Coordination Mechanisms from (almost) all Scheduling Policies*

Sayan Bhattacharya† Sungjin Im‡ Janardhan Kulkarni§ Kamesh Munagala¶

August 16, 2013

Abstract

We study the *price of anarchy* of coordination mechanisms for a scheduling problem where each job j has a weight w_j, processing time p_{ij}, assignment cost h_{ij}, and communication delay (or release date) r_{ij} on machine i. Each machine is free to declare its own scheduling policy. Each job is a selfish agent and selects a machine that minimizes its own disutility, which is equal to its weighted completion time plus its assignment cost. The goal is to minimize the total disutility incurred by all the jobs. Our model is general enough to capture scheduling jobs in a distributed environment with heterogeneous machines (or data centers) that are situated across different locations.

Our main result is a *characterization* of scheduling policies that give a small (robust) Price of Anarchy. More precisely, we show that whenever each machine independently declares any scheduling policy that satisfies a certain *bounded stretch* condition introduced in this paper, the game induced between the jobs has a small Price of Anarchy. Our characterization is powerful enough to test almost all popular scheduling policies. On the technical side, to derive our results, we use a potential function whose derivative leads to an instantaneous smoothness condition, and linear programming and dual fitting. To the best of our knowledge, this is a novel application of these techniques in the context of coordination mechanisms, and we believe these tools will find more applications in analyzing PoA of games. We also extend our results to the ℓ_k-norms objectives.

*This work is supported by Munagala's Alfred P. Sloan Research Fellowship, an award from Cisco, and by NSF grants CCF-0745761, CCF-1008065, and IIS-0964560.

†Max Plank Institute for Informatics, Saarbrucken, Germany. Email: bsayan@mpi-inf.mpg.de. Work done while the author was in Duke University.

‡Department of Computer Science, Duke University, Durham NC 27708. Email: sungjin@cs.duke.edu.

§Department of Computer Science, Duke University, Durham NC 27708. Email: kulkarni@cs.duke.edu.

¶Department of Computer Science, Duke University, Durham NC 27708. Email: kamesh@cs.duke.edu.

1 Introduction

Explosive growth of data has driven distributed computing to evolve at an unprecedented pace. In modern distributed systems, there are a large number of machines which are clustered and connected in a variety of topologies, and situated across different geographical locations. Hence routing jobs can incur considerable costs and communication delays. Machines are also inherently heterogeneous, having very different architectures and accesses to energy resources – some machines can process some jobs more efficiently and at cheaper costs. Due to the large scale of such systems, centralized algorithms for scheduling jobs is not very practical. Moreover, in many scenarios each job is a selfish agent that strategically selects a machine for getting processed. Can such a decentralized system perform well in spite of the strategic behaviors of the jobs? We explore this question under a mechanism design paradigm called *coordination mechanisms* [8].

1.1 Model

There is a set $\mathcal{J}$ of n jobs, and a set $\mathcal{M}$ of m unrelated machines. A job j has a weight w_j, and it needs p_{ij} units of *processing time* if scheduled on machine i.[1] The job j has a *communication delay* (or *release date*) r_{ij} on machine i, i.e, the machine can start processing the job only after time r_{ij}. Further, the job j incurs an *assignment cost* h_{ij} if it is assigned to the machine i. This, for example, captures the energy costs.

In a sharp contrast with the centralized view of classical scheduling models, here each job is a self-interested and autonomous agent free to select its own machine. Every machine declares its scheduling policy in advance, and this induces a *simultaneous-move* game between the jobs. The strategy of a job consists of choosing the machine where it will get processed. Each job wants to minimize its own *disutility*, which is its weighted completion time plus its assignment cost. A Nash equilibrium of this game is a stable outcome where no job can reduce its disutility by switching to another machine. The strategic interactions among the jobs may lead to overall degradation in system performance. The standard benchmark to measure this deterioration is the Price of Anarchy (PoA), first introduced in [18]. This is the worst case (maximum possible) ratio of total disutility of the jobs in a Nash Equilibrium to that in an optimal solution (which assumes centralized assignment and no strategic behavior).

