The price of anarchy on uniformly related machines revisited*

Leah Epstein† Rob van Stee‡

January 10, 2012

Abstract

Recent interest in Nash equilibria led to a study of the *price of anarchy* (POA) and the *strong price of anarchy* (SPOA) for scheduling problems. The two measures express the worst case ratio between the cost of an equilibrium (a pure Nash equilibrium, and a strong equilibrium, respectively) to the cost of a social optimum.

The atomic players are the jobs, and the delay of a job is the completion time of the machine running it, also called the load of this machine. The social goal is to minimize the maximum delay of any job, while the selfish goal of each job is to minimize its own delay, that is, the delay of the machine running it.

We consider scheduling on uniformly related machines. While previous studies either consider identical speed machines or an arbitrary number of speeds, focusing on the number of machines as a parameter, we consider the situation in which the number of different speeds is small. We reveal a linear dependence between the number of speeds and the POA. For a set of machines of at most p speeds, the POA turns out to be exactly $p + 1$. The growth of the POA for large numbers of related machines is therefore a direct result of the large number of potential speeds. We further consider a well known structure of processors, where all machines are of the same speed except for one possibly faster machine. We investigate the POA as a function of both the speed ratio between the fastest machine and the number of slow machines.

1 Introduction

Many "solution concepts" are used to study the behavior of selfish agents in non-cooperative games. A *Nash equilibrium* [23] is a state in non-cooperative games which is stable in the sense that no agent can gain from unilaterally switching strategies. A *strong equilibrium* is a pure Nash equilibrium, in which not only single players cannot benefit from changing their strategy (to a different pure strategy), but no non-empty subset of players can form a coalition, where a coalition means that all of them can change their strategies together, and all gain from the change (see [2, 1, 6]).

Following recent interest of computer scientists in game theory [24, 17, 18, 27], we study pure Nash equilibria and strong equilibria for a scheduling problem on uniformly related machines. This is a basic assignment problem. A set of jobs $\mathcal{J} = \{j_1, j_2, \ldots, j_n\}$ is to be assigned to a set of m machines $\mathcal{M} = \{M_1, \ldots, M_m\}$, where machine M_i has a speed s_i. The size of job j_k is denoted by w_k and it is equal to its running time on a unit speed machine. Moreover, the running time of this job on a machine of speed s is $\frac{w_k}{s}$. An assignment or schedule is a function $\mathcal{A} : \mathcal{J} \to \mathcal{M}$. The completion time of machine M_i, which is also called the *delay* or *load* of this machine, is $\sum_{k:\mathcal{A}(j_k)=M_i} \frac{w_k}{s_i}$. The cost, or the *social cost* of a schedule is the maximum delay of any machine, i.e., the makespan. We see

*A preliminary version of this paper appeared in the Proceedings of First Symposium on Algorithmic Game Theory (SAGT 2008), LNCS 4997, pages 46–57. Springer, 2008.

†Department of Mathematics, University of Haifa, 31905 Haifa, Israel. lea@math.haifa.ac.il.

‡Max-Planck-Institut für Informatik, Saarbrücken, Germany. vanstee@mpi-inf.mpg.de. Research supported by German Research Foundation (DFG).

jobs as atomic players, thus we use terms such as choice and benefit for these players. As in previous work [1, 5, 7, 8, 9, 10, 12, 17, 22], the *delay* of a player is the load that it experiences, i.e., the load of the machine on which this job is running. This is known as the *makespan* mechanism. Other so-called *coordination mechanisms* have also been studied [15].

A schedule is a Nash equilibrium if there exists no job that can decrease its delay by migrating to a different machine unilaterally. Formally, consider an assignment $\mathcal{A} : \mathcal{J} \rightarrow \mathcal{M}$. The class of schedules $\mathcal{C}$ contains all schedules $\mathcal{A}'$ that differ from $\mathcal{A}$ only in the assignment of a single job. That is, $\mathcal{A}' \in \mathcal{C}$ if there exists a job $j_k \in \mathcal{J}$ such that $\mathcal{A}'(j_\ell) = \mathcal{A}(j_\ell)$ for all $J_\ell \in \mathcal{J}$, $J_\ell \neq J_k$, and $\mathcal{A}'(j_k) \neq \mathcal{A}(j_k)$. We say that $\mathcal{A}$ is a (pure) Nash equilibrium if for any job j_k, the delay of j_k in any schedule $\mathcal{A}' \in \mathcal{C}$, for which $\mathcal{A}'(j_k) \neq \mathcal{A}(j_k)$, is no smaller than its delay in $\mathcal{A}$. Pure Nash equilibria do not necessary exist for all games (as opposed to mixed Nash equilibria). It is known that for the scheduling game we consider, a pure Nash equilibrium always exists [12, 8].

A schedule is a strong equilibrium if there exists no (non-empty) subset of jobs, such that if all jobs in this set migrate to different machines of their choice simultaneously, this results in a smaller delay for each and every one of them. Formally, given a schedule $\mathcal{A}$, we can define a class of schedules $\tilde{\mathcal{C}}$ which contains all sets of schedules $\mathcal{C}_K$, where $K \subseteq \mathcal{J}$, $K \neq \emptyset$. For any $\mathcal{A}' \in \mathcal{C}_K$, and $\ell \notin K$, we have $\mathcal{A}'(j_\ell) = \mathcal{A}(j_\ell)$ whereas for $\ell \in K$, we have $\mathcal{A}'(j_\ell) \neq \mathcal{A}(j_\ell)$. $\mathcal{A}$ is a strong equilibrium if for any $K \neq \emptyset$, and any $\mathcal{A}' \in \mathcal{C}_K$, there exists at least of job $j_k \in K$ whose delay in $\mathcal{C}_K$ is no smaller than its delay in $\mathcal{A}$. A strong equilibrium is always a pure Nash equilibrium (by definition). Again, strong equilibria do not necessarily exist. Andelman, Feldman and Mansour [1] were the first to study strong equilibria in the context of scheduling and proved that scheduling games (of a more general form) admit strong equilibria. More general classes of congestion games which admit strong equilibria were studied in [14, 29].

In our scheduling game, the *coordination ratio*, or *price of anarchy (*POA*)* (see [26]) is the worst case ratio between the cost of a pure Nash equilibrium and the cost (i.e., maximum delay or makespan) of an optimal schedule. In this paper, we will use OPT to denote a specific optimal schedule (chosen arbitrarily or in some other way). The *strong price of anarchy (*SPOA*)* is defined similarly, but only strong equilibria are considered. Therefore we refer to the pure price of anarchy by POA and when we discuss the mixed price of anarchy we call it the mixed POA. Note that a pure equilibrium is a special case of mixed equilibria. It is noted in a series of papers (e.g., [17, 22, 25, 5, 4]) that the model which we study is a simplification of problems arising in real networks. This is also where our definition of the delay of a job comes from.

Previous work A number of papers studied equilibria for scheduling on uniformly related machines [17, 22, 5, 9, 10]. Czumaj and Vöcking [5] showed that the POA is $\Theta(\frac{\log m}{\log \log m})$ (and $\Theta(\frac{\log m}{\log \log \log m})$ for mixed strategies). Feldmann et al. [9] proved that the POA for $m = 2$ and $m = 3$ is $\frac{\sqrt{4m-3}+1}{2}$ which equals $\phi = \frac{\sqrt{5}+1}{2}$ for two machines and 2 for three machines. In [7], the exact POA and SPOA for two machines is found as a function of the machine speeds. The two measures give different results for the interval $(\phi, 2.247)$ of speed ratios between the two machines, and identical results otherwise. As for the mixed POA, it was shown in [17] that it is at least $1 + \frac{s}{s+1}$ for $s \leq \phi$, where the speeds of the two machines are denoted by 1 and $s > 1$. Fiat et al. [10] showed that the SPOA for this model is $\Theta(\frac{\log m}{(\log \log m)^2})$.

For m identical machines (i.e., the case where all speed are equal), the POA is $\frac{2m}{m+1}$, which can be deduced from the results of [11] (the upper bound) and [28] (the lower bound). It was shown in [1] that the SPOA has the same value as the POA for every m. Note, however, that the mixed POA is non constant already in this case, and equals $\Theta(\frac{\log m}{\log \log m})$, where the lower bound was shown by Koutsoupias and Papadimitriou [17] and the upper bound by Chumaj and Vöcking [5] and independently by Koutsoupias, Mavronicolas and Spirakis [16]. Tight bounds of $\frac{3}{2}$ on the mixed POA for two identical machines were

shown in [17].

Our results It can be seen that the POA and SPOA were studied mainly as a function of the number of machines. Another relevant parameter for uniformly related machines is the number of different speeds. A natural question is whether the POA and SPOA grow as the number of machines increases even if the number of different speeds is constant, or whether it is actually the number of speeds that needs to increase. Previous results, and in particular, the POA for identical machines, already hint that the second option is the right one. We prove this property formally, specifically, in Section 2 we show that the POA for inputs with at most p different speeds is exactly $p+1$. We note that it can be deduced from [10] that the SPOA for inputs with at most p different speeds is $\Omega(\frac{p}{\log p})$ (and $O(p)$ by our result), therefore the SPOA is quite close to the POA and it is it influenced by the number of different speeds as well.

We further focus on a well known architecture of machines, which consists of a single "fast machine" of speed $s \geq 1$ together with $m-1$ unit speed machines. Such a structure, where one processor is fast, and all others are identical, is natural, and was studied in [21, 13, 3, 20, 19]. In Section 3, we give a complete analysis of the exact POA as a function of the speed of the faster machine, s, and the number of identical machines, $M = m-1$. We believe that our comprehensive analysis would contribute to a deeper understanding of the POA as a function of several parameters, rather than as a function of the number of machines as a single parameter. Our results imply that the worst case POA (the supremum POA over all values of s and m) for this special case of two different speeds is already 3. We conclude the paper by showing in Section 4 that the worst case SPOA for this variant is strictly smaller than the POA, already in this special case, but it is still strictly larger than the SPOA for m identical machines.

2 A tight bound on the POA for p speeds

In this section, we consider the general case of a machine set with a fixed number of different speeds, and show that the POA is linearly dependent on the number of speeds, namely, it is $p+1$ if there are p different speeds. We use ingredients of the proofs in [5], focusing on the load in different groups of machines. We assume that $p > 1$, since $p = 1$ is simply the case of identical machines, for which a tight bound is known [11, 28, 1].

Lemma 1 *The price of anarchy on m related machines that have exactly p different speeds is at most $p+1$.*

Proof Consider a job assignment to machines, denoted by $\mathcal{A}$, that satisfies the conditions of a Nash equilibrium. Let $\sigma_1 > \cdots > \sigma_p$ be a sorted list of the speeds. We define the speed class ℓ as the subset of machines with speed σ_ℓ. We assume that machines are numbered by $1, \ldots, m$, and their speeds $s_1, \ldots, s_m$ are sorted by non-increasing speed (i.e., $s_1 \geq s_2 \geq \ldots \geq s_m$). Moreover, we assume that the machines of each speed class are sorted by non-increasing load in $\mathcal{A}$. Let T be the maximum load over all machines and scale the instance so that OPT $= 1$. Assume $T > 1$, otherwise we are done. Note that since some machine has a load that exceeds 1, then there must exist at least one machine whose load is strictly smaller than 1.

Let C be the load of the least loaded machine of speed class 1, by the order defined above, that is, a machine r of speed $s_r = \sigma_1$ such that $s_{r+1} = \sigma_2$. We claim that $C \geq T - 1$. If the maximum load is achieved on this machine, then we have $C = T$ and we are done. Otherwise, let k be a machine of load T. For a given job j of the instance, OPT (which has makespan 1) runs j on one of the machines, which we denote by i_j. Therefore we have that its size satisfies $w_j \leq s_{i_j} \leq \sigma_1$ and thus $\frac{w_j}{\sigma_1} \leq 1$. Since moving a job from machine k to machine r is not beneficial, for such a job we have $T \leq C + \frac{w_j}{\sigma_1} \leq C + 1$. This proves the claim. If $C \leq 1$ then $T \leq 2 < p+1$. Therefore we assume $C > 1$.

