shown, in particular, that the strategyproofness property that motivated the choice of DRF for an assumed fixed set of jobs or flows, is largely irrelevant when demand is dynamic.

Categories and Subject Descriptors

C.2.1 [**Computer communication networks**]: Network Architecture and Design—*packet-switching networks*; C.4 [**Performance of systems**]: [modeling techniques, performance attributes]

General Terms

Algorithms, Performance

Keywords

Dominant resource fairness, proportional fairness, bottleneck max fairness, multi-resource sharing, cluster computing

1. INTRODUCTION

Multi-resource fairness has recently received a lot of attention thanks mainly to two papers by A. Ghodsi and co-authors [6, 5]. In the present paper we argue that the objective of dominant resource fairness (DRF), advocated by Ghodsi *et al.* for resource sharing in compute clusters and routers equipped with middleboxes, respectively, is not preferable to more classical sharing objectives like proportional fairness (PF) which achieve a better efficiency-fairness tradeoff. Such objectives were discarded by Ghodsi *et al.* as being vulnerable to manipulation by users seeking to gain more than their fair share by falsely stating their requirements. The advocated *strategyproofness* property, possessed by DRF but not PF in a context of *static* demand, is not in fact discriminating when considering the more realistic context of *dynamic* demand.

Multi-resource sharing in a compute cluster consists in launching appropriate numbers of tasks of multi-task jobs. Each task of a given job has its particular requirements for CPU, RAM and other resources. In practice, various constraints on the placement of tasks on physical machines need to be taken into account but, for present purposes, we follow Ghodsi *et al.* [6] and assume resources of the same type are assembled in homogeneous pools. The question is, how should a central scheduler determine the numbers of simultaneous tasks to run for all the currently active jobs, ensuring efficiency and some measure of fairness.

Routers increasingly employ middleboxes to process packets and these constitute potential bottlenecks for flows, in addition to link bandwidth. Different flows have different per-packet resource requirements (e.g., some require complex packet processing, others none) and the issue here is what packet rates should be imposed for concurrent flows in order to efficiently and fairly share all types of resource of these software routers.

Bandwidth sharing in a network is a particular form of multi-resource sharing. The problem is generally simpler since all flows sharing a given link are assumed to have identical requirements. This assumption is not true, however, for some wireless links where the resource to be allocated is spectrum time and the amount required to realize a given bit rate varies considerably between terminals, depending on their particular radio conditions. It is important to share wireless and wired links in a manner that adequately balances efficiency and fairness.

We have already made the case in a recent paper for proportional fair sharing of compute cluster resources [2]. Here we extend those arguments to the case of router resources and networks and introduce a new allocation objective called bottleneck max fairness (BMF). BMF is a pragmatic alternative to PF to which it has similar performance in all three application areas.

*Both authors are members of the LINCS, Paris, France. See www.lincs.fr.

In addition to the introduction of BMF, the main contributions of the paper are

- a characterization of the properties of BMF including, in Theorem 1, a first result on the stability conditions of this sharing objective,
- practical algorithms allowing the implementation of PF and BMF allocations, the algorithms for BMF being rather simpler than those for PF,
- results on the impact of falsely declared resource requirements, illustrating the practical strategyproofness of PF and BMF allocations under dynamic demand.

We first describe DRF, PF and BMF sharing objectives in abstract terms common to the above three application contexts. We define the corresponding allocations and discuss their significant characteristics. Algorithms are then discussed for realizing the respective allocations and we propose practical, low-complexity algorithms to realize PF and BMF sharing in compute clusters, software routers and networks, respectively. The following section on performance includes an important theorem on BMF stability and presents numerical results that justify our claim that DRF is not in fact the preferred objective. Under dynamic demand, where jobs or flows arrive over time and have finite size, completion times are consistently and significantly smaller with either PF or BMF allocations. We further highlight the implausibility of players being able to game the system when these allocations are applied to a dynamic population of jobs or flows.

2. OBJECTIVES

We define DRF, PF and BMF multi-resource sharing objectives with respect to a fluid model where resources are assumed infinitely divisible. The model is abstract and, rather than jobs or flows, we refer to *transactions*. These are assumed divisible into a large number of infinitesimal components (representing tasks or packets, for instance) having homogeneous resource usage characteristics.

2.1 The fluid model

Consider R infinitely divisible resources of normalized capacity 1 to be shared by n transactions indexed by i. Each transaction i requires amounts of each resource in fixed proportions and the amount allocated determines the rate at which the transaction progresses. We denote the requirements of transaction i by a vector $a_i = (a_{i1}, \ldots, a_{iR})$. We say that transaction i requires resouce j if $a_{ij} > 0$. A resource j for which $a_{ij} = \max_k\{a_{ik}\}$ is called a *dominant resource* for transaction i.

An allocation is defined by a vector of real numbers $\varphi = (\varphi_1, \ldots, \varphi_n)$ such that $\varphi_i a_{ij}$ is the fraction of resource j allocated to transaction i. The allocation must satisfy capacity constraints:

$$\sum_{i=1}^{n} \varphi_i a_{ij} \leq 1, \quad \text{for } j = 1, \ldots, R. \tag{1}$$

We say that resource j is *saturated* if $\sum_{i=1}^{n} \varphi_i a_{ij} = 1$. We refer to such a resource as a bottleneck.

2.2 Sharing properties

The allocation φ should arguably possess the following properties [6, 2]:

Pareto-efficiency. Each transaction requires some resource which is saturated.

Envy-freeness. No transaction should prefer the allocation of some other transaction.

Sharing-incentive. Each transaction gets no less than a $1/n$ share of its dominant resource.

Single-resource fairness. Resource shares $\varphi_i a_{i1}$ are all equal to $1/n$ when $R = 1$.

Single-bottleneck fairness. The shares $\varphi_i a_{ij}$ of resource j are all equal to $1/n$ whenever j is the only bottleneck.

Strategyproofness. The resource shares of any transaction do not increase if this transaction modifies its vector of resource requirements.

Scale-invariance. The resource shares remain the same when the vector of resource requirements of any transaction is multiplied by a scaling factor.

It is straightforward to check that single-resource fairness is a rather crucial requirement in that it is realized by any Pareto-efficient allocation that fulfills any one of the other properties.

2.3 Dominant resource fairness

Dominant resource fairness (DRF) is the unique Pareto efficient allocation where transactions obtain fractions of their dominant resource that are as equal as possible. Equality may not be possible if some transactions require only a subset of the resources [15]. DRF in fact corresponds to *weighted max-min fairness* where the weight of transaction i is inversely proportional to its dominant resource requirement $\max_j\{a_{ij}\}$.

DRF satisfies all properties listed in §2.2 except single-bottleneck fairness [6, 2]: even in the presence of a single bottleneck, the shares of this resource are not equal and depend on requirements for non-saturated resources. The main advantage of DRF over other allocations is that it is *strategyproof*. This means a transaction cannot gain a greater share of resources by boosting some component of its requirement vector.

2.4 Proportional fairness

An allocation φ is proportional fair (PF) if, for any other allocation φ' satisfying (1), the sum of proportional changes is non-positive, $\sum_i (\varphi'_i - \varphi_i)/\varphi_i \leq 0$ [12]. Equivalently, φ maximizes $\sum_i \log \varphi_i$ subject to (1) and can thus be said to maximize social welfare assuming a logarithmic utility function.

While PF was introduced in [12] as an objective for sharing bandwidth in a wired network, it was advocated independently by Tse and co-authors as the preferred allocation for a time-shared wireless downlink channel [19].

By the Karush-Kuhn-Tucker theorem, PF is uniquely characterized in terms of Lagrange multipliers ν_j. We have,

$$\frac{1}{\varphi_i} = \sum_{j=1}^{R} a_{ij} \nu_j, \tag{2}$$

for $i = 1, \ldots, n$, where $\nu_j \geq 0$ and $\nu_j > 0$ if and only if resource j is a bottleneck.

PF is clearly Pareto-efficient. In fact, it satisfies all properties of §2.2 except strategyproofness. The proof of envy-freeness is given below; other properties are proved in [2].

PROPOSITION 1. *The PF allocation is envy-free.*

PROOF. For any two transactions i_1, i_2, it follows from (2) that

$$\sum_{j=1}^{R} \nu_j(\varphi_{i_1} a_{i_1 j} - \varphi_{i_2} a_{i_2 j}) = 0.$$

Since not all Lagrange multipliers are zero, this implies that $\varphi_{i_1} a_{i_1 j} \geq \varphi_{i_2} a_{i_2 j}$ for some resource j: transaction i_1 does not prefer the allocation of transaction i_2. □

PF is the only utility maximizing allocation that is single-resource fair [2]. In particular, none of the properties except Pareto-efficiency is satisfied by a utility maximizing allocation other than PF.