We let different machines declare different scheduling policies. Each of these policies, however, must be *strongly local* [2] since a machine i only knows the w_j, p_{ij}, r_{ij}, and h_{ij} values of the jobs j that were assigned to it. In the absence of a global view of the input, a reasonable option for a machine is to declare a scheduling policy that (approximately) minimizes its own share of the objective, namely, the total weighted completion time of all the jobs assigned to it.[3] This raises a compelling question:

- *Do all scheduling policies that are $O(1)$-approximate on a single machine result in coordination mechanisms with $O(1)$ price of anarchy? If the answer is no, is there a characterization of single-machine scheduling policies that induce $O(1)$ price of anarchy?*

1.2 Our Results

In this paper, we present a general recipe for designing coordination mechanisms for minimizing the total weighted completion time of the jobs plus their assignment costs. The machines need not agree upon a specific scheduling policy. Nevertheless, the system will have small constant price of anarchy as long as every machine selects a scheduling policy that satisfies a certain *bounded stretch* condition introduced in this paper (see Sections 1.2.1,1.2.2). We further show that almost all scheduling policies used in practice satisfy this

[1] Our analysis can be easily extended to unrelated weights w_{ij}.

[2] If a job j is assigned to a machine i, then the machine i is not aware of the characteristics of the job j (processing lengths, release dates etc.) on other machines $i' \neq i$.

[3] The machine can ignore the assignment costs as their contribution to the objective is fixed once each job selects its strategy.

condition. We complement this positive result by showing that there exists a widely used $O(1)$-approximate single-machine scheduling policy that does not satisfy the bounded stretch condition, *and* induces a game with large price of anarchy. Finally, we extend our results to *all norms* of completion times of the jobs (see Section 1.2.3), and present a general black box reduction to *non-preemptive* scheduling policies (see Section 1.2.4).

All the previous works on coordination mechanisms for completion time scheduling [10, 9] focused on the case without communication delays (release dates) and assignment costs, i.e., $r_{ij} = h_{ij} = 0$. We note that release dates introduce a significant complexity in scheduling, and algorithms for problems without release dates typically do not generalize to those with release dates. For example, the underlying optimization problem on a single machine is polynomial time solvable via a greedy algorithm without release dates, but is NP-HARD with release dates [22]. Furthermore, the previous works [10, 9] analyze very specific scheduling policies, and restrict all machines to announce the same scheduling policy. In contrast, we give a new dimension to the problem by allowing different machines to declare different scheduling policies. This generalization models the real world applications more accurately, since the machines (or data centers) are typically owned and operated by different entities.

Our results rely upon two novel techniques. (a) A potential function that leads to an instantaneous *smoothness* condition; and (b) Linear programming and dual fitting.

1.2.1 Characterization of Good Single-machine Scheduling Policies

We introduce the notion of a scheduling policy with *bounded stretch* in the definition given below.[4]

Definition 1.1. *Suppose that a machine is processing a given set of jobs. The scheduling policy followed by the machine has a stretch α iff the completion time C_j of each job j (with weight w_j, release date r_j, and processing time p_j) satisfies the inequality:*

$$C_j \leq r_j + p_j + \sum_{j' \neq j : C_{j'} \geq r_j} \alpha \cdot \min\left(p_{j'}, (w_{j'}/w_j) \cdot p_j\right)$$

To understand the above definition, assume for a while that all the jobs have unit weight, and note that the total *delay* encountered by a job j is equal to $C_j - (r_j + p_j)$. What should be a reasonable upper bound on the contribution (say $\eta_{j'}$) of some specific job $j' \neq j$ towards this delay $C_j - (r_j + p_j)$? Without any loss of generality, we can assume that the job j' completes after the release date of the job j, i.e. $C_{j'} \geq r_j$; otherwise $\eta_{j'}$ is zero. The α-stretch condition says that $\eta_{j'}$ can be at most α times $\min(p_j, p_{j'})$. Note that the job j' can delay the job j when it gets processed with a higher priority, and this can happen for an amount of time equal to the job j''s size, which is $p_{j'}$. Further, the α-stretch condition requires that the policy is fair to both the jobs and hence the bound $\alpha \cdot \min(p_j, p_{j'})$. Finally, the weights of the jobs are also factored in.

This seemingly simple characterization turns out to be quite powerful. In Appendix B, we describe many scheduling policies that are used in practice, and most of these policies have bounded stretch.

Theorem 1.1. *The scheduling policies – Highest Density First, Highest Residual Density First, Weighted Round Robin, and Weighted Shortest Elapsed Time First – all have stretch $\alpha = 1$.*

The reader may find it helpful to compare Definition 1.1 with the definition of the Weighted Round Robin (WRR) scheduling policy (see Appendix B). An important distinction between the two definitions is that the stretch condition does not specify a scheduling policy but is only concerned with the *final* completion times of the jobs. On the other hand, the scheduling policies with bounded stretch behave similar to WRR, in the sense that each job delays another job by at most α times it own processing length. This provides an intuitive explanation as to why such policies should lead to equilibria with small PoA.