We introduce additional notations. Let $C' = \lceil C \rceil \geq 2$. We define $J_1, \ldots, J_{C'-1}$ and $I_1, \ldots, I_{C'-1}$ which are indices of machines. We let I_i be the first machine (in the sorted order above) with a load

which is strictly smaller than $C' - i$, and $J_i = I_i - 1$. We show that all I_i's are well defined and the values J_i are actual indices of machines (i.e., $J_i \geq 1$ for $i \geq 1$). Since machine r has load C and by definition $C' < C + 1$, we have that machine r has load $C > C' - 1$. By the ordering of machines, machines $1, \ldots, r-1$ have a load of at least $C' - 1$ as well. By the definition of the indices I_i, we have $I_1 \geq r + 1$ and thus $J_1 \geq r \geq 1$. Moreover, $I_i \geq I_{i-1}$ for all $2 \leq i \leq C' - 1$, thus $J_i \geq 1$ for all $i \geq 1$, as claimed. Since $I_{C'-1}$ is the first machine with a load smaller than $C' + 1 - C' = 1$, this last index must exist, since some machine must have load less than 1. Note that $I_{C'}$ is not defined and cannot exist since this would imply a machine of load less than 0.

By definition, the load of machines $1, \ldots, J_i$ is at least $C' - i$. We now claim that the speed of I_i is no larger than σ_{i+1} for $i = 1, \ldots, C' - 1$. We prove this by induction. For $i = 1$ we showed that $I_1 \geq r + 1$, so the speed of I_1 is at most σ_2. For other values of i, we prove that the speed of I_i is strictly smaller than the speed of I_{i-1}. Let s' be the speed of I_{i-1}. All machines up to J_{i-1} have load of at least $C' - (i-1) = C' + 1 - i > 1$ since $i \leq C' - 1$. Recall that $I_i \geq r + 1$ for $i \geq 1$. We showed that in $\mathcal{A}$, machines $1, \ldots, J_{i-1}$ are loaded by more than 1. Thus in this schedule they must have a job that OPT schedules on one of the machines $I_{i-1}, \ldots, m$. Denote such a job and its size by a. The machine that runs it in $\mathcal{A}$ has load of at least $C' + 1 - i$. Let y be the machine to which a is assigned in OPT. We have $a \leq s_y \leq s'$ and $J_{i-1} < I_{i-1} \leq I_i$. If the speed of machine I_i is s' as well, moving job a to I_i will result in load of less than $(C' - i) + 1$, which would be a contradiction to $\mathcal{A}$ being a Nash equilibrium, since the load of the machine running a in $\mathcal{A}$ is larger.

From this claim it follows that the speed of $I_{C'-1}$ is at most $\sigma_{C'}$, i.e., $C' \leq p$ (since σ_p is the smallest speed). We conclude that $T \leq C + 1 \leq C' + 1 \leq p + 1$. □

Lemma 2 *The price of anarchy on m related machines that have exactly p different speeds is at least $p + 1$.*

Proof Let $\varepsilon > 0$ be such that $1/\varepsilon$ is an even integer. We consider a set of machines with speeds in the set $\{2^{p-1}, 2^{p-2}, \ldots, 1\}$ for some integer $p \geq 2$. There are N_i machines of speed 2^i, where N_i will be determined later. In OPT, each machine of speed 2^i has a job of size $(1 - \varepsilon)2^i$, for $i \geq 1$. $4N_1$ of the machines of speed 1 have a single job of size $1 - \varepsilon$ and the rest have sand (throughout the paper, we use the common term *sand* to describe arbitrarily small jobs) of total size 1. We will define N_0 to be large enough to ensure $N_0 > 4N_1$. Therefore OPT $= 1$.

In the Nash equilibrium that we define, there is one machine of speed 2^{p-1} which contains $p + 1$ jobs of size $(1 - \varepsilon)2^{p-1}$. We let $N_{p-1} = p + 1$. Each one of the other machines of speed 2^{p-1} contains $2p$ jobs of size $(1 - \varepsilon)2^{p-2}$. We let $N_{p-2} = 2p(N_{p-1} - 1) = 2p^2$. For $1 \leq i \leq p - 2$, each machine of speed 2^i in the Nash equilibrium contains $2(i + 1)$ jobs of size $(1 - \varepsilon)2^{i-1}$. Therefore, for these values of i (except for $i = 1$), $N_{i-1} = 2(i + 1)N_i$. We let $N_0 = 4N_1/\varepsilon$. Thus if in the Nash equilibrium, each machine of speed 1 has a total of $1 - \varepsilon$ of sand, and in OPT, each machine except $4N_1$ machines have a total of 1 of sand, we get that the amount of sand is constant; $4N_1/\varepsilon - 4N_1 = (1 - \varepsilon)4N_1/\varepsilon$.

Moreover, the load of a machine of speed 2^i is $(1 - \varepsilon)(i + 1)$, except for one machine of speed 2^{p-1} which has a load of $(1 - \varepsilon)(p + 1)$.

To show that this is indeed a Nash equilibrium. We do not need to consider cases in which jobs move to faster machines, since they are more loaded. We first consider the case where a job of size $(1-\varepsilon)2^{p-1}$ moves from the machine of speed 2^{p-1} that contains all jobs of this size, to a machine of some speed 2^j ($j \leq p - 1$). It increases the load of the target machine by $(1 - \varepsilon)2^{p-1-j}$. The load of this machine was $(1 - \varepsilon)(j + 1)$, so we need to show $(1 - \varepsilon)(j + 1 + 2^{p-1-j}) \geq (1 - \varepsilon)(p + 1)$ or equivalently $2^{p-1-j} \geq p - j$. It is enough to show $2^{t-1} \geq t$ for $t \geq 1$. This is easily shown by induction.

We now consider a job of size $(1 - \varepsilon)2^i$ moving from a machine of speed 2^{i+1} to a machine of speed 2^j, where $j \leq i$. The load of the target machine increases by $(1 - \varepsilon)2^{i-j}$. The load there was $(1 - \varepsilon)(j + 1)$ so we need to show $2^{i-j} + j + 1 \geq i + 2$ for $i - j \geq 0$. Taking $t = i - j + 1$, we get $2^{t-1} \geq t$ as before. □

The previous two lemmas together imply the following Theorem.

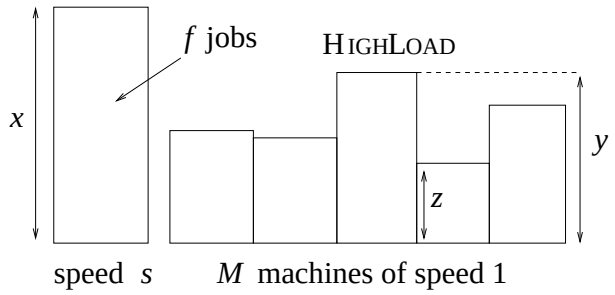

Figure 1: Definitions for Section 3, the structure of the Nash equilibrium which is examined.

Theorem 1 *The price of anarchy on m related machines that have exactly p different speeds is exactly $p + 1$.*

Note that the SPOA increases rapidly as a function of the number of speeds as well. The lower bound construction of Fiat et al. [10] uses a parameter ℓ, such that the SPOA is $\Omega(\ell)$ and the number of speeds is $\Theta(\ell \log \ell)$. This implies a lower bound of $\Omega(\frac{p}{\log p})$ on the SPOA for instances with at most p different speeds.

3 One fast machine: the POA

Recall that the configuration of processors that we consider here consists of $M = m - 1$ identical slow machines of speed 1, and one fast machine of speed s, where $M \geq 2$. Note that the case $M = 1$ is fully covered in [7], for which case the POA is equal to $1 + \frac{s}{s+2}$ for $1 \leq s \leq \sqrt{2} \approx 1.4142$, it is equal to s for $\sqrt{2} \leq s \leq \frac{1+\sqrt{5}}{2} \approx 1.618$ and to $1 + \frac{1}{s}$ otherwise.

We scale all sizes of jobs in the instances which we consider so that OPT $= 1$. We can therefore assume that the sum of jobs sizes is at most $s + M$. Moreover, in an optimal schedule, all slow machines contain only jobs that are no larger than 1, and the largest job of any instance is no larger than s.

We assume that we are given a specific schedule $\mathcal{A}$ which is a pure Nash equilibrium and study its properties. The price of anarchy is determined by the highest possible load of any machine. Obviously, if there is a machine with load above 1, there must also be a machine with load less than 1. To prove upper bounds we consider two basic cases; the price of anarchy is either determined by the fast machine, or by some other machine. In this schedule, denote the load on the fast machine by x, and the number of jobs there by f. Additionally, let y be the highest load of any slow machine, let HIGHLOAD be a slow machine with this load, and let z be the smallest load of any slow machine. See Figure 1.

We will give a closed formula for the POA for all possible combinations of s and M. We begin with the case $1 \leq s \leq 2$ in Section 3.1. An upper bound for x is given in Lemma 5, and upper bounds for y are given in Lemma 6. Together, these bounds give a closed formula for the POA for $1 \leq s \leq 2$ in Theorem 2, where we show matching lower bounds for all possible cases.

We then move to the case $s > 2$. In Theorem 3, we show a lower bound instance with a specific load on the fast machine. Given this theorem, the question is whether instances exist with a higher load on a slow machine. To answer this question, in the remaining text we derive upper bounds on y, and structural properties for worst-case equilibria in the case that the $y \geq x$ in Lemmas 7 to 10. Eventually, we will find that there are *always* instances with higher load on a slow machine for $s > 2$.

In Section 3.3, we derive a condition (Equation (5)) under which the POA is equal to a global upper bound for it, denoted by GLOBMAX. We show that for sufficiently large s, condition (5) is always satisfied, where the threshold value for s depends on M. In particular, (5) holds for $s \geq 4.562$ and any M (Theorem 6). In Section 3.4, we determine the POA in the cases where (5) does not hold. We first show an upper bound for y (Lemma 14) which depends on f (the number of jobs on the fast machine).

We then define a value f^* in Definition 2 and show that the POA is maximized for $f = f^*$ or $f = f^* - 1$. We deal separately with the case $f^* = M + 1$, because if there are $M + 1$ jobs on the fast machine, then we can show that one of them is on the fast machine in the optimal schedule as well (Lemma 14). The results of our analysis are summarized in Theorems 7 and 8.

To conclude, in Section 3.5, we consider the behavior of the POA for $M \to \infty$ and for $s \to \infty$.

3.1 The exact POA for $1 \leq s \leq 2$

Some of the lemmas and observations in this section hold not only for $s \leq 2$, and are used in other sections as well. When this is the case, we state it explicitly (these are Lemma 3, Lemma 6 and Observation 1) . Otherwise we may assume $s \leq 2$.

We begin by defining two values which will upper bound the load x on the fast machine.

$$\begin{aligned} \text{FastMax} &= \frac{2M+s}{M+s} &&= 2 - \frac{s}{M+s} &&= 1 + \frac{M}{M+s} \\ \text{MaxLoad}(f) &= \frac{1+\frac{s}{M}}{1+\frac{s}{M}-\frac{s}{f}} &&= \frac{(M+s)f}{(M+s)f - Ms} &&= 1 + \frac{Ms}{(M+s)f - Ms} \quad \text{for } f \geq s, f \in \mathbb{N} \end{aligned}$$

We motivate these definitions by proving the following lemma. FastMax will turn out to be the maximum possible load on the FAST machine, whereas $\text{MaxLoad}(f)$ is the maximum possible LOAD on the fast machine *given that there are exactly f jobs on it*. $\text{MaxLoad}(f)$ is only defined for $f \geq s$ (because then $(M+s)f - Ms > 0$).

Lemma 3 *If $x > 1$, then $x \leq$ FastMax. If in addition $f \geq sx$, then $x \leq \text{MaxLoad}(f)$. This holds for any $s \geq 1$.*

Proof The average load on the slow machines is at most

$$\frac{s + M - sx}{M} = 1 - (x-1)\frac{s}{M} \,. \tag{1}$$

Since $x > 1$, and the optimal makespan is 1, there exists a job of size at most 1 on the fast machine in $\mathcal{A}$. This job does not reduce its delay by moving to the least loaded slow machine. If it moves, the load on the machine that it moves to becomes at most $2 - (x-1)\frac{s}{M}$, which must be at least x. This implies $x(1 + \frac{s}{M}) \leq 2 + \frac{s}{M}$, and therefore $x \leq$ FastMax.

If $f \geq sx$, then $f > s$. The average size of jobs on the fast machine is sx/f, so among these jobs there is at least one job of size at most sx/f. Since this job does not benefit from moving to the least loaded slow machine, using (1), we find $x \leq 1 - (x-1)\frac{s}{M} + \frac{sx}{f}$ which implies $x(1 + \frac{s}{M} - \frac{s}{f}) \leq 1 + \frac{s}{M}$, and therefore $x \leq \text{MaxLoad}(f)$. □

We now consider the question of when each of these upper bounds on x is tight and when these bounds correspond to the POA (i.e., when is $x \geq y$).