To see why PF is not strategyproof, consider the following example: we have $R = 2$ resources and $n = 2$ transactions with requirements $a_1 = (1/2, 1)$ and $a_2 = (1, 1/2)$. The PF allocation is then $\varphi = (2/3, 2/3)$. If transaction 1 claims requirement $a_1' = (2/3, 1)$, we find $\varphi' = (3/4, 1/2)$ yielding a bigger share for transaction 1 at the expense of transaction 2. Note, however, that if a user does not know the other requirements, an ill-chosen modification can instead lead to it receiving a smaller share. Continuing the above example, if transaction 1 claims $a_1' = (1, 1)$, the PF allocation would be $\varphi' = (1/2, 1/2)$. Both transactions lose out compared to the result of the truthful declaration, $a_1 = (1/2, 1)$.

It is interesting to note that, for $n = 2$, one can characterize the maximum gain a transaction can obtain by a false declaration. We have the following proposition.

PROPOSITION 2. *For $n = 2$ transactions, the relative gain obtained by a transaction declaring false requirements cannot exceed 50% under PF.*

PROOF. Without loss of generality, we can assume that $R = 2$ (whenever $R > 2$, one of the resources is never saturated since its requirements are dominated by those of another resource). By scale invariance, we can take $a_{12} = a_{21} = 1$ and $a_{11}, a_{22} \leq 1$. Assume transaction 1 declares false requirement a_{11}' on resource 1. If $\frac{1}{2}(1/a_{11} + a_{22}) < 1$ then resource 1 is the single bottleneck and no winning strategy exists. Otherwise, resource 2 is a bottleneck for the true requirement and transaction 1 gets the maximum share of this resource, given by:

$$\varphi_1 = \min\left(\frac{1}{2}, \frac{1 - a_{22}}{1 - a_{11}a_{22}}\right).$$

The best strategy for transaction 1 is to increase a_{11} until it gets the maximum share of both resource 1 and resource 2, that is to set

$$a_{11}' = \frac{1}{2 - a_{22}},$$

in which case the allocation is given by

$$\varphi_1' = 1 - \frac{a_{22}}{2}.$$

It can be verified that the relative gain satisfies

$$\frac{\varphi_1' - \varphi_1}{\varphi_1} \leq \frac{1}{2},$$

with equality for $a_{11} = 0$, $a_{22} = 1/2$, that is $a_{11}' = 2/3$. □

2.5 Bottleneck max fairness

Dolev *et al.* proposed the "no justified complaints" objective as an alternative to DRF [4]. To be concise, we adopt a simplified version of this objective: a transaction has no justified complaint if it receives at least a fraction $1/n$ of at least one bottleneck. Pareto efficient allocations fulfilling this objective for all users are said to realize "bottleneck-based fairness" [24, 9].

There are in fact many allocations that are bottleneck-based fair. We restrict the class of such allocations somewhat by requiring in addition that every transaction i receives an allocation $\varphi_i a_{ij}$ of some bottleneck resource j that is *maximal* for that resource. We call such allocations "bottleneck max fair" (BMF).

Discussions on bottleneck-based fairness in [4] and [9] suggest the existence of a BMF allocation is far from obvious. The following proposition, proved in the appendix, establishes the existence of a BMF allocation.

PROPOSITION 3. *A BMF allocation always exists.*

Like PF, BMF meets all properties of §2.2 except strategyproofness. It is Pareto-efficient by definition and envy-free since, for any transaction i, there is some resource j such that $\varphi_i a_{ij}$ is maximal over all transactions. The no justified complaints objective clearly implies BMF has the sharing-incentive property. It is both single-resource fair and single-bottleneck fair since all transactions attain the maximum allocation of the resource in question and this is necessarily equal to $1/n$. It is scale-invariant since the definition involves the resource shares $\varphi_i a_{ij}$ only.

It turns out that BMF is not necessarily unique. The following proposition, also proved in the appendix, shows however that BMF is unique for $R = 2$ resources.

PROPOSITION 4. *The BMF allocation is unique for two resources.*

That uniqueness does not extend to more than 2 resources is established by the following counter-example. Consider a 3 resource system with 3 transactions having requirement vectors (1, 1, 1), (1, 1/2, 3/4) and (1/2, 1, 3/4). Allocations $\varphi^{(0)} = (2/5, 2/5, 2/5)$ and $\varphi^{(1)} = (1/3, 4/9, 4/9)$ both satisfy the definition. In fact, allocations $\varphi^{(x)} = (2/5 - x/15, 2/5 + 2x/45, 2/5 + 2x/45)$ for $0 \leq x \leq 1$ are all BMF. Note that the absence of uniqueness is not necessarily problematic since all the allocations may be considered satisfactory with respect to the considered properties.

The fact that BMF is not strategyproof follows on observing that BMF coincides with PF for $n = 2$ transactions, so that the counterexample of §2.4 applies to BMF as well. BMF and PF in fact coincide for any two homogeneous populations of transactions.

PROPOSITION 5. *BMF coincides with PF for two types of transactions.*

PROOF. We assume without loss of generality that $R = 2$ (see the proof of Proposition 2). Take n_1 transactions with resource requirements a_1 and n_2 transactions with resource requirements a_2. By scale invariance, we can assume that $a_{11}a_{22} \neq a_{12}a_{21}$ (otherwise, there is actually a single type of transaction and each transaction gets a fraction $1/n$ of the most constraining resource).

Assume resource 1 is the single bottleneck. By single-bottleneck fairness, the allocation φ is the same under PF and BMF and given by $\varphi_1 a_{11} = \varphi_2 a_{21} = 1/n$. Since resource 2 is not saturated, we obtain

$$\frac{1}{n}\left(n_1\frac{a_{12}}{a_{11}} + n_2\frac{a_{22}}{a_{21}}\right) < 1. \tag{3}$$

Similarly, resource 2 is the single bottleneck if and only if

$$\frac{1}{n}\left(n_1\frac{a_{11}}{a_{12}} + n_2\frac{a_{21}}{a_{22}}\right) < 1, \tag{4}$$

in which case PF and BMF coincide. Now assume that neither (3) nor (4) hold. Then both resources are bottlenecks under both BMF and PF. We deduce that

$$\begin{cases} n_1\varphi_1 a_{11} + n_2\varphi_2 a_{21} = 1, \\ n_1\varphi_1 a_{12} + n_2\varphi_2 a_{22} = 1, \end{cases}$$

which has a unique solution whenever $a_{11}a_{22} \neq a_{12}a_{21}$. □

It is worth noting that the proof of the above proposition is based on the following three properties of BMF and PF only: scale-invariance, Pareto efficiency and single-bottleneck fairness. This means in particular that, for two types of transaction, there is no other allocation that satisfies these three properties. By the counterexample of §2.4, this allocation is not strategyproof, which shows that there is no allocation meeting all properties listed in §2.2. Any Pareto-efficient, scale-invariant allocation is either strategyproof (like DRF) or single-bottleneck fair (like PF and BMF).

3. ALGORITHMS

We discuss algorithms to realize DRF, PF and BMF, successively for the fluid model, for a compute cluster shared by multi-task jobs and for router or network resources shared by flows of packets.

3.1 Dominant resource fairness

Since DRF corresponds to max-min fair sharing with weights equal to the dominant resource requirements, the allocation can be computed through a water-filling algorithm [6, 15]: the φ_i are increased at rates inversely proportional to the dominant resource requirements $\max_j\{a_{ij}\}$ until some resource is fully used; transactions using that resource are frozen while rates of the others with non-zero requirements on non-saturated resources are further increased at their respective rates until a second resource is full; the process continues until all the φ_i are frozen. The complexity of this algorithm is of order $O(Rn)$.

Ghodsi *et al.* [6] show how the DRF allocation can be realized in practice for compute cluster resources. As resources are freed on task completion they are re-allocated preferentially to the most deprived job. This is the job with the smallest share on its dominant resource. Tasks are assigned when jobs arrive, if resources are available, or when other tasks end. If available resources are insufficient to accommodate a task of the most deprived job, allocations are frozen until a sufficient number of other tasks end or some other job takes over the most-deprived status. Note that we continue to assume resources are pooled though the assignment algorithm could be adapted to account for additional practical constraints such as assigning CPU and RAM on the same physical server, say [16], or meeting other compatibility constraints [7].

To share router resources between flows, we can apply the DRFQ algorithm defined by Ghodsi *et al.* [5]. To highlight parallels with similar algorithms defined below for PF and BMF, we adopt a slightly different specification of DRFQ.

DRFQ approximately realizes weighted max-min fairness by applying the start-time fair queuing (SFQ) algorithm at each resource [8]. This algorithm serves waiting packets in increasing order of their virtual start time. The virtual start time S_{ij}^k of packet k of flow i at resource j is determined recursively,

$$S_{ij}^k = \max\left(V_j(u_{ij}^k), S_{ij}^{k-1} + \max_j\{a_{ij}\}\right), \tag{5}$$

where u_{ij}^k is the packet arrival time at resource j and the virtual time functions $V_j(t)$ of real time t are set equal to the largest start time at t of any packet to have begun service at that resource. Each resource needs to maintain a sorted list of virtual start times, yielding a complexity of $O(\log n)$ per packet arrival.