[4] Our notion of stretch is different from the standard definition of stretch used in scheduling literature.

1.2.2 Price of Anarchy Bounds for Coordination Mechanisms

With the above definition of bounded stretch, we prove the following general result.

Theorem 1.2. *Suppose that each machine declares a (possibly different) scheduling policy with stretch α. Then the resulting game has a robust (smooth) price of anarchy of at most*

$$\frac{1+\alpha(\sqrt{\alpha^2+1}+\alpha)}{1-\alpha(\sqrt{\alpha^2+1}-\alpha)} \leq 4\alpha^2+2\alpha = O(\alpha^2)$$

Particularly when $\alpha = 1$, the bound is at most 5.8284, and this holds for many popular scheduling policies.

The interesting aspect of the above result is the analysis. We present two analysis techniques which yield somewhat different bounds - potential functions and dual fitting. Conceptually, our techniques are inspired by the elegant work on online scheduling [7, 1, 15] – the connection being that in both cases, we need to compare the decisions made by the optimal solution (that is non-strategic and omniscient) with the solution that arises due to the execution of the implemented policy. However, the similarity ends there - unlike online algorithms, in a coordination game, a job is selfish and cannot be forced to go to a specific machine and hence equilibrium state cannot be controlled by an algorithm. Moreover, a game can have multiple equilibria, and the analysis should hold for all them simultaneously.

Potential Functions. Our first technique uses a smoothness argument (see Section 2) via a carefully constructed *potential function* (see Section 3). The difficulty in a direct smoothness argument is that we have to compare the execution of two policies that make decisions over time, and these decisions could be very divergent. Note that unlike the case without release dates [10], we cannot write closed form expressions for the completion time induced by specific policies. Instead, we show that the *derivative* of the potential function (w.r.t. time) gives an *instantaneous* smoothness inequality that is easy to compute. We then integrate this inequality over time to derive the final smoothness bound. To our knowledge, this type of approach inspired by online algorithms has not been used before in the context of price of anarchy.

Dual Fitting. (See Section 4.) Consider the optimization problem underlying our game-theoretic framework, where the goal is to minimize the total weighted completion time of the jobs plus their assignment costs. We write a time-indexed LP-relaxation of the problem, similar to the one in [1]. Using the dual of this LP, we bound the price of anarchy of the game induced between the jobs, when every machine declares a scheduling policy satisfying certain natural conditions (see Section 4). The idea is to take any Nash equilibrium of the game, and appropriately charge the disutility of each job to the dual variables in a way such that (a) all the dual constraints are satisfied, and (b) the dual objective is at least η times the total disutility incurred by all the jobs in the Nash equilibrium, for some $\eta \in (0, 1]$. This shows that the price of anarchy is at most $1/\eta$ due to weak duality.

In contrast to a potential function based argument, one apparent drawback of the dual-fitting framework is that for technical reasons we need to impose two restrictions on the allowable class of scheduling policies in addition to the bounded stretch condition. These restrictions, however, are fairly intuitive. We are not aware of any simple, combinatorial scheduling policy with bounded stretch that violates the additional assumptions required in the dual-fitting proof. Further, unlike the potential function analysis, the dual-fitting proof only bounds the price of anarchy of pure, mixed Nash and correlated equilibria, and at present we do not see any way to extend the proof to get a robust (smooth) price of anarchy bound which, in addition to these three solution concepts, also applies to no regret sequences [19].

Nevertheless, we feel the dual-fitting framework is interesting in its own right, as it establishes a connection between the price of anarchy of a coordination mechanism and the LP relaxation of the underlying optimization problem. Further, it yields the following improved theorem:

Theorem 1.3. *The price of anarchy (of pure, mixed, and correlated equilibria) of the Highest Density First scheduling policy (see Appendix B for definition) is at most 4.*

This matches the lower bound [10] known for non-preemptive scheduling policies when there are no release dates and assignment costs.[5] The dual-fitting approach also helps us compare against an optimal migratory solution where a job is processed over multiple machines. To our knowledge, this type of approach to bound the price of anarchy of games has not been considered previously in the literature.