Lemma 4 *If $f \geq s$, we have $\text{MaxLoad}(f) \leq$ FastMax if and only if $\frac{s}{f}$FastMax ≤ 1.*

Proof If $\text{MaxLoad}(f) \leq 2 - \frac{s}{M+s}$, then $\frac{Ms}{(M+s)f - Ms} \leq \frac{M}{M+s}$ and therefore

$$s \leq \frac{(M+s)f - Ms}{M+s} = f - \frac{Ms}{M+s},$$

so $s + \frac{Ms}{M+s} \leq f$ and FastMax $= 1 + \frac{M}{M+s} \leq \frac{f}{s}$, which implies $\frac{s}{f}$FastMax ≤ 1. It can be seen that we have in fact an equivalence as long as $(M+s)f - Ms > 0$ (and $M > 0$, $s > 0$ hold as well). □

Lemma 5 *If $f \geq s$, then $x \leq \min(\text{FastMax}, \text{MaxLoad}(f))$.*

Proof We assume $x > 1$, otherwise the claim follows from $\text{MAXLOAD}(f) > 1$ for $f \geq s$ and $\text{FASTMAX} > 1$. The first term is an upper bound by Lemma 3. If $\text{MAXLOAD}(f) \leq \text{FASTMAX}$ (and $f \geq s$), then we have $\frac{s}{f}\text{FASTMAX} \leq 1$ by Lemma 4, so $f \geq s \cdot \text{FASTMAX} \geq sx$. The second statement of Lemma 3 then implies that $x \leq \text{MAXLOAD}(f)$. □

Lemma 6 *If there is only one job on* HIGHLOAD*, then* $y \leq s$*. If there are at least two jobs on* HIGHLOAD*, then* $y \leq 2z$ *and*

$$y \leq \frac{2(M+s)}{M+2s}\,.$$

This holds for any $s \geq 1$ *and* $M \geq 2$*.*

Proof The first bound follows as there cannot be a job larger than s if the optimal makespan is 1.

Suppose there are at least two jobs and $y > 2z$. The smallest job on HIGHLOAD has a size of at most $y/2$ and (using $M \geq 2$) it can reduce its delay by moving to a machine with a load of z where the load will be at most $z + y/2 < y$ as a result. Thus this is not an equilibrium, which leads to a contradiction.

Therefore $z \geq y/2$, and each slow machine has a load of at least $y/2$. Since none of the jobs on HIGHLOAD can improve by moving to the fast machine, we find $y \leq x + y/(2s)$ or equivalently $x \geq \frac{2s-1}{2s}y$. Since the total size of jobs is at most $M + s$, this implies $y + (M-1)\frac{y}{2} + \frac{2s-1}{2}y \leq y + (M-1)z + sx \leq M + s$, from which the desired bound follows directly. □

Corollary 1 *If* $s \leq 2$*, then* POA ≤ 2*.*

Proof Follows immediately from Lemma 5 and Lemma 6. In fact, $x \leq \text{FASTMAX} < 2$ and if $y > s$, then $y \leq \frac{2M+2s}{M+2s} < 2$. □

Note: we will show later that in fact $\lim_{M\to\infty} \text{POA} = 2$ for all $s \in [1, 2]$.

Observation 1 $\frac{2(M+s)}{M+2s} < \frac{2M+s}{M+s} = \text{FASTMAX}$ *for all positive* M *and* s*.*

Proof Since the denominators are positive, if it enough to prove $2(M+s)^2 < (2M+s)(M+2s)$, which is equivalent to $sM > 0$. □

Observation 2 $\frac{2(M+s)}{M+2s} < \frac{3(M+s)}{3(M+s)-Ms} = \text{MAXLOAD}(3)$ *for* $\frac{3}{2} < s \leq 2$ *and all* M*.*

Proof For $s \leq 2$, $\text{MAXLOAD}(3)$ is positive since $3 \geq s$. It is enough to prove $6(M+s) - 2Ms < 3M + 6s$, which is equivalent to $s > \frac{3}{2}$. □

Theorem 2 *For* $1 \leq s \leq 2$ *and* $M \geq 2$*, the* POA *is given by the following equation:*

$$\max\left(\min\left(\text{MAXLOAD}(2), \text{FASTMAX}, 1+\frac{1}{s}\right), \min\left(\text{MAXLOAD}(3), \text{FASTMAX}\right), \frac{2(M+s)}{M+2s}, s\right).$$

Proof The four terms represent the following situations in order: $f = 2$, $f \geq 3$, at least two jobs on HIGHLOAD, one job on HIGHLOAD. It is easy to see that this covers all the relevant possibilities: if $f \geq 3$, the second term is an upper bound for x since $\text{MAXLOAD}(f)$ is decreasing in f, so we can apply Lemma 5. On the other hand, if $f = 1$, we have $x \leq 1$ because the largest possible size of any job is s. Since $\text{OPT} = 1$, we have $y \geq x$, and Lemma 6 upper bounds y (last two terms).

We discuss the upper bound for each case below and show that it is tight. In the examples for the lower bound, whenever we want to enforce a specific high load on the fast machine, all other machines will contain sand. In such a case, each machine will receive the same amount of sand. The amount will be set such that the total size of all the jobs is $M + s$, and will be less than 1 per machine. This already ensures that none of these jobs can improve their delay by moving to the fast machine (where the load will be more than 1). Thus we only need to check that the jobs on the fast machine cannot benefit from moving.

The cases which need to be considered are the following.

1. There are two jobs on the fast machine. To prove the upper bound, we note that the first two terms in the minimum are implied by Lemma 5. The last term follows because the total size of any two jobs is at most $s + 1$ if the optimal makespan is 1. We now show matching lower bounds using suitable instances for all three terms in the minimum.

   (a) We deal with the cases where the minimum is $\text{MAXLOAD}(f)$ for $f = 2$ or $f = 3$ together. (The case $f = 3$ is actually case 2(a) below.) We use $\text{MAXLOAD}(f) \leq \text{FASTMAX}$ to show that it is possible to enforce $x = \text{MAXLOAD}(f)$. We have $\frac{s}{f}\text{MAXLOAD}(f) \leq \frac{s}{f}\text{FASTMAX} \leq 1$ by Lemma 4. Therefore, consider the following instance. There are f jobs of size $\text{MAXLOAD}(f) \cdot \frac{s}{f} \leq 1$ which are running on the fast machine. The total amount of sand is $M + s - s \cdot \text{MAXLOAD}(f) \geq 0$ (since $M \geq 2, s \geq 1, s \cdot \text{MAXLOAD}(f) \leq f \leq 3$), which is distributed equally over the slow machines. The optimal makespan is 1 by putting each large job on one machine (we have $M + 1 \geq 3$ machines). This schedule is an equilibrium, since by moving a large job to a slow machine we there get a delay of $\frac{M+s-s\cdot\text{MAXLOAD}(f)}{M} + \frac{s}{f}\text{MAXLOAD}(f) = \text{MAXLOAD}(f)$ which is easily checked.

   (b) If the minimum is FASTMAX, we use $\text{FASTMAX} \leq 1 + \frac{1}{s}$ to enforce $x = \text{FASTMAX}$. By Lemma 4, $\frac{s}{2}\text{FASTMAX} \geq 1$. Consider the following instance. There is one job of size 1 on the fast machine and one job of size $s\text{FASTMAX} - 1 \geq 1$. Each slow machine has sand, where the amount of sand on each slow machine is $(M + s - s\text{FASTMAX})/M = (M + s - \frac{s(2M+s)}{M+s})/M = \frac{M}{M+s}$. This is an equilibrium since already moving the smaller job from the fast machine to a slow machine results in a load of $1 + \frac{M}{M+s} = \text{FASTMAX}$. If $s\text{FASTMAX} - 1 \leq s$, then the optimal makespan is 1, since the fast machine runs this job, one slow machine runs the job of size 1, and the sand is distributed so that the machines are balanced. The condition on the size of the largest job holds since $\text{FASTMAX} \leq 1 + \frac{1}{s}$ in this case.

   (c) In the last case, where the minimum is $1 + \frac{1}{s}$, we show how to enforce $x = 1 + \frac{1}{s}$. In the instance, there is one job of size 1 on the fast machine and one job of size s. Each slow machine has an amount of $(M + s - s - 1)/M = 1 - 1/M$ of sand. This is an equilibrium since using $1 + 1/s \leq \text{FASTMAX}$ in this case, we get $Ms \geq M + s$ and therefore $2 - \frac{1}{M} \geq 1 + \frac{1}{s}$, which means that already the job of size 1 does not benefit from moving to a slow machine. In an optimal assignment, the fast machine runs the job of size s, one slow machine runs the job of size 1, and the sand is spread evenly between the other slow machines.

2. There are at least three jobs on the fast machine. The upper bound follows from Lemma 5 as above. There are two cases depending on the term for which the minimum is achieved.

   (a) See Case 1(a).

   (b) If $\text{FASTMAX} < \text{MAXLOAD}(3)$, we enforce $x = \text{FASTMAX}$. There are two jobs of size 1 on the fast machine and one job of size $s\text{FASTMAX} - 2$. The size of the second job is more than 1 since $sx/3 > 1$ by Lemma 4 if we take $x = \text{FASTMAX}$, and at most s since $\text{FASTMAX} \leq 2 \leq 1 + 2/s$ for $s \leq 2$. In an optimal schedule, for $M \geq 2$ the jobs of size 1 can be assigned to two slow machines, and the larger job to the fast machine. This is an equilibrium (the proof of case 1b, including the calculation of the amount of sand on slow machines, holds here as well).

3. There are at least two jobs on HIGHLOAD. The upper bound on y follows from Lemma 6. Comparing this case to the previous one, by Observations 1 and 2, we have $s \leq 3/2$. We show how to enforce $y = \frac{2(M+s)}{M+2s}$, let y denote this value. Note that this function is monotonically increasing

in M. To prove the lower bound, we consider a schedule with two jobs of size $y/2 < 1$ on one slow machine (HIGHLOAD), each other slow machine has $y/2$ of sand.

If $M \leq 3$, the fast machine has one job of size $\frac{2s-1}{2}y$. For $M \geq 4$, the fast machine has two jobs of size $\frac{2s-1}{4}y \leq 1$. In both cases, its load is $\frac{2s-1}{2s}y \leq \frac{3}{4}y \leq \frac{3}{2}$.

We show that the optimal makespan of this instance is 1 in all cases. Each job that is not part of the sand is put on a separate machine. We will show that there is at most one job that is larger than 1. If there is such a job, it is put on the fast machine. The sand is added to the machines in a balanced way. The jobs have a total size of $\frac{y}{2}(2 + (M-1) + (2s-1)) = M + s$. For $M \leq 3$, it is sufficient to show for $M = 3$ that $\frac{2s-1}{2}y \leq s$. This holds because in this case $\frac{2(M+s)}{M+2s}(s - \frac{1}{2}) = \frac{6+2s}{3+2s}(s - \frac{1}{2}) \leq s \Leftrightarrow (6+2s)(s-\frac{1}{2}) \leq s(3+2s) \Leftrightarrow 5s - 3 \leq 3s \Leftrightarrow s \leq \frac{3}{2}$.

For $M \geq 4$, we find $\frac{2s-1}{4}y \leq \frac{1}{2}y < 1$ for $s \leq 3/2$.

The assignment is an equilibrium: the job(s) on the fast machine can not improve by moving, since already for the case where two jobs are assigned to this machine, $\frac{2s-1}{4}y + y/2 = \frac{2s+1}{4}y \geq \frac{2s-1}{2s}y$ for $s \geq 1$.

Since $\frac{2s-1}{2s}y + \frac{y}{2s} = y$, the jobs on HIGHLOAD do not improve by moving to the fast machine. They also cannot benefit from moving to another slow machine. The sand cannot improve since the load on the fast machine is $\frac{(2s-1)y}{2s} \geq \frac{y}{2}$ for $s \geq 1$.

4. There is one job on HIGHLOAD. The upper bound on y follows from Lemma 6. We show how to enforce a load of $y = s$ on HIGHLOAD using the following schedule. There is one job of size s on a slow machine HIGHLOAD. Let $f = \lceil s(s-1) \rceil$. There are f jobs of size $s(s-1)/f \leq 1$ on the fast machine, so that the load there is $s-1$. The remaining machines have load $\frac{M-s(s-1)}{M-1} = 1 - \frac{s(s-1)-1}{M-1}$, which we denote by z'. This load consists entirely of sand.