As long as each resource is aware of the weight to apply (i.e., the dominant resource requirement $\max_j\{a_{ij}\}$), the scheduling can be applied independently at each resource, even when the resources are separated geographically as in the network application. There is no constraint on the order in which resources are used and processing can proceed in parallel, if possible, without compromising the DRF properties.

Note that the virtual time functions determine how the first packet of a burst is inserted into the schedule while the second term in the max operator of (5) ensures fairness between flows that are backlogged. DRFQ basically inherits the latency and fairness properties of SFQ and is satisfactory as long as packet processing times are small, as is usual.

3.2 Proportional fairness

The optimization problem defining PF is computationally expensive in general. An approximate solution can be obtained by the gradient descent algorithm applied iteratively to the Lagrange multipliers,

$$\nu_j \leftarrow \max\left(\nu_j + \theta\left(\sum_{i=1}^n \psi_i a_{ij} - 1\right), 0\right), \tag{6}$$

where $\theta > 0$ is a sufficiently small step size and

$$\psi_i = \frac{1}{\sum_{j=1}^R a_{ij}\nu_j},$$

for $i = 1, \ldots, n$. The allocation is then given by (2).

This approximate solution can be used in a practical task-based algorithm for sharing a compute cluster. This consists in applying the same "most deprived job" approach from §3.1. Specifically, the most deprived job is that whose ratio of actual to ideal allocation is smallest.

An algorithm for packet-based PF can be derived on extending the analysis of Massoulié and Roberts for network bandwidth sharing [14]. This relies on an assumption that

resources are used consecutively and not in parallel. Each flow maintains a fixed window of W packets. This window might be realized within a router by creating an ingress queue and only admitting packet k to any internal buffer when packet $k-W$ has finished processing at all resources. Denote the number of flow i packets waiting or in service at resource j by Q_{ij} and let $Q_j = \sum_i Q_{ij}$. We have the conservation equation,

$$W = \sum_{j=1}^{R} Q_{ij}. \tag{7}$$

Assume the system attains a stable regime where the Q_{ij} are positive constants when resource j is a bottleneck, or zero otherwise. This corresponds to the fluid limit regime where W tends to infinity, as considered for instance by Schweitzer [17] and Walton [20]. In practice, assuming constant packet sizes, it is sufficient that $W \geq R$ so that each flow is backlogged on some resource. It turns out that serving packets at rates proportional to Q_{ij}/a_{ij} yields the PF allocation. We have,

$$\sum_{i=1}^{n} \varphi_i a_{ij} = 1 \quad \Longrightarrow \quad \frac{Q_{kj}}{Q_{ij}} = \frac{\varphi_k a_{kj}}{\varphi_i a_{ij}}$$

and summing over k yields $Q_{ij} = \varphi_i a_{ij} Q_j$. On substituting for Q_{ij} in (7) we derive,

$$\frac{1}{\varphi_i} = \sum_j a_{ij} \frac{Q_j}{W}$$

where $Q_j \geq 0$ and $Q_j > 0$ if and only if $\sum_i \varphi_i a_{ij} = 1$. Comparison with (2) shows that the variables Q_j/W coincide with the Lagrange multipliers ν_j and, therefore, that φ is indeed the unique PF allocation.

To realize the required packet rates we can again adapt the SFQ algorithm. Start times S_{ij}^k are defined recursively and independently for each resource j,

$$S_{ij}^k = \max\left(V_j(u_{ij}^k), S_{ij}^{k-1} + a_{ij}/Q_{ij}\right), \tag{8}$$

where notation is as defined above in §3.1. Complexity is still $O(\log n)$.

This algorithm is not entirely satisfactory. The assumption that resources are visited sequentially by each packet is restrictive for the software router application. Moreover, for the network application, it would be necessary to take account of non-zero propagation times between visits to successive resources. This case is considered in [14] where it is shown that, even with homogeneous resource requirements, the algorithm only realizes PF approximately if the window W is large compared to the bandwidth delay product, i.e., the product of φ_i and the round trip propagation time of flow i.

3.3 Bottleneck max fairness

For small systems, it is possible to determine a BMF allocation by testing the feasibility of all possible one-to-one mappings of transactions to potential bottlenecks. For a mapping to be feasible, it must be possible to find values φ_i satisfying (1) such that every resource j with at least one mapped transaction is a bottleneck and all transactions mapped to j have the maximum share of j. Fortunately, it is not necessary to apply this algorithm in any practical implementation, as discussed below.

The most deprived job approach can be adapted to realize BMF in a compute cluster. For every job i we note its rank at each resource j: jobs with the biggest allocation have rank 1, jobs of rank k for $k > 1$ have a smaller share than $k-1$ other jobs. The deprivation status of job i is its highest rank on a bottleneck resource. A resource is considered a bottleneck here if its residual capacity is less than the maximal requirement for that resource over all transactions. Tasks are launched preferentially for the job whose current status is furthest from 1. Each resource maintains a sorted list of n jobs, yielding a complexity of $O(\log n)$ per task arrival or departure.

For the router application, BMF can be realized by imposing local weighted max-min fairness at each resource with respective weights $1/a_{ij}$. To demonstrate this, we again assume the existence of a steady state and extend arguments from [14]. We assume each flow i maintains a window of W_i packets and, to include the possibility of remotely located resources (as in the network application), we introduce the round trip propagation times T_i. The following conservation relations generalize (7),

$$W_i = \varphi_i T_i + \sum_{j=1}^{R} Q_{ij}. \tag{9}$$

Relation (9) shows that for every flow i, as long as W_i is sufficiently large, there is at least one bottleneck resource j such that $Q_{ij} > 0$ (assuming constant packet sizes, it is in fact sufficient that $W_i > \varphi_i T_i + R$ to ensure that flow i is backlogged at some resource). Moreover, the weighted fair queuing scheduler at j ensures $\varphi_i a_{ij} = \max_k(\varphi_k a_{kj})$, completing the BMF defining conditions.

To realize BMF using SFQ, start times must be calculated as follows,

$$S_{ij}^k = \max\left(V_j(u_{ij}^k), S_{ij}^{k-1} + a_{ij}\right), \tag{10}$$

where notation is again as defined above in §3.1. Complexity is $O(\log n)$.

Note that this algorithm is much simpler than that of PF in not requiring knowledge of queue lengths or imposing the same window for all flows. Like DRFQ, it can be applied independently of whether the resources are visited in parallel or sequentially and it does not depend on propagation times beyond the requirement that W_i be big enough to maintain a positive queue at bottlenecks.

4. PERFORMANCE

While the sharing algorithms are defined above with respect to a fixed set of n transactions, their behavior can only be realistically appraised under dynamic demand where transactions occur over time, each bringing a finite amount of work to be accomplished. We propose a simple Markovian demand model and use it to compare the performance of DRF, PF and BMF.

4.1 Markovian demand model

We suppose transactions belong to one of K classes and transactions of class k arrive as a Poisson process of rate λ_k. Class k transactions have requirement $a_k = (a_{k1}, \ldots, a_{kR})$ and bring a random amount of work with an exponential distribution with parameter μ_k. This work corresponds to the job duration for the computer cluster application and to the flow size in bits for the network application, for instance.

The state vector $\vec{n} = (n_1, \ldots, n_K)$, giving the current number of transactions of each class in progress, is then a Markov process with component-k birth rate λ_k and death rate $\phi_k \mu_k$, where $\phi = (n_1\varphi_1, \ldots, n_K\varphi_K)$ denotes the per-class allocation vector. The Markovian model is adopted for the sake of simplicity. We expect, however, in view of the approximate insensitivity of fair resource sharing systems, that the comparative performance of DRF, PF and BMF will be similar under more accurate demand models (see [1], for example, for a discussion on the insensitivity property in the context of bandwidth sharing).

Capacity constraints (1) can be written

$$\sum_{k=1}^{K} \phi_k a_{kj} \leq 1, \tag{11}$$

for $j = 1, \ldots, R$. When the allocation ϕ is DRF or PF, known results for bandwidth sharing in networks (e.g., [23]) allow us to conclude that the above process is stable as long as loads $\rho_k = \lambda_k/\mu_k$ satisfy the following inequalities,

$$\sum_{k=1}^{K} \rho_k a_{kj} < 1, \tag{12}$$

for $j = 1, \ldots, R$. This follows for DRF since the objective is just a particular weighted max-min allocation. The BMF allocation is not included in the class of allocations considered by Ye [23] and it proves difficult to establish the stability condition for a general system. The following theorem, proved in the appendix, shows that condition (12) is sufficient for a system limited to two resources with homogeneous mean service requirements, $\mu_1 = \ldots = \mu_K$. We believe that, like DRF and PF, this result in fact extends to any number of resources and is insensitive to the distributions of service requirement beyond the means.