Scheduling Policies with Large Stretch. As our α-stretch condition is fairly general and most of the popular scheduling policies have bounded stretch, the reader may be tempted to conjecture that all scheduling policies that are $O(1)$-approximation on a single machine lead to coordination mechanisms with $O(1)$ price of anarchy. However, we show in Appendix E that this is not the case. The scheduling policy Weighted Latest Arrival Processor Sharing (WLAPS) gives $O(1)$-approximation to the total weighted completion time on a single machine. But it does not induce a game with constant price of anarchy. The scheduling policy WLAPS generalizes Round Robin to favor more recent jobs, and has been extensively studied in scheduling theory – particularly in broadcast scheduling problems [12, 3]. Not surprisingly, WLAPS fails our bounded stretch condition (α is $\Omega(n)$). Thus, our characterization seems to separate scheduling policies which are good in non-strategic settings from those which lead to small price of anarchy .

1.2.3 Extension to ℓ_k-norms

Next, we extend our result to the more general objective of minimizing ℓ_k-norms of completion time where $1 \leq k < \infty$. Note that the total completion time is the ℓ_1-norm and the maximum completion time (makespan) is the ℓ_∞-norm. The ℓ_1-norm objective may starve some jobs for an unacceptably long time while focusing too much on average performance. In contrast, the ℓ_∞-norm tries to be as fair as possible at the expense of average performance. The ℓ_k-norms, $k \geq 2$ make a natural balance between average performance and fairness [4, 14, 1]. We prove the following theorem in Appendix H.

Theorem 1.4. *Suppose that each machine declares a (possibly different) scheduling policy with stretch α. Then the resulting game has a robust price of anarchy of $O(k\alpha^{k+1})$ for minimizing the ℓ_k-norm of completion time of jobs, for any $k \geq 1$.*

We present our result only for the unweighted jobs case, and will extend our result to the ℓ_k-norm of weighted completion time in the full version of the paper. We note that previous work [9] considered only SHORTEST JOB FIRST (SJF) and ROUND ROBIN (RR) policies without release dates, and showed that they have robust price of anarchy of $O(k)$ and $O(2^k)$, respectively.[6] Theorem 1.4 implies an exponential improvement for ROUND ROBIN (which has stretch $\alpha = 1$), and this holds even with release dates. As observed in [9], due to a well-known relationship between ℓ_k-norms and the ℓ_∞-norm we have the following theorem as a corollary.

Theorem 1.5. *Suppose that each machine declares a (possibly different) strongly local scheduling policy with stretch $\alpha = 1$. Then the resulting game has a robust price of anarchy of $O(\log n)$ w.r.t. the objective of minimizing the makespan, where n is the number of jobs.*

The above theorem compliments the lower bound of $\Omega(m)$ known for the price of anarchy of strongly local scheduling policies [2], where m is the number of machines (see Section 1.3).

[5] Note that Highest Density First is a nonpreemptive policy in the absence of release dates.

[6] In [9], the authors present the analysis for unweighted ℓ_k-norms of completion time, and claim that their analysis extends to weighted ℓ_k norms. We remind the reader that SJF is the unweighted version of HDF.

1.2.4 Non-preemptive Scheduling Policies

In some applications, preempting a job can be costly, and non-preemptive scheduling policies are highly desirable. When the jobs arrive online, however, no natural non-preemptive scheduling policy gives $O(1)$ approximation to the weighted completion time, even on a single machine. This is particularly relevant since both our potential function and dual-fitting proofs are inspired by the frameworks developed for online scheduling problems [7, 1]. Nevertheless, in Appendix F we present a general black box reduction from preemptive scheduling policies to non-preemptive scheduling policies that preserve the stretch within a factor of two. This reduction is an adaptation of the idea used in [13].

Theorem 1.6. *There exists a reduction that takes any preemptive scheduling policy with stretch α and outputs a non-preemptive scheduling policy with stretch at most 2α.*

The above theorem, along with Theorem 1.1 and Theorem 1.2, shows how to construct non-preemptive (offline) scheduling policies that lead to small constant price of anarchy.