   If $z' > s-1$, redistribute the sand among all machines besides HIGHLOAD (i.e. including the fast machine) so that the load on these machines is equal. This is done by moving some sand to the fast machine.

   It is clear that if this redistribution takes place, we have an equilibrium: we only need to check whether jobs can benefit by moving to or from HIGHLOAD. But the job on HIGHLOAD cannot improve since the load on the fast machine is at least $s-1$, and $s - 1 + s/s = s$. No job can improve by moving to HIGHLOAD because HIGHLOAD has the highest load.

   Consider the case where $z' \leq s-1$. Then clearly, none of the sand jobs can improve. A job on the fast machine does not improve if it moves to HIGHLOAD (load is higher). We next consider the option of moving to another slow machine. We need to check that $z' + s(s-1)/f \geq s-1$. Since $1 \leq s \leq 2$, we have $f = 1$ or $f = 2$. For $f = 1$, the claim is obvious. For $f = 2$, it is sufficient to check the case $M = 2$, because z' is increasing in M (since $s^2 > s+1$ in this case). Here we get $1 - (s(s-1) - 1) + s(s-1)/2 \geq s - 1$, which holds for all $s \leq 2$.

This shows that the maximum can indeed be achieved in all four cases, and thus the bounds are tight. □

3.2 The case $x \geq y$ and $s \geq 2$

Here we consider the case where the fast machine has the highest load.

Theorem 3 *For $s \geq 2$, if $y \leq$ FASTMAX then* POA $=$ FASTMAX.

Proof We have $x \leq$ FASTMAX by Lemma 3, so POA $\leq$ FASTMAX in this case. For the lower bound we present an instance where the fast machine has a load of FASTMAX. We place a total size of jobs of

$w = s$FASTMAX on the fast machine. We do this by assigning $\lceil s(1 - s/(M + s)) \rceil \geq 1$ jobs of size 1 to this machine, as well as a job of size

$$q = w - \lceil s(1 - s/(M + s)) \rceil.$$

We have $q = s(2 - s/(M + s)) - \lceil s(1 - s/(M + s)) \rceil \leq s$ and $q \geq s(2 - s/(M + s)) - (s(1 - s/(M + s)) + 1) = s - 1 \geq 1$ by the assumption $s \geq 2$. We get $x =$ FASTMAX, and a total size of sand jobs of $s + M - sx$, thus each slow machine receives $1 - (x - 1)s/M$ of sand. This situation is similar to the one in the proof of Theorem 2 (Case 1b), since the loads of machines are similar, and the smallest job on the fast machine is of size 1, and we already saw that this is an equilibrium.

To show that the optimal makespan is 1, we need to show that $\lceil s(1 - s/(M + s)) \rceil \leq M$ (i.e., we can assign all jobs on the fast machine but one to slow machines) and $q \leq s$ (which we already showed). The first inequality holds because $s(1 - s/(M + s)) = M \cdot \frac{s}{M+s} < M$. □

3.3 The exact POA for sufficiently large values of s

This section is devoted to proving the following global upper bound on the POA:

$$\text{GLOBMAX} = \frac{s + 2M - 1}{s + (M - 1)(s - 1)/s} \,. \tag{2}$$

We derive bounds for GLOBMAX in Lemmas 12 and 13, and use these bounds to determine when POA = GLOBMAX holds in Theorems 5 and 6.

In the next few lemmas, we will first prove that there exist equilibrium instances with $y >$ FASTMAX and several distinct properties. In each case, as soon as we have proved such a statement, we will restrict our attention to instances which have these properties in the remainder of the text.

Lemma 7 *For $s \geq 2$, for any equilibrium instance in which $y >$ FASTMAX, there is one job on* HIGHLOAD *and there exists an instance which is an equilibrium with the same loads on all machines and the same optimal makespan, where all slow machines besides* HIGHLOAD *contain sand.*

Proof If there are at least two jobs on HIGHLOAD, then by Lemma 6 and Observation 1, $y <$ FASTMAX. Let Δ be the total size of jobs assigned to slow machines, excluding HIGHLOAD. Replace these jobs by sand and distribute it evenly. (The same process is applied on these jobs in the optimal solution.) The only case which the resulting schedule is not an equilibrium is the case where the fast machine has a smaller load than the resulting load of the slow machines. In this case, the jobs on it are replaced by sand as well, and the sand is redistributed so that all machines, except for HIGHLOAD, have equal load. Note that the load on the fast machine increases in this last case, so the job of size y does not improve by moving there since it did not do so before. □

Lemma 8 *For any equilibrium instance, there exists an instance which is an equilibrium with the same loads on all machines and the same optimal makespan, such that the fast machine has at most one job which is also on the fast machine in the optimal solution. Specifically, it has at most one job larger than 1.*

Proof If there are multiple such jobs, we can merge them into one job with size the total size of these jobs. This does not affect the optimal makespan, or the makespan of the schedule. Larger jobs can only benefit less from moving, thus the schedule is still an equilibrium if it was before. Regarding the second statement, clearly all jobs larger than 1 must be on the fast machine in an optimal solution with makespan 1. □

Lemma 9 *Any schedule which is an equilibrium and for which $y >$ FASTMAX satisfies*

$$y \leq \frac{sx}{s - 1} \,. \tag{3}$$

Moreover, (3) is a sufficient condition for the job on HIGHLOAD *not to benefit from moving.*

Proof Consider HIGHLOAD. This machine has a single job of size y by Lemma 7, which does not benefit from moving to the fast machine. This holds if and only if $y \leq x + \frac{y}{s}$, which implies (3). □

Lemma 10 *For $s \geq 2$ and $M \geq 2$, if $y >$ FASTMAX, then $f \geq 2$ and there exists an instance which is an equilibrium with the same loads on all machines and the same optimal makespan, in which the smallest job on the fast machine has size at most 1.*

Note: this holds even after possibly merging some jobs as in the proof of Lemma 8.

Proof Suppose there is at most one job on the fast machine. The total size of the jobs on the fast machine and HIGHLOAD (together) is then at most $s + 1$. This means that $sx + y \leq s + 1$. But then Lemma 9 implies $y \leq \frac{s+1-y}{s-1}$, and therefore $y \leq \frac{s+1}{s}$. But this value is smaller than FASTMAX for $s \geq 2$ and $M \geq 2$, a contradiction. To prove $\frac{s+1}{s} \leq$ FASTMAX, we note that $2 - s/(M + s)$ is increasing in M. For $M = 2$, it is equal to $1 + \frac{2}{2+s}$. However $\frac{2}{2+s} \geq \frac{1}{s}$ for $s \geq 2$.

The upper bound on the size of the smallest job on the fast machine now follows from Lemma 8. □

The next lemma relates FASTMAX to GLOBMAX, allowing us to prove a general upper bound for the POA in Theorem 4.

Lemma 11 *We have* FASTMAX $<$ GLOBMAX *for all $s \geq 1$, $M \geq 2$.*

Proof We have FASTMAX $= \frac{2M+s}{M+s}$. The desired inequality is equivalent to

$$\frac{(2M + s)s}{sM + s^2} < \frac{(2M + s - 1)s}{s^2 + (M - 1)(s - 1)} .$$

Simple algebra shows that this holds for all $s \geq 1, M \geq 2$ (actually it holds for $s > 0, M \geq 1$). □

Theorem 4 *For $s \geq 2$,* POA $\leq$ GLOBMAX.

Proof By Lemma 11 and Lemma 3, the claim holds if $y \leq$ FASTMAX. Therefore, suppose $y >$ FASTMAX. Then by Lemma 9, the load on the fast machine is at least $x = y \cdot \frac{s-1}{s}$, so the total size of the jobs there is at least $y(s - 1)$. By Lemma 10, there is at least one job of size at most 1 on the fast machine. Since we are considering an equilibrium, the load on each slow machine must be at least $x - 1$. Finally, the total size of all the jobs must be at most $M + s$. This implies

$$y\left(1 + (s - 1) + (M - 1)\frac{s - 1}{s}\right) - (M - 1) \leq s + M \tag{4}$$

which holds if and only if $y \leq \frac{s+2M-1}{s+(M-1)(s-1)/s} =$ GLOBMAX. This proves the lemma. □

We wish to find out when POA $=$ GLOBMAX holds exactly. To give a condition for this, we first study the function GLOBMAX further in Lemmas 12 and 13.

Lemma 12 *We have* GLOBMAX $\leq (s + M)/s$ *for all $M \geq 2$ and $s \geq 2$.*

Proof A straightforward calculation shows that (for $s > 0$) the inequality holds if $2Ms + M^2 \leq M^2 s + M + s$. This can be shown by induction on M. For $M = 2$, we get $4s+4 \leq 4s+2+s \Leftrightarrow s \geq 2$. For the induction step, we need to show

$$\begin{aligned} &2(M + 1)s + (M + 1)^2 \leq (M + 1)^2 s + M + 1 + s \\ \Leftrightarrow\ &2Ms + M^2 + (2s + 2M + 1) \leq M^2 s + M + s + (2Ms + s + 1) \end{aligned}$$

and indeed $s + 2M \leq 2Ms$ holds, since $(2M - 1)(s - 1) \geq 1$ for $M \geq 2, s \geq 2$. □

Lemma 13 *We have $\frac{s-1}{s}$*GLOBMAX > 1 *for all $M \geq 2$ and $s > 2$.*

Proof We can write the desired inequality as $1/\text{GLOBMAX} + 1/s < 1$. Thus we need to show that

$$\frac{s^2 + (M-1)(s-1)}{s^2 + 2Ms - s} + \frac{1}{s} = \frac{s^2 + (M-1)(s-1) + (s+2M-1)}{s^2 + 2Ms - s} < 1.$$

This holds if and only if $(M-1)(s-1) + s + 2M - 1 = Ms + M < 2Ms - s$, which is true for $s > \frac{M}{M-1}$ and a forteriori for $s > 2$ (using $M \geq 2$). □

Theorem 5 *Let $M \geq 2$ and $s > 2$. Then*

$$(s-1)\text{GLOBMAX} - \lceil s(\text{GLOBMAX} - 1) \rceil \geq 1 \tag{5}$$

implies that POA $=$ GLOBMAX.

Proof First note that by the condition (5), $s(\text{GLOBMAX} - 1) \leq \lceil s(\text{GLOBMAX} - 1) \rceil \leq (s-1)\text{GLOBMAX} - 1$ and so $\text{GLOBMAX} \leq s - 1$.

We present a class of instances where POA $=$ GLOBMAX as long as (5) holds. Note that POA $>$ GLOBMAX is impossible by Lemma 4. Place one job of size $y = \text{GLOBMAX}$ on a slow machine. Set $x = \frac{s-1}{s} \cdot y$. Place $k = \lceil s(\text{GLOBMAX} - 1) \rceil$ jobs of size 1 on the fast machine, as well as a job of size

$$\begin{aligned} q &= (s-1)\text{GLOBMAX} - k \\ &\leq (s-1)\text{GLOBMAX} - s(\text{GLOBMAX} - 1) = s - \text{GLOBMAX}. \end{aligned} \tag{6}$$

Then the total size of the jobs assigned to the fast machine is $(s-1)\text{GLOBMAX} = sx$, as desired.

On each empty slow machine, place $x - 1$ of sand. This is more than 0 by Lemma 13. We now have constructed an equilibrium, which can be verified as follows. Note that since $y > x > x - 1$, we only need to check that no job can improve by moving away from HIGHLOAD or from the fast machine to a slow machine with load $x - 1$. The first part follows from Lemma 9 and the fact that there is only one job on HIGHLOAD (so that job cannot improve by moving to another slow machine), and the second part holds as long as all the jobs on the fast machine (in particular, the job of size q) have size at least 1. This is exactly the condition (5).

We still need to verify that the optimal makespan of this instance is 1. First of all, the total size of all the jobs must be at most $M + s$. This follows because (4) holds for $y = \text{GLOBMAX}$, and our loads are exactly the loads described in Lemma 4. Since $q \leq s - \text{GLOBMAX}$ by (6), the jobs of size q and GLOBMAX can be placed together on the fast machine. Note that $q \geq 1$ since $\text{GLOBMAX} \leq s - 1$. It is now sufficient to show that $k \leq M$. This holds as long as $s(\text{GLOBMAX} - 1) \leq M$, or $\text{GLOBMAX} \leq 1 + M/s$. This is true by Lemma 12. □

Theorem 6 *For $s \geq \frac{5+\sqrt{17}}{2} \approx 4.562$, we have* POA $=$ GLOBMAX.