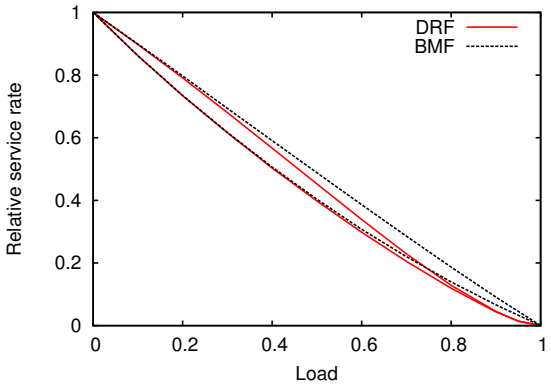
THEOREM 1. *Conditions* (12) *are sufficient to ensure the stability of BMF in the case of two resources with homogeneous mean service requirements.*

In the following we assume the system is stable and has a stationary distribution $\pi(\vec{n})$ from which we can compute performance measures like expected completion times. We compare algorithm performance via the mean service rate γ_k, defined as the ratio of the mean work $1/\mu_k$ to the mean completion time. Using Little's law, we find $\gamma_k = \rho_k/\mathbb{E}(n_k)$ where $\mathbb{E}(n_k)$ is the mean number of class k transactions in progress. Observe that, if there are only class-k transactions with normalized dominant resource requirement, that is $\max_j a_{kj} = 1$, the model reduces to a processor-sharing queue with arrival rate λ_k and service rate μ_k. In particular, $\mathbb{E}(n_k) = \rho_k/(1-\rho_k)$ and $\gamma_k = 1 - \rho_k$: the relative service rate decreases linearly with the system load.

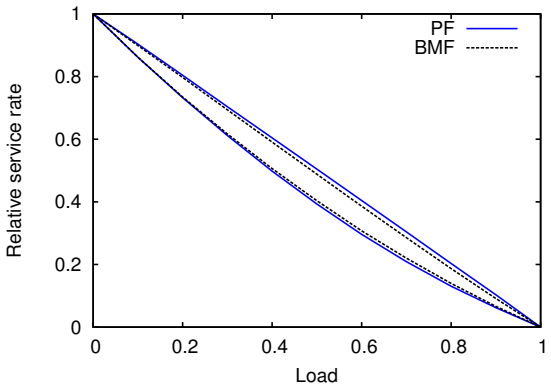
4.2 Performance of the fluid model

We illustrate the relative performance of DRF, PF and BMF using a numerical example. Two resources are shared by 3 classes of transaction with requirement vectors $a_1 = (.1, 1)$, $a_2 = (1, .1)$, $a_3 = (1, 1)$. The figures plot realized service rates γ_k against the load of the heaviest loaded resource, $\text{argmax}_{j=1,2} \sum_{k=1}^{K} \rho_k a_{kj}$.

Figure 1 corresponds to balanced load $\rho_1 = \rho_2 = \rho_3$ and shows performance is similar for all three allocations. BMF is very close to PF while both are somewhat better than DRF, especially at high load.



(a) BMF *vs* DRF



(b) BMF *vs* PF

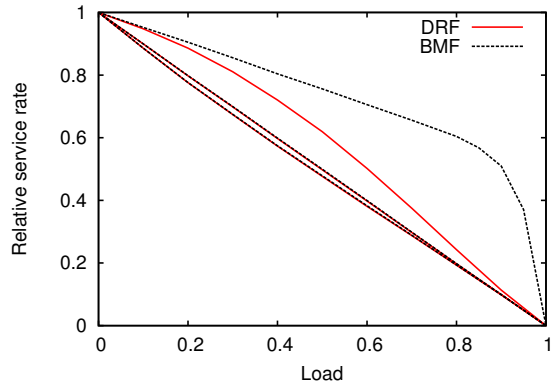
Figure 1: Service rates γ_k against resource load for BMF and DRF with balanced load: $a_1 = (.1, 1)$, $a_2 = (1, .1)$, $a_3 = (1, 1)$; $\rho_1 = \rho_2 = \rho_3$; $\gamma_1 = \gamma_2 > \gamma_3$.

The difference in performance is accentuated under unbalanced load, as illustrated in Figure 2 for the case $\rho_1 = 4\rho_2 = 4\rho_3$. In this scenario resource 1 has less than half the load of resource 2. The plots show that strict fairness imposed by DRF prevents class 2 transactions from fully exploiting this. The service rates of PF and BMF are roughly equivalent, both yielding a better efficiency-fairness tradeoff with a significant gain in γ_2 for negligible reductions in γ_1, γ_3.

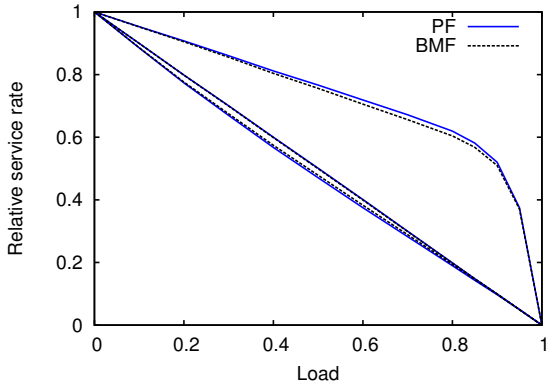
The generality of these results for DRF and PF was discussed in [2] where light and heavy load behavior was determined for two resources and two transaction classes. It is noteworthy that the performance advantage of PF for unbalanced load arises due to the fact that DRF sacrifices single-bottleneck fairness in the interest of strategyproofness.

4.3 Strategyproofness

The strategyproof property in §2.2 relates to a static population of transactions. We demonstrate in this subsection that the absence of this property for PF and BMF is hardly a disadvantage in dynamic traffic. Even if some transaction knows some statistics about the resource requirements of competing transactions, it does not know how many transactions are active or how this number will evolve and thus can hardly define a consistent strategy to improve its service rate by lying about its resource requirements.

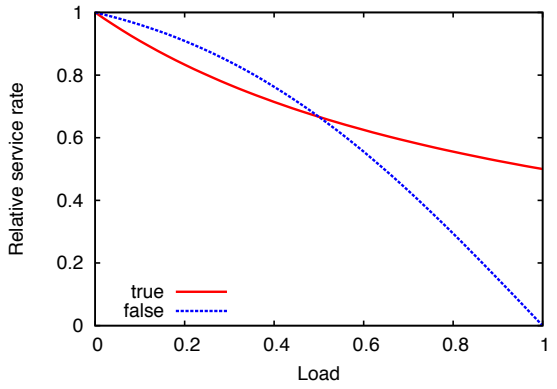


(a) BMF *vs* DRF

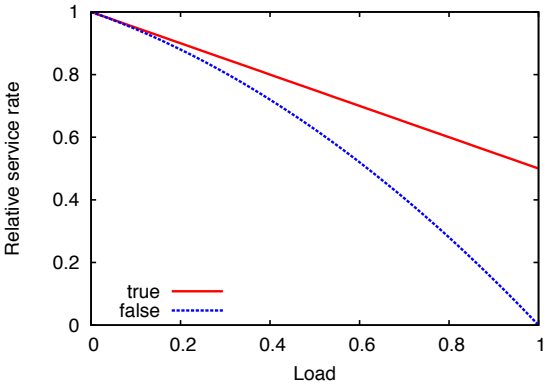


(b) BMF *vs* PF

Figure 2: Service rates γ_k against resource 2 load for BMF and DRF with unbalanced load: $a_1 = (.1, 1)$, $a_2 = (1, .1)$, $a_3 = (1, 1)$; $\rho_1 = 4\rho_2 = 4\rho_3$; $\gamma_2 > \gamma_1 > \gamma_3$.



(a) Very short transaction



(b) Very long transaction

Figure 3: Service rate of a test transaction against class 2 load: the test transaction requires (0,1) but falsely declares (2/3,1) and competes with a dynamic population of transactions of profile (1,1/2).

Consider the potential for gaming when some test transaction competes with a homogeneous population of other transactions evolving according to the model of §4.1. Recall from Proposition 5 that PF and BMF allocations are identical for this case (since only two resources can be limiting).