1.2.5 Pure Nash Equilibrium

Although a correlated equilibrium (which can be computed in polynomial time) and a mixed Nash equilibrium is guaranteed to exist in every finite game, not all games have pure Nash equilibrium (PNE). For example, when there are no communication delays (release dates) and assignment costs, it is known that the game induced by the Highest Density First policy might not have a PNE [10]. In contrast, a PNE is guaranteed to exist in the game induced by the Weighted Round Robin (WRR) policy (see Appendix B for definitions). It is not clear if this property of WRR continues to hold in the presence of release dates and assignment costs. We address this concern by transforming the WRR policy. The idea is to run the WRR schedule but withhold the completion time of a job, forcing it to satisfy the following condition.

$$w_j C_j = \sum_{j'} \min(w_j p_{j'}, w_{j'} p_j)$$

One way to achieve this is to process the last ϵ portion of job j at time $t = (1/w_j) \cdot \sum_{j'} \min(w_j p_{j'}, w_{j'} p_j)$. It is easy to verify that the stretch of the resulting schedule is at most 2. The proof for WRR in [10] can be easily extended to show that this induces a *potential game*. Hence, a PNE is guaranteed to exist, and the Nash dynamics converges to a PNE in pseudo-polynomial time.

1.3 Related Work

There has been a lot of work on approximation algorithms for minimizing the weighted sum of completion times [13, 20, 21]. Our potential function and dual-fitting techniques are inspired by the elegant framework developed for online scheduling in [7, 1]. We note that their framework can be adapted to yield combinatorial scheduling policies for weighted completion time that are also $O(1)$-approximations. Here, each machine simply schedules the set of jobs assigned to it using, for example, the Highest Residual Density First policy (see Appendix B for definition). The algorithm considers the jobs in increasing order of their release dates and applies a *greedy dispatch* rule, assigning a job j to that machine i which increases the overall objective function (for the currently dispatched jobs) by the least amount. Although our potential function and dual-fitting proofs are inspired by this framework, as mentioned above, the settings are actually very different. For instance, there is a scheduling policy (see Section E) that gives $O(1)$-approximation to the online optimization problem when used in conjunction with the greedy dispatch rule, but induces a game with very large price of anarchy.

Coordination mechanisms were first introduced in [8]. See the survey [16] for various selfish scheduling models. The study of coordination mechanisms for completion time objective was initiated in [10, 11]. In the absence of release dates and assignment costs, they show tight constant factor price of anarchy bounds for three specific policies - WEIGHTED ROUND ROBIN (WRR), HIGHEST DENSITY FIRST (HDF), and RANDOM (RAND). They also show that, both WRR and RAND induce pure Nash Equilibrium while HDF does not. The l_k-norms of the completion time were considered [9]. They prove a price of anarchy of $O(k)$ for SHORTEST JOB FIRST (when there are no weights, release dates, and assignment costs), and show that no strongly local deterministic policy can achieve a price of anarchy better than $O(k/\log\log k)$.

Azar et al [2] design coordination mechanisms for the makespan objective. They show a lower bound of $\Omega(m)$ for any strongly local scheduling policy (see Section 1.1 for deifinition), where m is the number of machines. In contrast, they present a weakly local scheduling policy (where a machine knows everything about the jobs assigned to it, including their processing lengths on other machines) that achieves a price of anarchy of $O(\log m)$, and a policy that induces a pure Nash Equilibrium with PoA of $O(\log^2 m)$. These results were later extended by Caragiannis [6]. Similar to our work, he showed a strong connection between coordination mechanisms and online algorithms [5]. It will be interesting to study whether this is purely coincidental, or if there is a deeper connection between price of anarchy and competitive ratios.

2 Preliminaries

Recall the concepts and notations introduced in Section 1.1. We index a machine by $i \in \mathcal{M}$, and a job by $j \in \mathcal{J}$. Each machine $i \in \mathcal{M}$ declares a strongly local scheduling policy A_i. Let the symbol $A = (A_1, \ldots A_i, \ldots A_{|\mathcal{M}|})$ denote the profile of scheduling policies. Let GAME(A) denote the resulting game induced between the jobs. An outcome of this game is a strategy-profile $\theta = (\theta_1, \ldots \theta_j, \ldots \theta_{|\mathcal{J}|})$, where $\theta_j \in \mathcal{M}$ is the machine selected by the job $j \in \mathcal{J}$. For notational convenience, we also define an *assignment-vector* Q that summarizes the outcome from the perspective of the machines. The vector $Q = (Q_1, \ldots Q_i, \ldots Q_{|\mathcal{M}|})$ has $|\mathcal{M}|$ components, and the i^{th} component of this vector refers to the set of jobs assigned to machine $i \in \mathcal{M}$. Thus, we have $Q_i = \{j \in \mathcal{J} : \theta_j = i\}$. The completion time of a job j under this outcome is given by $C_j^A(\theta)$. The disutility of the job equals its assignment cost plus its weighted completion time, and this is denoted by $\texttt{COST}_j^A(\theta) = h_{\theta_j,j} + w_j \cdot C_j^A(\theta)$. The outcome is a pure Nash equilibrium iff no job can reduce its disutility by switching to another machine, i.e., $\texttt{COST}_j^A(\theta) \leq \texttt{COST}_j^A(i, \theta_{-j})$ for all $j \in \mathcal{J}, i \in \mathcal{M}$, where θ_{-j} is the strategy-profile of all the jobs except the job j.