Proof We give a condition which ensures that (5) holds. Clearly, $(s-1)\text{GLOBMAX} - \lceil s(\text{GLOBMAX} - 1) \rceil \geq (s-1)\text{GLOBMAX} - s(\text{GLOBMAX} - 1) - 1 = s - \text{GLOBMAX} - 1$. Thus, it suffices to have $s - \text{GLOBMAX} - 1 \geq 1$, or $\text{GLOBMAX} \leq s - 2$, in order to ensure (5). GLOBMAX is monotonically increasing in M (we have $\partial\text{GLOBMAX}/\partial M = s(s^2+1)/(s^2 + (M-1)(s-1))^2 > 0$) and tends to $2s/(s-1)$ for $M \to \infty$. We have $2s/(s-1) \leq s - 2$ for $s \geq 4.562$. The result now follows from Theorem 5. □

In the following table, for several values of M the minimum value of s is given, based on (5), such that we can be certain that POA $=$ GLOBMAX for all speeds of at least s, rounded to three decimal places.

M	2	3	4	5	6	7	8
s	2.774	3.246	3.775	3.563	3.409	3.293	3.887

(7)

Using a computer program, it can be found that in fact POA $=$ GLOBMAX for $s \geq 4.365$ for all M, and that the value of M for which the bound on s is maximized is 31. There are also several values of M for which POA $=$ GLOBMAX in non-contiguous intervals. The smallest value of M for which this happens is $M = 14$.

3.4 The POA for intermediate values of s

Theorem 5 gives us a condition under which POA $=$ GLOBMAX. What happens if this condition is not satisfied? We certainly still have the upper bound from Lemma 3 for the case $y \leq$ FASTMAX. In this section, we therefore focus on the case $y >$ FASTMAX > 1. We assume that the modification of Lemma 8 was already applied on the schedule. We give an upper bound for y which depends on the number of jobs f on the fast machine in Lemma 14. This raises the question of which value of f should be selected to get the highest possible value of y. We first define a crucial value f^* in Definition 2, and examine this value in the remainder of Section 3.4.1. Section 3.4.2 then answers the question of how to select f based on f^*. The results are summarized in Theorem 8.

3.4.1 The bound MAXSLOW(f) and the value f^*

Definition 1 *For $f \geq s$, let*

$$\text{MAXSLOW}(f) = \frac{s+M}{s+(M-1)(s-1)(1-s/f)/s}\,.$$

We prove in the following lemma that MAXSLOW(f) is an upper bound for the load on HIGHLOAD (i.e., it is the maximum possible load on a SLOW machine) if $f \geq s$.

Lemma 14 *If $y >$ FASTMAX, and $f \geq s$, then $y \leq$ MAXSLOW(f).*

Proof For an equilibrium, we require $x \geq \frac{s-1}{s} \cdot y$ by Lemma 9. The load on any slow machine must be at least $x - sx/f = x(1-s/f)$, since sx/f is an upper bound on the size of the smallest job on the fast machine if there are f jobs on that machine. This implies

$$y\left(s+(M-1)\frac{s-1}{s}\left(1-\frac{s}{f}\right)\right) \leq s+M \tag{8}$$

which together with $f \geq s$ proves the upper bound (using the total size of the jobs). □

Observation 3 *Let $s \geq 2$, $M \geq 2$. If* MAXSLOW$(f_0) > 0$ *for some $f_0 \in \mathbb{R}^+$, then* MAXSLOW(f) *is continuous, decreasing and positive for all $f \geq f_0$.*

This observation follows because MAXSLOW$(f_0) > 0$ implies that its denominator is positive, and the denominator is strictly increasing in f for all $f > 0$. Given Observation 3, we would like to choose f as small as possible in order to maximize MAXSLOW(f). However, if f is too small, we find that one of the conditions $f < s$ or $f < sx$ will start to hold, in which case Lemma 14 does not give us a useful bound: in the proof, we use sx/f as an upper bound for the size of the smallest job on the fast machine; if $f < sx$, we have the stronger bound of 1. We therefore define the following value, which will give us an initial upper bound for the POA (Lemma 18). Later, we will deal with the cases where there are fewer jobs on the fast machine.

Definition 2 *Let f^* be the minimum value of $f \in \mathbb{N}$ such that $f \geq (s-1)$*MAXSLOW$(f) > 0$.

We will see in the following that f^* indeed exists (see Lemma 15) and that MAXSLOW(f^*) is the highest load on a slow machine that can be achieved using f^* or more jobs on the fast machine, while with fewer jobs on the fast machine, we get smaller bounds. Before we move on to the proofs of these statements, we first prove some useful properties of the value f^* that we will need later. In view of the condition in Lemma 14, we first derive a lower bound for f^*. Solving the equation $f = (s-1)$MAXSLOW(f) for $f \in \mathbb{R}, M \geq 2, s \geq 2$ gives

$$f_1 = \frac{s^2+2(M-1)(s-1)-1}{s^2+(M-1)(s-1)} \cdot s. \tag{9}$$

Lemma 15 *We have $f^* = \lceil f_1 \rceil \geq s$ for any $M \geq 2$ and $s \geq 2$.*

Proof The fraction in the right hand side of (9) is at least 1 for any $M \geq 2$ and $s \geq 2$, therefore $f_1 \geq s$. In particular, this implies $f_1 > 0$, and therefore $\textsc{MaxSlow}(f_1) = f_1/(s-1) > 0$. By Observation 3 we conclude $f^* = \lceil f_1 \rceil \geq s$: for $f \leq \lceil f_1 \rceil - 1$, $f \in \mathbb{N}$, we must have either $\textsc{MaxSlow}(f) \leq 0$ or $f < (s-1)\textsc{MaxSlow}(f)$. □

This lemma shows that for $f \geq f^*$, the second condition in Lemma 14 is always satisfied. We show two additional bounds involving f^*, which will restrict the number of cases that we need to consider.

Lemma 16 *For $s \geq 2$ and $M \geq 2$, we have $f^* - 1 < (s-1)\textsc{MaxSlow}(f^* - 1)$.*

Proof Given the definition of f^* and Observation 3, it is sufficient to show that $\textsc{MaxSlow}(f^*-1) > 0$. This holds if the denominator is positive. Solving $s + (M-1)(s-1)(1-s/f)/s = 0$ for $f \in \mathbb{R}$ gives

$$f_2 = \frac{(M-1)(s-1)}{s^2 + (M-1)(s-1)} \cdot s.$$

For any $f > f_2$, we have $\textsc{MaxSlow}(f) > 0$ and then $\textsc{MaxSlow}(f)$ is continuous, decreasing in f, and positive by Observation 3. Thus if $f^* > f_2 + 1$, we have $\textsc{MaxSlow}(f^* - 1) > 0$ as desired. Given Lemma 15, it is sufficient to show $f_1 > f_2 + 1$. Note that the denominator of f_2 is equal to that of f_1. Thus we need to verify

$$\begin{aligned} &(s^2 + 2(M-1)(s-1) - 1)s > (M-1)(s-1)s + (s^2 + (M-1)(s-1)) \\ \Leftrightarrow\ &(s^2 + (M-1)(s-1) - 1)s > s^2 + (M-1)(s-1) \end{aligned}$$

which holds for $s \geq 2$ and $M \geq 2$. □

Lemma 17 *If (5) does not hold, then $f^* \leq M + 1$ for $s \geq 2$ and $M \geq 2$.*

Proof We have that f_1 is monotonically strictly increasing in s and in M for $s \geq 2$ and $M \geq 2$: we can write it as

$$f_1 = s \cdot \left(1 - \frac{1}{s^2 + (M-1)(s-1)}\right) + \frac{1}{\frac{s}{(M-1)(s-1)} + \frac{1}{s}},$$

from which both assertions follow easily. Therefore $f^* = \lceil f_1 \rceil$ is monotonically increasing in s and M.

If (5) does not hold, then $s \leq 4.56$ by Lemma 4. From (9) it is clear that $f_1 < 2s$ and therefore $f^* \leq \lceil 2s \rceil$. Thus the claim holds for $M + 1 \geq 10$, i.e., $M \geq 9$. For $M = 2, \ldots, 8$, we use the values from Table (7). Thus only the interval $s \in [2, 3.9]$ remains to be checked, so we are done for $M = 8$ and $M = 7$, because $f^* \leq 8$ for $s \leq 4$. For $M = 6$, we know $s < 3.5$, so we are done for that value as well. For $M = 5$ and $s = 4$, we find $f^* = \lceil 39/7 \rceil = 6 = M + 1$, implying that if $M = 5$, we have $f^* \leq 6$ for all s for which (5) does not hold (since then $s \leq 4$, and f_1 and f^* are increasing in s).

For the remaining values, we have the following results. We have

$$s(\textsc{GlobMax} - 1) = \frac{Ms^2 + (M-1)s}{s^2 + (M-1)(s-1)} < M.$$

This value is more than $M - 1$ if $s^2 + (M-1)^2 > s((M-1)^2 - (M-1))$, which holds for $2 \leq M \leq 4$ and all $s \geq 2$. Hence, the values of s such that $s > 2$, for which (5) does not hold, satisfy $(s-1)\textsc{GlobMax} - M < 1$, or equivalently

$$\frac{(s-1)s^2 + (2M-1)(s-1)s)}{s^2 + (M-1)(s-1)} < M + 1. \tag{10}$$

On the other hand, we have

$$f_1 = \frac{s^3 - s + 2s(M-1)(s-1)}{s^2 + (M-1)(s-1)}.$$

We know that as long as $f_1 \leq M + 1$, we also have $f^* \leq M + 1$. But $f_1 \leq M + 1$ follows directly from (10). □

3.4.2 On the value of f which maximizes y

This section deals with the question: how should we select f, i.e. how many jobs should there be on the fast machine in order to get the highest possible value of y (recall that y is scheduled on a slow machine). Lemma 18 deals with the case where there are at least $f^* \leq M$ jobs on the fast machine, and shows that the worst case (highest value for y) is if there are exactly f^* jobs. Lemma 22 deals with the case where there are less than f^* jobs on the fast machine, and shows that the worst case is if there are exactly $f^* - 1$ jobs.

Finally, Lemma 26 (in general, the text below (13)) deals with the case where $f^* = M + 1$, which requires separate attention, and gives a new upper bound for y for this case.

Lemma 18 *If $f^* \leq M$, then there is an equilibrium instance with f^* jobs on the fast machine and $y = \min(s, \text{MAXSLOW}(f^*))$. If $y > 1$, then we have $y \leq \min(s, \text{MAXSLOW}(f^*))$ for **all** equilibria with at least f^* jobs on the fast machine.*

Proof To show existence, let $y = \min(s, \text{MAXSLOW}(f^*))$. Place a job of size y on a slow machine, f^* jobs of size $y(s-1)/f^* \leq 1$ on the fast machine and $z = y\frac{s-1}{s}(1 - s/f^*) \geq 0$ of sand on each empty slow machine. The claimed inequalities in the previous line follow from the definition of f^* and the fact that $f^* \geq s$ (Lemma 15). Then this is an equilibrium with POA $= y$. The job of size y does not benefit from moving by Lemma 9, and no job on the fast machine benefits from moving because $z + y(s-1)/f^* = y(s-1)/s$. Since $y > y(s-1)/s > y\frac{s-1}{s}(1 - s/f^*)$, the sand also does not benefit from moving. The total size of all the jobs is at most $M + s$ since $y \leq \text{MAXSLOW}(f^*)$ so that y satisfies (8). Since we also have $y \leq s$, this shows that the optimal makespan is 1 as long as $f^* \leq M$, because we can then assign each job which is on the fast machine to its own slow machine in the optimal solution, and the job of size y to the fast machine.

With exactly f^* jobs on the fast machine, the second claim follows from Lemma 6 if $y = s \leq \text{MAXSLOW}(f^*)$. Else, we can use Lemma 14. With more than f^* jobs on the fast machine, we use additionally that $\text{MAXSLOW}(f)$ is decreasing in f (Observation 3). □

Lemma 19 *If (5) does not hold, there is an equilibrium instance with $f^* - 1$ equal-sized jobs on the fast machine and $y = \min(s, (f^* - 1)/(s - 1))$. **Any** equilibrium instance with at most $f^* - 1$ jobs on the fast machine, where all those jobs have size at most 1, has $y \leq \min(s, (f^* - 1)/(s - 1))$.*

Proof To prove the first claim, we use an instance analogous to the one from the proof of Lemma 18. There is a job of size $y = \min(s, (f^* - 1)/(s - 1))$ on one slow machine, and $f^* - 1$ jobs of size $y(s-1)/(f^* - 1) \leq 1$ on the fast machine, where the inequality follows from Lemma 16. Each slow machine has an equal amount of sand $z = \max(0, y\frac{s-1}{s}(1 - s/(f^* - 1))$. If $x < z$, we redistribute the sand among the fast machine and the slow machines excluding HIGHLOAD so that all loads are equal (without changing the total size of all the jobs).