From Proposition 2, the greatest gain for two resources and two transactions is obtained for $a_1 = (0, 1)$ and $a_2 = (1, 1/2)$ with transaction 1 falsely declaring $a_1' = (2/3, 1)$. Suppose first that transaction 1 truly declares $a_1 = (0, 1)$. The service rate of transaction 1 is then $\phi_1 = 1$ for $n_2 = 0$ and $\phi_1 = 1/2$ otherwise, while $\phi_2 = 1$ for any n_2. In particular, n_2 has a geometric distribution with parameter ρ_2. If transaction 1 is infinitesimally small, n_2 remains constant during its service. We deduce that the mean duration of transaction 1 is proportional to $1 - \rho_2 + 2\rho_2$ for an infinitesimally small size, yielding

$$\gamma_1 = \frac{1}{1 + \rho_2}.$$

If transaction 1 is infinitely large, the service rates average and we get

$$\gamma_1 = 1 - \rho_2 + \frac{\rho_2}{2} = 1 - \frac{\rho_2}{2}.$$

Now assume transaction 1 claims requirement $a_1' = (2/3, 1)$. The service rate of transaction 1 is then $\phi_1 = 1$ for $n_1 = 0$ and $\phi_1 = 3/(2(n_2 + 1))$ otherwise, while $\phi_2 = n_2/(n_2 + 1)$ for any n_2. If transaction 1 is infinitesimally small then n_2 still has a geometric distribution with parameter ρ_2 and the mean duration of transaction 1 is proportional to

$$1 - \rho_2 + \frac{2}{3} \sum_{n_2 \geq 1} (n_2 + 1)(1 - \rho_2)\rho_2^{n_2},$$

yielding

$$\gamma_1 = \frac{1 - \rho_2}{1 - \frac{2}{3}\rho_2 + \frac{\rho_2^2}{3}}.$$

If transaction 1 is infinitely large, the probability of state $(1, n_2)$ is proportional to $(n_2 + 1)\rho_2^{n_2}$ so that

$$\gamma_1 = (1 - \rho_2)^2(1 + \frac{3}{2} \sum_{n_2 \geq 1} \rho_2^{n_2}) = (1 - \rho_2)(1 + \frac{\rho_2}{2}),$$

which is less that the original service rate for any load ρ_2.

Figure 3(a) shows how the service rate of transaction 1 behaves with the false declaration $a_1' = (2/3, 1)$ as a function of ρ_2 when this transaction occurs at an arbitrary instant and is infinitesimally small. The maximum gain in this configuration is only 10% compared to the 50% gain realized under static load. Moreover, this gain is realized only under

a very particular configuration and rapidly disappears as the worst case conditions are relaxed. For example, Figure 3(b) confirms that if transaction 1 is not small but very large, there is no gain from the false declaration at any load.

The fact that transaction 1's competitors all have the profile (1,1/2) further exaggerates the scope for illicit gain. Transaction 1 can take no additional rate from any concurrent transactions with the same dominant resource (i.e., having a profile $(a_{11}, 1)$ for $a_{11} \leq 1$). In any mixed demand scenario, it is clear that the maximum gain is considerably less than 10%. Moreover, we note from the results of Figure 3 that the false requirements only ever yield a marginal improvement in the smallest completion times. Longer completion times, because load is heavy, as on the right of Fig. 3(a), or because the transaction is intrinsically long, as in Fig. 3(b), will be increased and not decreased.

The above clearly does not show that PF and BMF are strategyproof. Indeed they are not in case of static demand. However, the results do suggest it is rather implausible that any transaction can define a winning strategy when demand is dynamic.

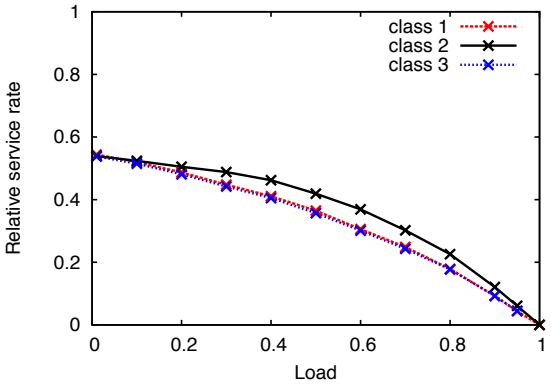
4.4 Task-based allocations

While the fluid models analyzed above are useful in appraising the relative merits of alternative fairness objectives, it is necessary in practice to account for the fact that jobs are not infinitely divisible. In this section we relax this assumption. We compare the performance of task-based versions of DRF, PF and BMF through the results of simulation experiments. We follow the proposal in [6] in seeking to realize the fairness objective by preferentially launching tasks of the most deprived job with respect to the objective fairness criterion. The respective criteria for defining deprivation status were presented in §3.

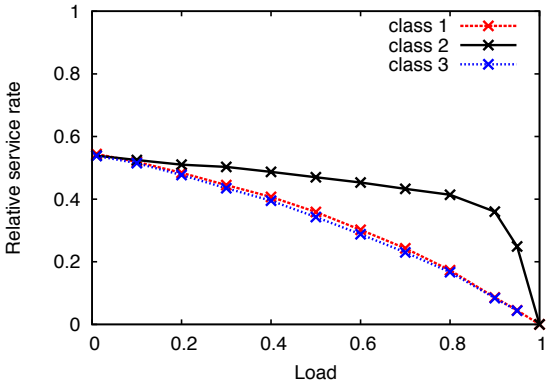
Figure 4 plots the results of simulations showing how the algorithms perform in a particular configuration. Two resources (e.g., CPU and RAM) are shared by jobs of three classes with profiles (.1,1), (1,.1) and (1,1) and demand is unbalanced with $\rho_1 = 4\rho_2 = 4\rho_3$. This is the same data used for the fluid model in Figure 2. The present results show the service rate is much smaller for the task-based allocation. This is due to the finite task size which invalidates the infinite divisibility assumption of the fluid model.

We have assumed here that each job has 500 tasks to run, each having an independent exponential duration, and that the capacities of CPU and RAM are 100. For very low load, each job is alone with high probability and 100 of its tasks are processed in parallel while there remain at least 100 to run. From then on, the task completion rate decreases with the number of remaining tasks in progress from 99 to 1. It is possible in this case to compute the maximum service rate of 0.54 [2]. The discrepancy with respect to the fluid model is smaller when the task duration is less variable or when the number of tasks to run is much greater than the resource capacity.

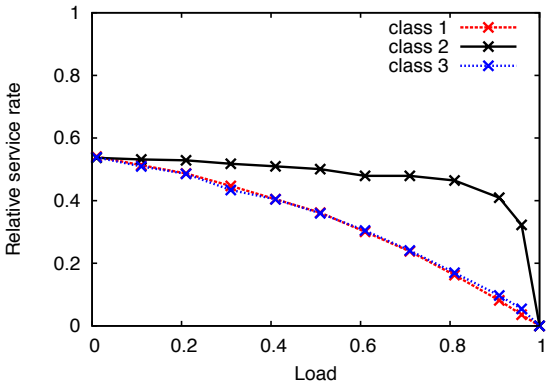
The figure shows that PF and BMF still significantly outperform DRF for the class of jobs whose dominant resource is the least loaded (class 2 and resource 1 in this case). The algorithm for BMF is rather more effective than that of PF and yields greater service rates whereas the fluid results were virtually the same.



(a) DRF



(b) PF



(c) BMF

Figure 4: Cluster sharing: service rates γ_k against resource 2 load for unbalanced load: $a_1 = (.1, 1)$, $a_2 = (1, .1)$, $a_3 = (1, 1)$; $\rho_1 = 4\rho_2 = 4\rho_3$.

4.5 Packet-based allocations

In this section we evaluate by simulation the effectiveness of the SFQ-based algorithms defined in §3 for router resource sharing. The simulations assume two resources (e.g., CPU and bandwidth) are used *successively* by flow packets. The number of packets in each flow has a geometric distribution of mean 20000. For all algorithms we set the window W to 30 for all flows. We have verified that these parameter choices do not critically impact the presented results.

Service rates are shown in Figure 5 for the same unbalanced load scenario used in the previous section. Crosses are results of packet-based simulations for 10^5 flow arrivals and lines are the fluid model results from Figure 2. The results show that all three packet-based algorithms closely approximate the service rates of the ideal allocations, confirming the performance advantage of PF and BMF over DRF.

From a practical point of view, BMF is preferable to PF in that the algorithm is simpler to implement and more robust. Like DRF, BMF sharing is also realized by the same algorithm when some resources can be used simultaneously and not sequentially. Crucially for the network application, BMF is also applicable when propagation times are non-zero.

5. RELATED WORK

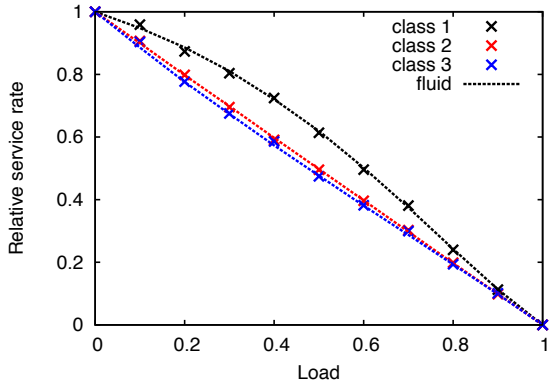
DRF [6] and "no justified complaints" [4] were placed in a more general economics framework in the work of Gutman and Nisan [9]. Joe-Wang *et al.* also generalized DRF by introducing two families of allocations that allow a controlled tradeoff between efficiency and fairness [11]. However, all these objectives are evaluated assuming static demand. It would be necessary to reappraise the impact of such generalizations under the more realistic assumption of dynamic, stochastic demand.