We reserve the term *scenario* for a triple $S = (A, Q, \theta)$. This specifies the scheduling policy followed by every machine, and an outcome of the resulting game. The symbol $p_{ij}(t)$ denotes the remaining processing length of a job $j \in Q_i$ on machine i at time t. We say that the job is *unfinished* at time t iff $p_{ij}(t) > 0$. The symbol $W_i(t)$ denotes the total weight of the unfinished jobs on machine i at time t.

Fix any profile of scheduling policies A, and let NE(A) be the set of all pure Nash equilibria of GAME(A). The objective is to minimize the total disutility of the jobs. The price of anarchy of this game is:

$$\text{PoA}(A) = \frac{\max_{\theta \in \text{NE}(A)} \sum_j \texttt{COST}_j^A(\theta)}{\min_{A', \theta'} \sum_j \texttt{COST}_j^{A'}(\theta')}$$

Note that the above expression compares the worst Nash equilibrium of GAME(A) with the optimal solution to the underlying optimization problem, which may use an entirely different profile of scheduling policies.

The notion of price of anarchy as defined above is not applicable in games that do not admit any pure Nash equilibrium. To address this issue, Roughgarden [19] introduced a *smoothness framework* which gives *robust* price of anarchy bounds for generalized solution concepts such as mixed Nash and correlated equilibria and no regret sequences. Adapting the smoothness framework to our context, we say that the

GAME(A) is (λ, μ)-smooth iff the following inequality holds for any two scenarios $S = (A, Q, \theta)$ and $S' = (A', Q', \theta')$.

$$\sum_{j \in \mathcal{J}} \texttt{COST}_j^A(\theta'_j, \theta_{-j}) \leq \lambda \cdot \sum_{j \in \mathcal{J}} \texttt{COST}_j^{A'}(\theta') + \mu \cdot \sum_{j \in \mathcal{J}} \texttt{COST}_j^A(\theta) \tag{1}$$

The reader may find it helpful to think of the scenario S as a pure Nash equilibrium of GAME(A), and the scenario S' as an optimal solution to the underlying optimization problem. It can be shown that the robust price of anarchy of any (λ, μ)-smooth game is at most $\lambda/(1-\mu)$.

3 Robust Price of Anarchy Bound via Potential Function

We devote this entire section to proving Theorem 1.2 by a potential function based argument. Throughout this section, we fix two scenarios $S = (A, Q, \theta)$ and $S' = (A', Q', \theta')$. Further, we assume that all the scheduling polices A_i (specified by A) under the scenario S has stretch α (see Definition 1.1). We will prove Equation 1. The quantities λ and μ will be decided later depending on α.

Zero Assignment Costs. For ease of analysis, we ignore the jobs' assignment costs throughout this section, i.e. $h_{ij} = 0$. It is straightforward to extend our analysis to incorporate nonnegative assignment costs.[7]

Note that the left hand side of Equation 1 results from mixing two completely different scenarios S and S'. Hence, it is not easy to relate this quantity with the right hand side, which consists of the weighted completion times under the two individual scenarios. This is the case particularly when the jobs are released over time, as it becomes extremely difficult to derive mathematical expressions for the terms in Equation 1.

To circumvent this difficulty, we upper bound the left hand side by a carefully chosen potential function, and consider its derivative with respect to time. The advantage of this approach is that we have a good understanding of how an algorithm makes instantaneous scheduling decision. For example, the instantaneous increase in the total weighted completion time is simply the total weight of the unfinished jobs. In other words, each unfinished job j incurs a penalty of w_j at each time step.