Then as in the previous proof, this is an equilibrium with POA $= y$. (If we redistributed some sand because $x < z$, the proof is even easier.) We still need to show that the optimal makespan is 1. Note that $f^* - 1 \leq M$ by Lemma 17, so that in the optimal schedule, we can assign each job which is on the fast machine to its own slow machine, and $y \leq s$ to the fast machine as before. It remains to be shown that the total size of all the jobs is at most $M + s$. If $z > 0$, this follows since $y \leq \text{MAXSLOW}(f^* - 1)$ by Lemma 16 so that y satisfies (8). If $z = 0$, this follows because $y \leq s$ and there are $f^* - 1 \leq M$ jobs of size at most 1.

For the second claim, note that if all jobs on the fast machine have size at most 1, their total size (which is sx) is at most $f^* - 1$ in this case. The claim then follows from Lemma 6, Lemma 9 and Lemma 16. □

Given Lemmas 18 and 19, the only option that we did not yet consider for $f^* \leq M$ is to have at most $f^* - 1$ jobs on the fast machine, where one of the jobs is larger than 1. We will consider the case where $f^* = M + 1$ separately later.

Lemma 20 *If there is a job which is larger than 1 on the fast machine, and $y >$ FASTMAX, then $y \leq 1 + \frac{f-1}{s}$.*

Proof The total size of the f jobs on the fast machine and the single job (Lemma 7) on HIGHLOAD is at most $s + f - 1$. This holds because all but one job (the one larger than 1) on the fast machine are on a slow machine in OPT by Lemma 8 and hence have size at most 1. Moreover, the job larger than 1 on the fast machine, together with the job of size $y > 1$ on HIGHLOAD have a total size of at most s. In other words, $sx + y \leq s + f - 1$, implying that $sx \leq s + f - 1 - y$, and with the help of Lemma 9 we then find that $y \leq \frac{s+f-1-y}{s-1}$, or equivalently $y(1 + \frac{1}{s-1}) \leq \frac{s+f-1}{s-1}$, and so $y \leq \frac{s+f-1}{s} = 1 + \frac{f-1}{s}$. □

Definition 3 *Let f_3 be the highest value of $f \in \mathbb{N}, f < f^*$ such that*

$$\frac{f^* - 1}{s - 1} < 1 + \frac{f - 1}{s} \leq \text{GLOBMAX}, \tag{11}$$

if such a value exists. (Else f_3 is left undefined.)

Lemma 21 *If (5) does not hold and f_3 is defined, then $f_3 = f^* - 1$.*

Proof We first note that if the condition $\frac{f^*-1}{s-1} < 1 + \frac{f-1}{s}$ holds for some value $f = f_0 < f^*$, then it holds for any value in the interval $[f_0, f^*)$.

Suppose that (5) does not hold and $f_3 \leq f^* - 2$. Thus the condition $1 + \frac{f-1}{s} \leq$ GLOBMAX does not hold for $f = f_3 + 1$, so $1 + \frac{f_3}{s} >$ GLOBMAX. Let $\varphi = 1 + f_3/s$. Then $\lceil s(\varphi - 1) \rceil = \lceil f_3 \rceil = f_3$. Moreover, from the definition of f_3 and the assumptions on f_3 we see that GLOBMAX $> \frac{f^*-1}{s-1} \geq \frac{f_3+1}{s-1}$. Therefore $\varphi > \frac{f_3+1}{s-1}$, which implies $(s-1)\varphi > f_3 + 1$ and therefore $(s-1)\varphi - \lceil s(\varphi - 1) \rceil = (s-1)\varphi - f_3 > 1$. Now, for any $\varphi' \leq 1 + f_3/s$, in particular for $\varphi' =$ GLOBMAX, we clearly have that $\lceil s(\varphi' - 1) \rceil \leq f_3$. But since we saw above that GLOBMAX $> \frac{f_3+1}{s-1}$, we find $(s-1)$GLOBMAX $> f_3 + 1$ and therefore (5) holds. This is a contradiction. □

Lemma 22 *If $y >$ FASTMAX and $f_3 = f^* - 1$, there is an equilibrium instance with $f^* - 1$ jobs on the fast machine where one job is larger than 1 and $y = 1 + (f^* - 2)/s$. For any equilibrium instance with at most $f^* - 1$ jobs on the fast machine, $y \leq 1 + (f^* - 2)/s$ if one of those jobs is larger than 1.*

Note that $f^* \geq 2$ and therefore $f^* - 2 \geq 0$ by Lemma 15.

Proof Consider the following instance. There is a job of size $y = 1 + \frac{f^*-2}{s}$ on the slow machine HIGHLOAD. On the fast machine, there are $k = \lceil s(y-1) \rceil$ jobs of size 1 as well as 1 job of size $q = (s-1)y - \lceil s(y-1) \rceil \leq s - y$. Thus $x = (s-1)y/s$. On each empty slow machine, we place $\max(x - 1, 0)$ of sand. It then immediately follows that this is an equilibrium, since the condition of Lemma 9 is satisfied, and no job can improve by moving to a slow machine with load $\max(x - 1, 0)$.

We need to show that the optimal makespan is 1. The total size of all the jobs is at most $M + s$ because $y \leq$ GLOBMAX by definition of f_3, so that the loads on the fast machine and on HIGHLOAD are at most those from the example from Theorem 5, where the total size was exactly $M + s$. This holds because we maintain $x = (s-1)y/s$, which is now not larger. If $x > 1$, the loads on the remaining machines are also smaller in the current example.

Suppose $x - 1 < 0$. Then $y < s/(s-1)$. For $y = s/(s-1) <$ GLOBMAX (Lemma 13), we have $x = 1$, and the loads on the other machines are zero. It is clear that for smaller y, if we maintain $x = \frac{s-1}{s}y$, the total size of the jobs on HIGHLOAD and the fast machine is smaller. Thus also in the case that $x - 1 < 0$ we have that the total size of all the jobs in the current example is not more than $M + s$.

We also need that $q \geq 1$. For $y = 1 + (f^* - 2)/s$, we have $\lceil s(y-1) \rceil = \lceil f^* - 2 \rceil = f^* - 2$. This means that $q \geq 1$ holds if

$$(s-1)(1 + (f^* - 2)/s) - (f^* - 2) \geq 1,$$

or $(s-1)(1 + (f^* - 2)/s) \geq f^* - 1$, or equivalently $1 + (f^* - 2)/s \geq (f^* - 1)/(s-1)$. But this follows from the assumption that $f_3 = f^* - 1$.

Note also that this immediately implies that $y \leq s - q \leq s - 1$. In addition, we actually find that $k = f^* - 2 \leq M$, so each of these $f - 1$ jobs can be placed on their own machine in the optimal solution, thus the optimal makespan is 1. The second claim follows immediately from Lemma 20. □

We are now ready to give a full characterization of the POA in the case that $f^* \leq M$ and (5) does not hold.

Theorem 7 *If $f^* \leq M$ and (5) does not hold, and $y >$ FASTMAX, then*

$$y = \text{POA} = \min\left(s, \max\left(\text{MAXSLOW}(f^*), \frac{f^* - 1}{s - 1}, 1 + \frac{f^* - 2}{s}\right)\right).$$

Proof The first upper bound follows from Lemma 6. There are three cases, depending on where the maximum is achieved. The case numbers indicate the term that achieves the maximum.

Case 1. We use Lemma 18 to get an instance with $y = \min(s, \text{MAXSLOW}(f^*))$. The lemma (combined with Lemma 6) states that no higher load can be achieved on a slow machine using at least f^* jobs on the fast machine. If there are less than f^* jobs on the fast machine, we have the bounds from Lemma 19 and Lemma 22 which are not larger in this case.

Case 2. We use Lemma 19 to get an instance with $y = \min(s, (f^* - 1)/(s - 1))$ with $f^* - 1$ jobs of size at most 1 on the fast machine. Similar to in Case 1, it can be seen that other possibilities for jobs on the fast machine do not give higher values for y.

Case 3. We use Lemma 22 to get an instance with $y = 1 + (f^* - 2)/s$. The proof of Lemma 22 shows that if $1 + (f^* - 2)/s \geq \frac{f^* - 1}{s - 1}$, then $1 + (f^* - 2)/s \leq s - 1$. □

3.4.3 The case $f^* = M + 1$

Suppose that $f^* = M + 1$. This case requires special attention because of the following lemma.

Lemma 23 *For $s > 2$, consider an instance where $y >$ FASTMAX. In this case, there exists an instance which is an equilibrium with the same loads on all machines and the same optimal makespan, where no two jobs on the fast machine have a total size of at most 1.*

Proof We first show that we can assume $f \leq M + 1$. Suppose this does not hold. By Lemma 8, there are at least $f - 1 \geq M + 1$ jobs on the fast machine which are on slow machines in the optimal solution, so there are at least two jobs from the same slow machine. Now these two jobs can be merged without affecting the equilibrium or the optimal makespan.

For the second statement, if there do exist two such jobs, we merge them into one larger job. Since $f \leq M + 1$, this leaves at most M jobs on the fast machine, all of which have size at most 1. Thus we can assign each such job to its own slow machine, and the other jobs as in the previous case. □

Thus, if we have $M + 1$ jobs on the fast machine, in the optimal solution all these jobs are on different machines, and in particular one of them is on the fast machine both in the optimal solution and in $\mathcal{A}$. Hence the sum of y and one of the jobs on the fast machine must be at most s.

Definition 4 *Let*

$$\text{SPECIALMAX} = \frac{(s + 1)M}{s + (M - 1)(2 - 1/s)}.$$

We will see that SPECIALMAX is an upper bound for the POA in this special case. We first prove two technical lemmas which give upper bounds for the value SPECIALMAX.

Lemma 24 *For $M \geq 2$ and $s > 2$, if* SPECIALMAX $\leq$ MAXSLOW$(M + 1)$, *then* SPECIALMAX $< 1 + M/s$.

Proof First of all, SPECIALMAX and $1+M/s$ are both continuous for $s \geq 2$ and $M \geq 2$. Furthermore, MAXSLOW$(M+1)$ is also continuous for $s \geq 2$ and $M \geq 2$ by Observation 3 and Lemma 16. Solving for M, we have SPECIALMAX $=$ MAXSLOW$(M+1)$ for

$$M_{1,2} = \frac{s \pm \sqrt{17s^2 - 4s^3 - 20s + 8}}{2s - 4} .$$

(For $s = 2$, we find $M_{1,2} = \{\frac{1}{2}, 1\}$, and SPECIALMAX $<$ MAXSLOW$(M+1)$ for $M \geq 2$.) The values $M_{1,2}$ are not real if $17s^2 - 4s^3 - 20s + 8 < 0$, that is, if $s \geq 2.65$. Taking for instance $M = 3$ and $s = 3$, we find SPECIALMAX $= 1.89 > 1.8 =$ MAXSLOW$(M+1)$. Thus if SPECIALMAX $\leq$ MAXSLOW$(M+1)$, we know that $2 \leq s < 2.65$ since both functions are continuous for $s \geq 2$ and $M \geq 2$. On the other hand, we have SPECIALMAX $= 1 + M/s$ for

$$M_3 = s(s-1)^2/(2s-1).$$

The value M_3 is continuously increasing for all $s \geq 2$: the derivative is $(4s^3 - 7s^2 + 4s - 1)/(2s-1)^2$, which is positive for all $s \geq 2$ since the numerator is larger than $4s^3 - 8s^2 + 4s - 1 \geq 4s - 1 > 0$, and the denominator is positive. Furthermore, for $s = 2.65$, $M_3 < 1.678 < 2$. Thus for $M \geq 2$, we never have SPECIALMAX $= 1 + M/s$ for $2 < s \leq 2.65$. Since SPECIALMAX $= 12/7 < 1 + M/s = 2$ for $s = 2$ and $M = 2$, and both functions are continuous for $s \geq 2$ and $M \geq 2$, the lemma is proved. □

Lemma 25 *For $s \geq 2$ and $M \geq 2$,* SPECIALMAX $< s$.