The "serve the most deprived job" approach introduced in [6] proves very versatile. It is used by Zeldes and Feitelson [24] to implement bottleneck-based fairness and by Ghodsi and co-authors [7] to account for compatibility constraints in task placement. Our proposed implementations of PF and BMF for sharing cluster resources are further illustrations of this versatility. It would be not difficult to account for possible placement constraints in these implementations.

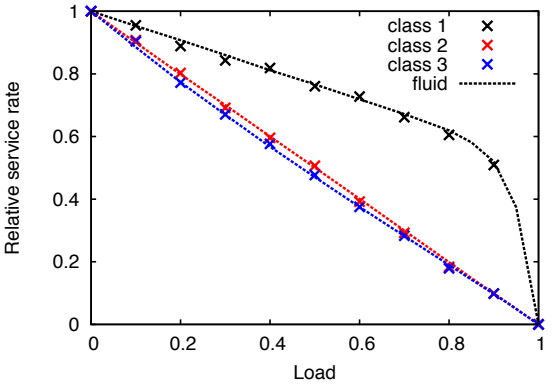
The packet-based algorithms designed by Ghodsi *et al.* [5] to realize DRFQ for shared router resources are based on start-time fair queuing. Wang and co-authors have proposed alternative realizations that adapt Deficit Round Robin [18] to the multi-resource context [22]. Our implementations of PF and BMF rely on SFQ though it would be straightforward to substitute alternative fair queuing mechanisms.

The need to evaluate the performance of resource sharing objectives under dynamic demand is still not widely recognized. The paper by Massoulié and Roberts [13] was perhaps the first to note the importance of this while some of the most significant subsequent findings are summarized by Bonald *et al.* [1]. Our earlier paper [2] and the present work extend this analysis to the domain of multi-resource sharing.

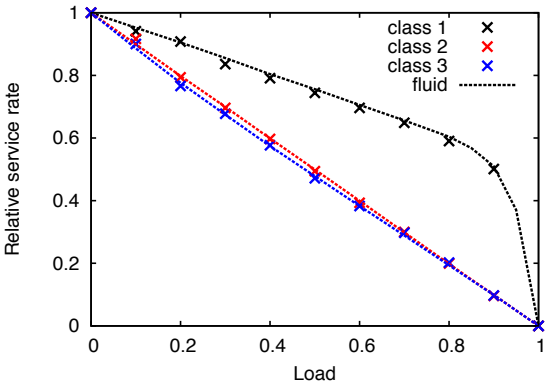
The stability of network bandwidth sharing under various allocation objectives, with and without Markovian demand assumptions, is discussed by Walton and Mandjes [21]. Their paper surveys the literature on the stability question and illustrates its inherent difficulty. While existing results can be applied to prove the stability of DRF and PF, this is not the case for BMF.



(a) DRF



(b) PF



(c) BMF

Figure 5: Router sharing: service rates γ_k against resource 2 load for unbalanced load: $a_1 = (.1, 1)$, $a_2 = (1, .1)$, $a_3 = (1, 1)$; $\rho_1 = 4\rho_2 = 4\rho_3$.

6. CONCLUSIONS

Multi-resource sharing for efficiency and fairness is an old issue in networking with challenging new variants occurring in the domains of cluster computing and software routers. Recent prominent publications have led to the emergence of DRF and its apparent acceptance as the preferred sharing objective. DRF is implemented in the Hadoop Next Generation Fair Scheduler, for instance.

We have argued in this paper that this popularity is misplaced since alternative objectives like PF display a better efficiency-fairness tradeoff. This result is revealed on considering the completion time performance of alternative objectives under the realistic and crucial assumption that demand is dynamic: jobs and flows occur over time and their completion times depend critically on the implemented resource sharing objective.

We have proposed BMF as a pragmatic alternative to PF. It has similar performance and can be realized more simply, especially when sharing router and network resources between flows. It is clearly the most practical objective for a network integrating radio access and wired backhaul. Independent schedulers simply share their own resource equitably. BMF thus generalizes network-wide max-min fairness that is known to be realized by fair schedulers acting independently on each link [10].

Concerns about the vulnerability of objectives like PF and BMF to malicious gaming have been shown to be largely unfounded. While users can manipulate allocations by falsely boosting their declared requirement of non-dominant resources, their gain depends critically on knowing the requirements of competitors. Such knowledge is hardly conceivable in the context of highly dynamic populations of active transactions occurring in a realistic model of demand. Moreover, we have shown that the falsification is counter-productive in most cases, especially for large transactions.

While BMF is relatively straightforward to implement, it remains difficult to completely characterize its essential properties. We have proved that the usual per-resource stability conditions are sufficient in a system limited to two resource types. However, the extension of this result to more than two resources is a challenging open problem.

7. REFERENCES

[1] T. Bonald, L. Massoulié, A. Proutière, and J. Virtamo. A queueing analysis of max-min fairness, proportional fairness and balanced fairness. *Queueing Syst. Theory Appl.*, 53(1-2):65–84, June 2006.

[2] T. Bonald and J. Roberts. Enhanced cluster computing performance through proportional fairness. *Performance Evaluation*, 79(0):134 – 145, 2014. Special Issue: Performance 2014.

[3] G. De Veciana, T.-J. Lee, and T. Konstantopoulos. Stability and performance analysis of networks supporting elastic services. *Networking, IEEE/ACM Transactions on*, 9(1):2–14, 2001.

[4] D. Dolev, D. G. Feitelson, J. Y. Halpern, R. Kupferman, and N. Linial. No justified complaints: On fair sharing of multiple resources. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ITCS '12, pages 68–75, New York, NY, USA, 2012. ACM.

[5] A. Ghodsi, V. Sekar, M. Zaharia, and I. Stoica. Multi-resource fair queueing for packet processing. In *Proceedings of ACM SIGCOMM 2012*, pages 1–12, New York, NY, USA, 2012. ACM.

[6] A. Ghodsi, M. Zaharia, B. Hindman, A. Konwinski, S. Shenker, and I. Stoica. Dominant resource fairness: Fair allocation of multiple resource types. In *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation*, NSDI'11, pages 24–24, Berkeley, CA, USA, 2011. USENIX Association.

[7] A. Ghodsi, M. Zaharia, S. Shenker, and I. Stoica. Choosy: Max-min fair sharing for datacenter jobs with constraints. In *Proceedings of the 8th ACM European Conference on Computer Systems*, EuroSys '13, pages 365–378, New York, NY, USA, 2013. ACM.

[8] P. Goyal, H. Vin, and H. Cheng. Start-time fair queueing: a scheduling algorithm for integrated services packet switching networks. *Networking, IEEE/ACM Transactions on*, 5(5):690–704, Oct 1997.

[9] A. Gutman and N. Nisan. Fair allocation without trade. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS '12, pages 719–728, Richland, SC, 2012. International Foundation for Autonomous Agents and Multiagent Systems.

[10] E. Hahne. Round-robin scheduling for max-min fairness in data networks. *Selected Areas in Communications, IEEE Journal on*, 9(7):1024–1039, Sep 1991.

[11] C. Joe-Wong, S. Sen, T. Lan, and M. Chiang. Multi-resource allocation: Fairness-efficiency tradeoffs in a unifying framework. In *INFOCOM, 2012 Proceedings IEEE*, pages 1206–1214, 2012.

[12] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan. Rate control for communication networks: Shadow prices, proportional fairness and stability. *The Journal of the Operational Research Society*, 49(3):pp. 237–252, 1998.

[13] L. Massoulié and J. Roberts. Bandwidth sharing and admission control for elastic traffic. *Telecommunication Systems*, 15(1-2):185–201, 2000.

[14] L. Massoulié and J. Roberts. Bandwidth sharing: Objectives and algorithms. *IEEE/ACM Trans. Netw.*, 10(3):320–328, June 2002.

[15] D. C. Parkes, A. D. Procaccia, and N. Shah. Beyond dominant resource fairness: extensions, limitations, and indivisibilities. In *ACM Conference on Electronic Commerce*, pages 808–825. ACM, 2012.

[16] C.-A. Psomas and J. Schwartz. Beyond beyond dominant resource fairness : Indivisible resource allocation in clusters. Tech Report Berkeley, 2013.

[17] P. J. Schweitzer. Bottleneck determination in networks of queues. In R. Disney and T. Ott, editors, *Applied Probability-Computer Science: The Interface Volume 1*, volume 2 of *Progress in Computer Science*, pages 471–485. Springer, 1982.