Proof We have equality for $s = \frac{1}{2}(-M + 2 + \sqrt{M^2 + 4M})$. This is less than 2 for all $M \geq 2$, and for $s = 2$ and $M = 2$ we have SPECIALMAX $= 6/3.5 < 2 = s$. Finally, SPECIALMAX is continuous in s and M for $s \geq 2$, $M \geq 2$. This proves the lemma. □

Lemma 26 *Let $M \geq 2$ and $s > 2$. If there are $M+1$ jobs on the fast machine, and $y >$* FASTMAX*, then*

$$y \leq \min\left(\text{MAXSLOW}(M+1), \text{SPECIALMAX}\right). \tag{12}$$

An instance with this y exists if $f^* = M + 1$.

Proof The first upper bound follows from Lemma 14. Denote the size of the smallest job on the fast machine by a. Since the optimal makespan is 1, and since we may assume no two jobs on the fast machine have total size less than 1 by Lemma 23, we must have $a \leq s - y$ (and $y \leq s - a < s$).

We have $x \geq \frac{s-1}{s} y$ as usual (Lemma 9), and the condition that $z + a \geq x$, because the job of size a may not benefit from moving to a slow machine. This implies $z \geq x - a \geq y(s-1)/s + y - s = y(2 - 1/s) - s$. Moreover, the total size of all the jobs must be at most $M + s$, leading to the condition that

$$y\left(1 + (s-1) + (M-1)(2-1/s)\right) - (M-1)s \leq M + s. \tag{13}$$

For $M \geq 2$, $s > 2$, this is equivalent to $y \leq$ SPECIALMAX. Note that this bound is also valid in case $y(2 - 1/s) - s < 0$. (In this case, it would however be better to use the bound $z \geq 0$.) In particular, the denominator of SPECIALMAX is positive for all $s > 2$, $M \geq 2$.

For the second claim, assume $f^* = M + 1$. Note that MAXSLOW$(f^*) > 0$ by definition, and $f^* > s$ by Lemma 15. If MAXSLOW$(M+1) \leq$ SPECIALMAX, it follows that if we take $y =$ MAXSLOW$(M+1) > 0$, inequality (13) is satisfied, whereas (8) holds with equality. We therefore have

$$\begin{aligned} y\left(s + (M-1)(2-1/s)\right) - (M-1)s \leq M + s = y\left(s + (M-1)\left(1 - \frac{1}{s}\right)\left(1 - \frac{s}{M+1}\right)\right) \\ \Rightarrow y(M-1) - (M-1)s \leq y(M-1)\left(1 - \frac{1}{s}\right)\left(-\frac{s}{M+1}\right) \\ \Rightarrow y - s \leq y\left(-\frac{s-1}{M+1}\right) \end{aligned}$$

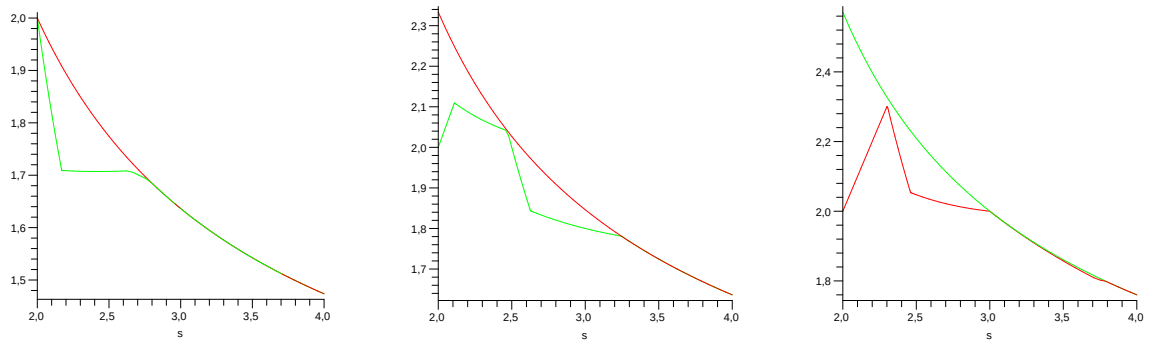

Figure 2: The price of anarchy for $M = 2, 3, 4$ as a function of s. The top line in each case is GLOBMAX, a global upper bound on the POA. The bottom line is the actual POA for each s. For $M = 4$ and $s \in [3, 3.7]$, we have POA $= 1 + 3/s <$ GLOBMAX.

This implies $y(s-1)/(M+1) \leq s - y$. This immediately shows that we can use the instance from Lemma 18 for $f^* = M+1$, and in the optimal solution assign the job of size $y = \text{MAXSLOW}(M+1)$ to the fast machine together with one job of size $y(s-1)/(M+1) \leq s - y$. Note that in this case we also have $y \leq s - y(s-1)/(M+1) < s$, that is, we do not have to worry about the case $\text{MAXSLOW}(M+1) > s$.

On the other hand, if $\text{SPECIALMAX} < \text{MAXSLOW}(M+1)$, it follows that if we take $y = \text{SPECIALMAX}$, we find $s - y < y(s-1)/(M+1)$. Since $\text{SPECIALMAX} < s$ by Lemma 25, we have $s - y > 0$ also in this case. In this case we place one job of size $s - y$ on the fast machine and M jobs of total size $y(s-1) - (s-y) = s(y-1)$. In order for the optimal makespan to be 1, we must have $s(y-1)/M \leq 1$.

To prove this, we use that $y = \text{SPECIALMAX} \leq 1 + M/s$, which holds by Lemma 24. This implies that $sy \leq M + s$, and then $y(s-1) \leq M + s - y$. This last value, $M + s - y$, would be the total size of the jobs on the fast machine if we placed M jobs of size 1 there plus a job of size $s - y$. Thus that last inequality implies that the M jobs in our instance have size at most 1, since we have $sx = y(s-1)$.

Finally, since we have $M+1$ jobs on the fast machine in this instance, one of them of size $s - y < y(s-1)/(M+1)$, and the other M jobs all equal-sized, it follows that those M jobs all have size more than $s - y$. Thus the job of size $s - y$ is indeed the smallest on the fast machine, and since $sx = y(s-1)$, this means that (13) is a sufficient condition to have an equilibrium. □

Theorem 8 *If (5) does not hold, $y >$ FASTMAX, $s > 2$ and $f^* = M + 1$, the* POA *is given by*

$$\min\left(s, \max\left(\min\left(\text{MAXSLOW}(M+1), \text{SPECIALMAX}\right), 1 + \frac{M-1}{s}, \frac{M}{s-1}\right)\right).$$

Proof If the maximum is achieved in the first term, we use one of the instances from Lemma 26 to give a tight lower bound, depending on where the inner minimum is achieved. Else, the bound follows as in the proof of Theorem 7. Note that Lemmas 19 and 22 do not require $f^* \leq M$. □

Theorem 9 *The* POA *is achieved on a slow machine for all $M \geq 2$ and $s > 2$.*

Proof It can be verified that the bounds in Theorem 7 and 8 are larger than FASTMAX for $M \geq 2$ and $s \in (2, 4.57]$. The claimed result then follows for all $s > 2$ by Lemma 11 and Theorem 6.

For instance, for $M \geq 10$, POA $> 2 >$ FASTMAX in the interval $s \in (2, 4.57]$. See Figures 2 and 3 for graphs of the POA as a function of s for several values of M. □

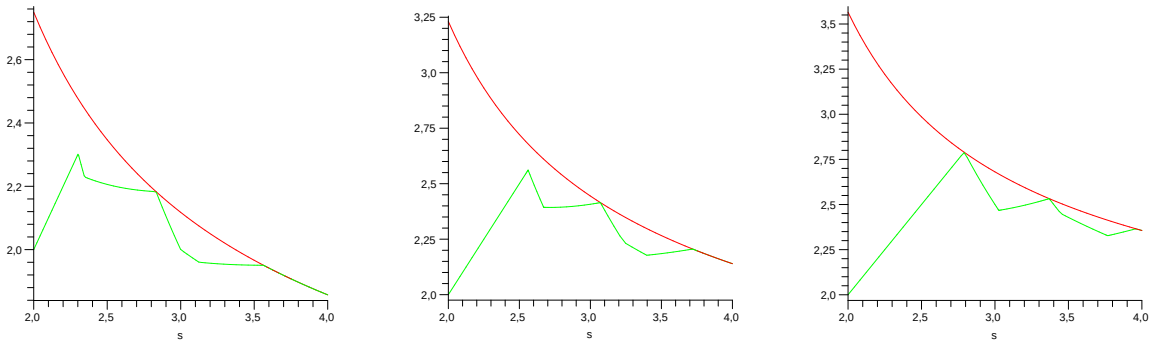

Figure 3: The price of anarchy for $M = 5, 10, 20$ as a function of s. The top line in each case is GLOBMAX, a global upper bound on the POA. The bottom line is the actual POA for each s.

3.5 The limit of the POA for $s \to \infty$ and for $M \to \infty$

If $s \to \infty$, by Theorem 5 we find POA $\to 1$ for any $M \geq 2$. What happens with the POA if M grows without bound? By Theorem 2, POA $\to 2$ as $M \to \infty$ for $s \in [1, 2]$; the third term in the maximum tends to 2 (so the limit is not lower) and FASTMAX $\to 2$ as $M \to \infty$ (so the limit is not higher). By Theorem 6, POA $=$ GLOBMAX for $s \geq 4.562$. To answer this question for $s \in (2, 4.562]$, we first need to consider the value MAXSLOW(f). By Definition 1, we have

$$\lim_{M\to\infty} \text{MAXSLOW}(f) = \frac{1}{(s-1)(1-s/f)/s} = \frac{1}{(s-1)/s-(s-1)/f} = \frac{sf}{(s-1)(f-s)}\,.$$

From this, we can derive $\lim_{M\to\infty} f^*$ using Definition 2. We have

$$f = (s-1)\cdot\frac{sf}{(s-1)(f-s)} = \frac{sf}{f-s} \Leftrightarrow s = f-s \vee f = 0 \Leftrightarrow f = 2s \vee f = 0.$$

Hence, for $s \leq 4.562$ and large enough M, we certainly have $f^* < M$. (Note that $f^* > 0$ by Lemma 15.) We have

$$\lim_{M\to\infty} \text{MAXSLOW}(2s) = \frac{2s^2}{(s-1)s} = \frac{2s}{s-1} = \lim_{M\to\infty} \text{GLOBMAX}.$$

Since POA $\leq$ GLOBMAX by Lemma 4, and FASTMAX $\leq$ GLOBMAX by Lemma 11, we can conclude the following from Lemma 18.

Theorem 10 *For* $s \in [1, 2]$, $\lim_{M\to\infty}$ POA $= 2$. *For* $s \in [2, 3]$, $\lim_{M\to\infty}$ POA $= s$. *For* $s \geq 3$, $\lim_{M\to\infty}$ POA $= 2s/(s-1)$.

4 One fast machine: the SPOA

In this section we demonstrate the fact that the SPOA is strictly smaller than the POA. We consider the overall bounds (i.e., the supremum bounds over all values of s and M) and compare them. The overall bound on the POA, as implied by the previous sections, is 3.

Theorem 11 *The* SPOA *is 2 for* $M \leq 5$. *For any* M, SPOA $\leq \frac{3+\sqrt{5}}{2} \approx 2.618$. *For* $M \geq 16$, SPOA $\geq \frac{1+\sqrt{13}}{2} \approx 2.3027756$.

Proof We first slow a lower bound of 2 for any value of M. Consider from the following instance. The fast machine has speed 2. There are M jobs of size 1, and one job of size 2. An optimal solution is clearly to assign one unit job to each slow machine, and the larger job to the faster machine. This gives OPT $= 1$. In a schedule S that we consider, two jobs of size 1 are scheduled on the fast machine. One slow machine is empty, one has a job of size 2, and all remaining slow machines have one job of size 1. It can be seen that no coalition can improve from trading places; the two jobs on the fast machine can not obtain smaller load by moving, so they would not move to a slow machine. As long as these two jobs do not move, no other job can benefit from moving.

We next prove an upper bound. Consider a strong equilibrium S. We use the notations HIGHLOAD, x, y and z, as before. Let r be the smallest job on the fast machine as well as its size. Lemma 3 and Lemma 6 both hold for any $s \geq 1$ and any schedule that is a pure equilibrium, thus we can use them in this proof. If $x \leq 1$ then since any job on HIGHLOAD is of size at most s, we get that $y \leq x + 1 \leq 2$ (since moving this job to the fast machine is not beneficial). In this case the SPOA is no larger than 2, and therefore, since by Lemma 3, we have $x < 2$, we only need to consider a case where $1 < x < 2$, and the SPOA is achieved on HIGHLOAD.

Since $x > 1$, there must be a machine with load smaller than 1, and therefore $z < 1$. If HIGHLOAD contains a job of size d that OPT assigns to a slow machine, we have $z + d \geq y$ and therefore $y \leq z + 1 < 2$. Thus HIGHLOAD only contains jobs assigned by OPT to the fast machine (and SPOA $\leq s$). We therefore have $y \leq s$ and we can assume that $s > 2$, otherwise we would again get a SPOA of at most 2.

Since $x > 1$, in the schedule S, the fast machine must have a job that OPT assigns to a slow machine. Thus $r \leq 1$. Since the job of size r does not benefit from moving to the least loaded slow machine, we get $z + r \geq x$.

We claim that $sx+y \geq r+sy$ and therefore $sx \geq (s-1)y+r$. Recall that HIGHLOAD contains only jobs that belong to the fast machine (otherwise $y \leq 2$ in OPT). Consider the coalition consisting of all the jobs scheduled on HIGHLOAD and a job of size r, scheduled on the fast machine. Upon a deviation of this coalition, the job r moves to the slow machine HIGHLOAD and as a result, has a delay of $r \leq 1$. Its previous delay was $x > 1$. Since there exists a job of the coalition which does not reduce it load upon deviation, the jobs of HIGHLOAD are those that do not benefit from moving: we find $(sx-r+y)/s \geq y$. This proves the claim.

Let W be the total size of all the jobs. We get

$$\begin{aligned} Ms + s^2 \geq Ws &\geq s^2x + ys + (M-1)zs \geq rs + s^2y + (M-1)(x-r)s \\ &= rs + s^2y + sx(M-1) - rs(M-1) \\ &\geq r(s - sM + s) + s^2y + ((s-1)y + r)(M-1) \\ &= r(2s - sM + M - 1) + y(s^2 + sM - M - s + 1). \end{aligned}$$

If $2s-sM+M-1 \leq 0$, then we use $r \leq 1$ to get $Ms+s^2 \geq 2s-sM+M-1+y(s^2+sM-M-s+1)$ or

$$y \leq \frac{s^2 + 2Ms - 2s - M + 1}{s^2 + sM - s - M + 1} \tag{14}$$

(note that $s^2 + sM - M - s + 1 = s^2 + (s-1)(M-1) > 0$). If $2s + M - 1 - sM \geq 0$ we use $r \geq 0$ to get,

$$y \leq \frac{Ms + s^2}{s^2 + Ms - s - M + 1} = 1 + \frac{s + M - 1}{s^2 + Ms - s - M + 1} \leq 2\,,$$

since $s + M - 1 \leq s^2 + Ms - s - M + 1 \Leftrightarrow s^2 + Ms + 2 \geq 2s + 2M$ which holds for any $s \geq 2$ (by $(s-1)^2 \geq 0$).

By (14), $y \leq 2$ holds if $s^2 - M + 1 \geq 0$, i.e., if $M \leq 5$ (since $s \geq 2$). For larger M, we show that $y \leq \frac{2s-1}{s-1}$ or $y - 1 \leq \frac{s}{s-1}$. For this we need to show $\frac{Ms-s}{s^2+sM-s-M+1} \leq \frac{s}{s-1}$, i.e., $(M-1)(s-1) \leq s^2 + sM - s - M + 1$ which holds since $s^2 > 0$. Since we also know SPOA $\leq s$, we get SPOA ≤ 2.618.

For the lower bound, consider a fast machine of speed $\sigma = \frac{1+\sqrt{13}}{2} \approx 2.303$. In an optimal schedule, the fast machine has a job of size σ, there are 12 slow machines that contain two jobs, of sizes $\frac{1}{4}$ and $\frac{3}{4}$ and the remaining slow machines have one job of size 1 each. Therefore OPT $= 1$. In the schedule we consider, the fast machine has four jobs of size 1, 12 slow machines have jobs of size $\frac{3}{4}$, three slow machines have four jobs of size $\frac{1}{4}$ each, one slow machine has a job of size σ and the remaining slow machines have jobs of size 1. The load on the fast machine is $\frac{2(\sqrt{13}-1)}{3} \approx 1.736865$ and the makespan is achieved on the slow machine which contains the job of size $\sigma \approx 2.303$.

Consider the terms on which each type of job would join a coalition. We first discuss the case where the job of size σ does not join. If no job which is assigned to the fast machine joins, then no job which is scheduled to a slow machine would want to move to the fast machine, and jobs that are single on their machine would not join, so no coalition can be created. On the other hand, since the load on the fast machine is strictly less than 1.75, then the jobs on this machine would join a coalition only if they could move to a slow machine with a resulting load of less than 1.74, i.e. due to the structure of the instance, the load excluding the additional job should be at most $\frac{1}{2}$. For that, some jobs of size $\frac{1}{4}$, $\frac{3}{4}$ or 1 would need to join the coalition. There is clearly no advantage to exchanges between jobs of size 1, thus we need to consider only smaller jobs. A job of size $\frac{3}{4}$ benefits from moving to the fast machine only if the resulting total size there is no larger than 1.74, i.e., at most 1.5, but this can happen if all jobs of size 1 on the fast machine join the coalition. Jobs of size $\frac{1}{4}$ would move to the fast machine if the resulting total size there is at most 2.25. For that, at least two jobs from the fast machine need to join the coalition. We consider three cases based on the number of jobs migrating from the fast machine. If two jobs migrate, only a single job of size $\frac{1}{4}$ can migrate, so the room created for the migrating jobs of size 1 does not suffice. If three jobs migrate, then five jobs of size $\frac{1}{4}$ can migrate, and there is room only for two migrating jobs of size 1. If four large jobs migrate, in order to make room for the migrating jobs of size 1, six jobs of size $\frac{1}{4}$ and one job of size $\frac{3}{4}$ must migrate (if there are more jobs of size $\frac{3}{4}$ migrating, and less pairs of jobs of size $\frac{1}{4}$, then the load on the fast machine would only be larger). This would create a total size of 2.25 on the fast machine, therefore the job of size $\frac{3}{4}$ would not join the coalition.

If the job of size σ joins the coalition, at least two jobs of size 1 from the fast machine must join the coalition as well, since $\frac{3+\sigma}{\sigma} = \sigma$. In order to make it beneficial for these two jobs to migrate, and since moving both of them to the machine that becomes empty would create a load of 2 there, at least two jobs of size $\frac{1}{4}$ or one job of size $\frac{3}{4}$ needs to join the coalition. But then the load on the fast machine is already larger than 1 due to the job of size σ, so no such jobs would join the coalition. □

5 Conclusion

We studied the POA as a function of the number of different speeds. We found a tight overall bound, and completely resolved the case where all machines are identical, except for one faster machine. It can be interesting for find a tighter result for the SPOA as a function of the number of different speeds, p, and find whether it is strictly smaller than $p + 1$, which is the POA for this case. Another direction is to study the influence of additional factors on the POA, such as the ratio of the largest and smallest speeds, or even as a function of all the machine speeds, possibly as the solution of a mathematical program.

References

[1] N. Andelman, M. Feldman, and Y. Mansour. Strong price of anarchy. *Games and Economic Behavior*, 65(2):289–317, 2009.

[2] R. J. Aumann. Acceptable points in general cooperative n-person games. In A. W. Tucker and R. D. Luce, editors, *Contributions to the Theory of Games IV, Annals of Mathematics Study 40*, pages 287–324. Princeton University Press, 1959.

[3] Y. Cho and S. Sahni. Bounds for List Schedules on Uniform Processors. *SIAM Journal on Computing*, 9(1):91–103, 1980.

[4] A. Czumaj. Selfish routing on the internet. In J. Leung, editor, *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*, chapter 42. CRC Press, 2004.

[5] A. Czumaj and B. Vöcking. Tight bounds for worst-case equilibria. *ACM Transactions on Algorithms*, 3(1), 2007.

[6] A. Epstein, M. Feldman, and Y. Mansour. Strong equilibrium in cost-sharing connection games. In *Proc. of the 8th ACM Conference on Electronic Commerce (EC'07)*, pages 84–92, 2007.

[7] Leah Epstein. Equilibria for two parallel links: the strong price of anarchy versus the price of anarchy. *Acta Inf.*, 47(7–8):375–389, 2010.

[8] Eyal Even-Dar, Alexander Kesselman, and Yishay Mansour. Convergence time to nash equilibrium in load balancing. *ACM Transactions on Algorithms*, 3(3), 2007.

[9] R. Feldmann, M. Gairing, T. Lücking, B. Monien, and M. Rode. Nashification and the coordination ratio for a selfish routing game. In *Proc. of the 30th International Colloquium on Automata, Languages and Programming (ICALP2003)*, pages 514–526, 2003.

[10] A. Fiat, H. Kaplan, M. Levy, and S. Olonetsky. Strong price of anarchy for machine load balancing. In *Proc. of the 34th International Colloquium on Automata, Languages and Programming (ICALP2007)*, pages 583–594, 2007.

[11] G. Finn and E. Horowitz. A linear time approximation algorithm for multiprocessor scheduling. *BIT Numerical Mathematics*, 19(3):312–320, 1979.

[12] Dimitris Fotakis, Spyros C. Kontogiannis, Elias Koutsoupias, Marios Mavronicolas, and Paul G. Spirakis. The structure and complexity of Nash equilibria for a selfish routing game. *Theor. Comput. Sci.*, 410(36):3305–3326, 2009.

[13] T. Gonzalez, O. H. Ibarra, and S. Sahni. Bounds for LPT Schedules on Uniform Processors. *SIAM Journal on Computing*, 6(1):155–166, 1977.

[14] R. Holzman and N. Law-Yone. Strong equilibrium in congestion games. *Games and Economic Behavior*, 21(1-2):85101, 1997.

[15] Nicole Immorlica, Li (Erran) Li, Vahab S. Mirrokni, and Andreas S. Schulz. Coordination mechanisms for selfish scheduling. *Theor. Comput. Sci.*, 410(17):1589–1598, 2009.

[16] E. Koutsoupias, M. Mavronicolas, and P. G. Spirakis. Approximate equilibria and ball fusion. *Theory of Computing Systems*, 36(6):683–693, 2003.

[17] E. Koutsoupias and C. H. Papadimitriou. Worst-case equilibria. In *Proc. of the 16th Annual Symposium on Theoretical Aspects of Computer Science (STACS'99)*, pages 404–413, 1999.

[18] E. Koutsoupias and C. H. Papadimitriou. Worst-case equilibria. *Computer Science Review*, 3:65–69, 2009.

[19] Annamária Kovács. Tighter approximation bounds for LPT scheduling in two special cases. *J. Discr. Alg.*, 7(3):327–340, 2009.

[20] R. Li and L. Shi. An on-line algorithm for some uniform processor scheduling. *SIAM Journal on Computing*, 27(2):414–422, 1998.

[21] J. W. S. Liu and C. L. Liu. Bounds on scheduling algorithms for heterogeneous computing systems. In Jack L. Rosenfeld, editor, *Proceedings of IFIP Congress 74*, volume 74 of *Information Processing*, pages 349–353, 1974.

[22] Marios Mavronicolas and Paul G. Spirakis. The price of selfish routing. *Algorithmica*, 48(1):91–126, 2007.

[23] J. Nash. Non-cooperative games. *Annals of Mathematics*, 54(2):286–295, 1951.

[24] N. Nisan and A. Ronen. Algorithmic mechanism design. *Games and Economic Behavior*, 35:166–196, 2001.

[25] C. H. Papadimitriou. Algorithms, games, and the internet. In *Proc. of the 33rd Annual ACM Symposium on Theory of Computing (STOC2001)*, pages 749–753, 2001.

[26] T. Roughgarden. *Selfish routing and the price of anarchy*. MIT Press, 2005.

[27] T. Roughgarden and É. Tardos. How bad is selfish routing? *Journal of the ACM*, 49(2):236–259, 2002.

[28] P. Schuurman and T. Vredeveld. Performance guarantees of local search for multiprocessor scheduling. *Informs Journal on Computing*, 19(1):52–63, 2007.

[29] M. Tennenholtz and O. Rozenfeld. Strong and correlated strong equilibria in monotone congestion games. In *Proc. of the 2nd International Workshop on Internet and Network Economics (WINE2006)*, page 7486, 2006.