[18] M. Shreedhar and G. Varghese. Efficient fair queueing using deficit round robin. *SIGCOMM Comput. Commun. Rev.*, 25(4):231–242, Oct. 1995.

[19] P. Viswanath, D. Tse, and R. Laroia. Opportunistic beamforming using dumb antennas. *Information Theory, IEEE Transactions on*, 48(6):1277–1294, Jun 2002.

[20] N. S. Walton. Proportional fairness and its relationship with multi-class queueing networks. *The Annals of Applied Probability*, 19(6):2301–2333, 2009.

[21] N. S. Walton and M. R. Mandjes. A stability conjecture on bandwidth sharing networks. *Queueing Syst. Theory Appl.*, 68(3-4):237–250, Aug. 2011.

[22] W. Wang, B. Li, and B. Liang. Multi-resource round robin: A low complexity packet scheduler with dominant resource fairness. In *ICNP 2013*, 2013.

[23] H.-Q. Ye. Stability of data networks under an optimization-based bandwidth allocation. *Automatic Control, IEEE Transactions on*, 48(7):1238–1242, July 2003.

[24] Y. Zeldes and D. G. Feitelson. On-line fair allocations based on bottlenecks and global priorities. In *Proceedings of the 4th ACM/SPEC International Conference on Performance Engineering*, ICPE '13, pages 229–240, New York, NY, USA, 2013. ACM.

APPENDIX

Proposition 3 – Existence of BMF

It is convenient here to abandon the normalization of resource capacities and define the vector $C = (C_1, ..., C_R)$ associated with capacity constraints

$$\sum_{i=1}^{n} \varphi_i a_{ij} \le C_j, \quad \text{for } j = 1, \dots, R.$$

We prove by induction on n that there exists a BMF allocation φ which is a continuous function of the vector C. For $n = 1$ transaction, we take $\varphi_1 = \min_{j:a_{1j}>0} C_j/a_{1j}$, which is a continuous function of C. Now assume that the property holds for $n-1$ transactions, for some $n \ge 2$. For any non-negative $\theta < \min_{j:a_{1j}>0} C_j/a_{1j}$, there exists a bottleneck-max fair allocation $\varphi^{(\theta)}$, which is a continuous function of θ, for the system restricted to transactions $2, \dots, n-1$ and the capacity vector $C^{(\theta)} = C - \theta a_1$. For any sufficiently small θ, we have, for all $j = 1, \dots, R$,

$$\sum_{i=2}^{n} \varphi_i^{(\theta)} a_{ij} = C_j^{(\theta)} \quad \Longrightarrow \quad \theta a_{1j} < \max_{i=2,\dots,n} \{\varphi_i^{(\theta)} a_{ij}\}. \quad (13)$$

Let θ_1 be the largest number such that this property is satisfied for all $\theta < \theta_1$. Observe that, by the continuity of $\varphi^{(\theta)}$ in θ, this property is violated for $\theta = \theta_1$. In particular, there exists some resource j_1 such that

$$\sum_{i=2}^{n} \varphi_i^{(\theta_1)} a_{ij_1} = C_{j_1}^{(\theta_1)}$$

and

$$\theta_1 a_{1j_1} \ge \max_{i=2,\dots,n} \{\varphi_i^{(\theta_1)} a_{ij_1}\}. \quad (14)$$

Now define $\varphi_1 = \theta_1$ and $\varphi_i = \varphi_i^{(\theta_1)}$ for $i = 2, \dots, n$. In view of (14), transaction 1 gets the maximum share of bottleneck resource j_1 under allocation φ. Now take any increasing sequence $\theta(1), \theta(2), \dots$ tending to θ_1 such that for each transaction $i = 2, \dots, n$, there is some bottleneck resource j_i in the system restricted to transactions $2, \dots, n$ whose transaction-i share $\varphi_i^{(\theta)} a_{ij_i}$ is maximum over all transactions $2, \dots, n$, for all $\theta = \theta(1), \theta(2), \dots$ Such a sequence exists because $\varphi^{(\theta)}$ is bottleneck-max fair and there is a finite number of resources. It follows from (13) that $\theta a_{1j_i} < \varphi_i^{(\theta)} a_{ij_i}$ for all transactions $i = 2, \dots, n$ and all $\theta = \theta(1), \theta(2), \dots$. Taking the limit, it follows from the continuity of $\varphi^{(\theta)}$ that each transaction i gets the maximum share of resource j_i under allocation φ: the allocation is BMF. It remains to prove the continuity of the function φ in C, which follows from that of $\varphi^{(\theta)}$ in θ and that of θ_1 in C. □

Proposition 4 – Uniqueness of BMF

We first assume that $a_{ij} > 0$ for all i, j and number the transactions in increasing order of a_{i2}/a_{i1}. Observe that $\varphi_1 a_{11} \ge \varphi_2 a_{21} \ge \dots \ge \varphi_n a_{n1}$ (if $\varphi_1 a_{11} < \varphi_2 a_{21}$ for instance then $\varphi_1 a_{12} \le \varphi_1 a_{11} \times a_{22}/a_{21} < \varphi_2 a_{22}$ so that the bottleneck-max property is violated for transaction 1). Similarly, $\varphi_1 a_{12} \le \varphi_2 a_{22} \le \dots \le \varphi_n a_{n2}$. Without loss of generality, we can assume that $n \ge 2$ and

$$\frac{a_{12}}{a_{11}} < 1 < \frac{a_{n2}}{a_{n1}}. \quad (15)$$

Otherwise, one of the two resources is not limiting and the allocation is unique by single-bottleneck fairness.

We consider three cases:

- Case $\frac{1}{n}\sum_{i=1}^{n} a_{i2}/a_{i1} \le 1$. The allocation φ such that $(\varphi_1 a_{11}, \dots, \varphi_n a_{n1}) = (1/n, \dots, 1/n)$ is BMF. Assume there is another BMF allocation, say $\varphi' \ne \varphi$. We have

$$\sum_{i=1}^{n} \varphi_i' a_{i2} = \frac{1}{n}\sum_{i=1}^{n} \frac{a_{i2}}{a_{i1}} + \sum_{i=1}^{n} \frac{a_{i2}}{a_{i1}}(\varphi_i' a_{i1} - \frac{1}{n}). \quad (16)$$

  Since $\varphi' \ne \varphi$, $\varphi_1' a_{11} > 1/n > \varphi_n' a_{n1}$. Let k be the maximum index i such that $\varphi_i' a_{i1} > 1/n$ and define

$$\theta = \sum_{i=1}^{k} (\varphi_i' a_{i1} - \frac{1}{n}) > 0.$$

  Observe that

$$\sum_{i=k+1}^{n} (\frac{1}{n} - \varphi_i' a_{i1}) \ge \sum_{i=1}^{k} (\varphi_i' a_{i1} - \frac{1}{n}) = \theta.$$

  Then

$$\begin{aligned}\sum_{i=1}^{n} \frac{a_{i2}}{a_{i1}}(\varphi_i' a_{i1} - \frac{1}{n}) &= \sum_{i=1}^{k} \frac{a_{i2}}{a_{i1}}(\varphi_i' a_{i1} - \frac{1}{n}) \\ &\quad - \sum_{i=k+1}^{n} \frac{a_{i2}}{a_{i1}}(\frac{1}{n} - \varphi_i' a_{i1}), \\ &\le (\frac{a_{k,2}}{a_{k,1}} - \frac{a_{k+1,2}}{a_{k+1,1}})\theta \le 0,\end{aligned}$$

  where one of the last two inequalities is strict in view of (15). Using (16), we get

$$\sum_{i=1}^{n} \varphi_i' a_{i2} < \frac{1}{n}\sum_{i=1}^{n} \frac{a_{i2}}{a_{i1}} \le 1.$$

  By single-bottleneck fairness, this implies $\varphi' = \varphi$, a contradiction.

- Case $\frac{1}{n}\sum_{i=1}^{n} a_{i1}/a_{i2} \le 1$. We prove similarly that the BMF allocation is unique.

- Case $\frac{1}{n}\sum_{i=1}^{n} a_{i2}/a_{i1} > 1$ and $\frac{1}{n}\sum_{i=1}^{n} a_{i1}/a_{i2} > 1$. By the bottleneck-max property, there exists some index $k < n$ such that $\alpha_1 = \varphi_1 a_{11} = \dots = \varphi_k a_{k1} \ge \varphi_{k+1} a_{k+1,1}$

and $\alpha_2 = \varphi_{k+1}a_{k+1,2} = \ldots = \varphi_n a_{n2} \geq \varphi_k a_{k2}$. Without loss of generality, we can assume that $\alpha_1 > \varphi_{k+1}a_{k+1,1}$ (we take index $k+1$ instead of k otherwise). We have

$$\begin{cases} k\alpha_1 + S_2(n-k)\alpha_2 = 1, \\ S_1(k)\alpha_1 + (n-k)\alpha_2 = 1, \end{cases}$$

where

$$S_1(k) = \sum_{i=1}^{k} \frac{a_{i2}}{a_{i1}} \quad \text{and} \quad S_2(k) = \sum_{i=n-k+1}^{n} \frac{a_{i1}}{a_{i2}}.$$

Now

$$\frac{S_1(k)}{k}\frac{S_2(n-k)}{n-k} \leq \frac{a_{k,2}}{a_{k,1}} \times \frac{a_{k+1,1}}{a_{k+1,2}} \leq 1,$$

where one of the last two inequalities is strict in view of (15). Then $S_1(k)S_2(n-k) < k(n-k)$ and

$$\alpha_1 = \frac{n-k-S_2(n-k)}{k(n-k)-S_1(k)S_2(n-k)},$$

$$\alpha_2 = \frac{k-S_1(k)}{k(n-k)-S_1(k)S_2(n-k)}.$$

In particular, $S_1(k) < k$ and $S_2(n-k) < n-k$.

It remains to prove the uniqueness of the index k. Define $\delta_k = \alpha_2/\alpha_1$. Note that

$$\frac{a_{k,2}}{a_{k,1}} \leq \delta_k < \frac{a_{k+1,2}}{a_{k+1,1}}. \tag{17}$$

Let P_k be the property that $S_1(k) < k$ and $S_2(n-k) < n-k$. In view of (15), there exists some index i_1 such that

$$\frac{a_{i_1,2}}{a_{i_1,1}} \leq 1 < \frac{a_{i_1+1,2}}{a_{i_1+1,1}}.$$

If P_k holds and $k < i_1$ then P_{k+1} holds; similarly if P_k holds and $k > i_1 + 1$ then P_{k-1} holds. Thus the set of indices k such that P_k holds is of the form $\{k_1, \ldots, k_2\}$ with $k_1 \leq k_2$. For all $k = k_1 + 1, \ldots, k_2$,

$$\begin{aligned} \delta_k \geq \frac{a_{k2}}{a_{k1}}, & \\ \iff \quad & k - S_1(k) \geq \frac{a_{k2}}{a_{k1}}(n-k-S_2(n-k)), \\ \iff \quad & k - S_1(k-1) \geq \frac{a_{k2}}{a_{k1}}(n-k+1-S_2(n-k)), \\ \iff \quad & k-1-S_1(k-1) \geq \\ & \frac{a_{k2}}{a_{k1}}(n-k+1-S_2(n-k+1)), \\ \iff \quad & \delta_{k-1} \geq \frac{a_{k2}}{a_{k1}}, \end{aligned}$$

where we have used the fact that $S_1(k) = S_1(k-1) + a_{k2}/a_{k1}$ and $S_2(n-k) = S_2(n-k+1) - a_{k1}/a_{k2}$. Now assume that k satisfies (17). If $k > k_1$ then $\delta_k \geq a_{k2}/a_{k1}$ implies $\delta_{k-1} \geq a_{k2}/a_{k1} \geq a_{k-1,2}/a_{k-1,1}$ and, by induction, $\delta_l \geq a_{l+1,2}/a_{l+1,1}$ for all $l = k_1, \ldots, k-1$. Similarly, if $k < k_2$ then $\delta_k < a_{k+1,2}/a_{k+1,1}$ implies $\delta_{k+1} < a_{k+1,2}/a_{k+1,1} \leq a_{k+2,2}/a_{k+2,1}$ and, by induction, $\delta_l < a_{l,2}/a_{l,1}$ for all $l = k+1, \ldots, k_2$. We conclude that there is at most one index k such that P_k holds and (17) is satisfied.

We now relax the assumption that $a_{ij} > 0$ for all i, j. We still number the transactions in increasing order of a_{i2}/a_{i1}, with $a_{i2}/a_{i1} = +\infty$ if $a_{i1} = 0$. Again, we consider three cases:

- Case $\frac{1}{n}\sum_{i=1}^{n} a_{i2}/a_{i1} \leq 1$. This implies that $a_{i1} > 0$ for all $i = 1, \ldots, n$ and we prove in the same way that the allocation φ such that $(\varphi_1 a_{11}, \ldots, \varphi_n a_{n1}) = (1/n, \ldots, 1/n)$ is the unique BMF allocation.
- Case $\frac{1}{n}\sum_{i=1}^{n} a_{i1}/a_{i2} \leq 1$. This implies that $a_{i2} > 0$ for all $i = 1, \ldots, n$ and the allocation φ such that $(\varphi_1 a_{12}, \ldots, \varphi_n a_{n2}) = (1/n, \ldots, 1/n)$ is the unique BMF allocation.
- Case $\frac{1}{n}\sum_{i=1}^{n} a_{i2}/a_{i1} > 1$ and $\frac{1}{n}\sum_{i=1}^{n} a_{i1}/a_{i2} > 1$. By the bottleneck-max property, there exists some index $k < n$ such that $\alpha_1 = \varphi_1 a_{11} = \ldots = \varphi_k a_{k1} \geq \varphi_{k+1}a_{k+1,1}$ and $\alpha_2 = \varphi_{k+1}a_{k+1,2} = \ldots = \varphi_n a_{n2} \geq \varphi_k a_{k2}$. Observe in particular that, by the sharing-incentive property, $a_{i1} > 0$ for all $i \leq k$ and $a_{i2} > 0$ for all $i > k$. The rest of the proof is identical.

Theorem 1 – Stability of BMF

Assume that $\max(a_{k1}, a_{k2}) = 1$ for all k; this assumption is not restrictive by scale invariance. We know by the proof of Proposition 4 that resource 1 is the single bottleneck if and only if $a_{k1} > 0$ for all k and $\sum_{k=1}^{K} n_k \frac{a_{k2}}{a_{k1}} < n$, which implies $\vec{n}.a_2 < \vec{n}.a_1$ since

$$\begin{aligned} \vec{n}.(a_2 - a_1) &= \sum_{k=1}^{K} n_k \frac{a_{k2}-a_{k1}}{a_{k1}} + \sum_{k=1}^{K} n_k \frac{a_{k2}-a_{k1}}{a_{k1}}(a_{k1}-1), \\ &\leq \left(\sum_{k=1}^{K} n_k \frac{a_{k2}}{a_{k1}}\right) - n < 0. \end{aligned}$$

In particular, $\vec{n}.a_2 \geq \vec{n}a_1$ implies that resource 2 is a bottleneck; similarly, $\vec{n}.a_1 \geq \vec{n}a_2$ implies that resource 1 is a bottleneck.

Let V be the function defined by $V(x) = \max(x.a_1, x.a2)$ for all $x \in \mathbb{R}_+^K$. Observe that the sets $\{\vec{n} : V(\vec{n}) \leq m\}$ are finite for any $m > 0$ so that V is a Lyapunov function for the Markov process $\vec{n}(t)$. The drift is given by

$$\begin{aligned} \Delta V(\vec{n}) = &\sum_{k=1}^{K} \lambda_k (V(\vec{n}+e_k) - V(\vec{n})) \\ &+ \sum_{k:n_k>0} \mu\phi_k(\vec{n})(V(\vec{n}-e_k) - V(\vec{n})), \end{aligned}$$

where e_k denotes the unit vector on component k and $\mu = \mu_1 = \ldots = \mu_k$. For any state $\vec{n}$ such that $\vec{n}.(a_1 - a_2) \geq 0$ and $(\vec{n} \pm e_k).(a_1 - a_2) \geq 0$ for all k, resource 1 is a bottleneck both in state $\vec{n}$ and after any jump from state $\vec{n}$. In particular, $V(\vec{n}) = \vec{n}.a_1$ and

$$\Delta V(\vec{n}) = \mu\left(\sum_{k=1}^{K} \rho_k a_{k1} - 1\right) \leq -\mu\delta,$$

where $\delta = 1 - \max_{j=1,2}\sum_{k=1}^{K} \rho_k a_{kj}$ is a positive constant.

Similarly, in any state $\vec{n}$ such that $\vec{n}.(a_2 - a_1) \geq 0$ and $(\vec{n} \pm e_k).(a_2 - a_1) \geq 0$ for all k, resource 2 is a bottleneck both in state $\vec{n}$ and after any jump from state $\vec{n}$ and $\Delta V(\vec{n}) \leq -\mu\delta$. Since V is a continuous, piecewise linear function and the transition rates of the Markov process $\vec{n}(t)$ are bounded, there exists some smoothened version W of V, which is a Lyapunov function for $\vec{n}(t)$ and such that $\Delta W(\vec{n}) \leq -\mu\delta/2$ outside a compact set [3]. By Foster's criterion, the Markov process $\vec{n}(t)$ is ergodic.