For a technical reason that will become clear as we proceed with the proof, we slow down the schedule under S' by a (suitably chosen) constant factor $\delta \in (0, 1)$. Let $A'(\delta)$ denote the new profile of scheduling policies, and let $S'(\delta) = (A'(\delta), Q', \theta')$ denote the new resulting scenario. More precisely, a job is processed on machine i at time t in S' iff it is processed on the same machine i at time t/δ in $S'(\delta)$. Note that the assignment vector and the strategy-profile remain the same across the two scenarios S' and $S'(\delta)$.

We emphasize that $S'(\delta)$ is *not* the schedule that results from the machines executing the policies A' with speed δ. Rather, $S'(\delta)$ is obtained by *stretching out* the schedule S by a factor of $1/\delta$ over the time horizon. It is easy to see that this transformation increases the completion time of a job by a factor of $1/\delta$.

Fact 3.1. *The completion time of every job j' under $S'(\delta)$ is exactly $1/\delta$ times its completion time under S'.*

Overview of our approach. Recall the notations introduced in Section 2. We will always index the jobs by j under the scenario S, and by j' under the scenario $S'(\delta)$. The remaining processing lengths will be denoted by $p_{ij}(t)$ under the scenario S, and by $p^*_{ij'}(t)$ under the scenario $S'(\delta)$. Similarly, the total weight of the unfinished jobs on a machine will be denoted by $W_i(t)$ under the scenario S, and by $W^*_i(t)$ under the scenario $S'(\delta)$. With these notations in place, we are now ready to define our potential function.

$$\Phi(t) = \sum_{i \in \mathcal{M}} \sum_{j' \in Q'_i} \sum_{j \in Q_i} \min\left(w_{j'} \cdot p_{ij}(t), w_j \cdot p^*_{ij'}(t)\right) \tag{2}$$

[7] The dual fitting proof in Section 4 is presented in its full generality, and does not require this simplifying assumption.

Let $\texttt{COST}^S = \sum_j \texttt{COST}^A_j(\theta)$ denote the total disutility of all the jobs under the scenario $S = (A, Q, \theta)$, which is the same as their total weighted completion time (assuming zero assignment costs). Hence, we can write this quantity as $\texttt{COST}^S = \int_0^\infty \frac{\mathrm{d}}{\mathrm{d}t}\texttt{COST}^S(t)$, where the derivative $\frac{\mathrm{d}}{\mathrm{d}t}\texttt{COST}^S(t)$ equals the total weight of the unfinished jobs at time t under the scenario S. Similarly, let $\texttt{COST}^{S'}$ (resp. $\texttt{COST}^{S'(\delta)}$) denote the total disutility of all the jobs under the scenario S' (resp. $S'(\delta)$). Fact 3.1 implies that $\texttt{COST}^{S'(\delta)} = (1/\delta) \cdot \texttt{COST}^{S'}$. We will show that $\Phi(t)$ is a good estimate of the left hand side of Equation 1. In particular, we will prove that $\Phi(t)$ satisfies the following conditions.

$$\Phi(\infty) = 0 \tag{3}$$

$$\sum_{i \in \mathcal{M}} \sum_{j' \in Q'_i} \texttt{COST}^A_{j'}(i, \theta_{-j'}) \leq \texttt{COST}^{S'} + \alpha \cdot \Phi(0) \tag{4}$$

$$-\frac{\mathrm{d}}{\mathrm{d}t}\Phi(t) \leq \frac{\mathrm{d}}{\mathrm{d}t}\texttt{COST}^{S'(\delta)}(t) + \delta \cdot \frac{\mathrm{d}}{\mathrm{d}t}\texttt{COST}^{S}(t) \qquad \text{at every time } t \tag{5}$$

In Equation 4, the symbol α denotes the stretch of the scheduling policies declared by the machines (see Definition 1.1) under the scenario S. The proof of the next theorem appears in Appendix C.

Theorem 3.1. *The potential function as defined in Equation 2 satisfies Equations 3, 4, 5.*

We now use Theorem 3.1 to derive:

$$\begin{aligned}
\text{Left hand side of Equation 1} &= \sum_{i \in \mathcal{M}} \sum_{j' \in Q'_i} \texttt{COST}^A_{j'}(i, \theta_{-j'}) \\
&\leq \texttt{COST}^{S'} + \alpha \cdot \Phi(0) \\
&= \texttt{COST}^{S'} - \alpha \cdot \int_{t=0}^{\infty} \frac{\mathrm{d}}{\mathrm{d}t}\Phi(t)\mathrm{d}t \\
&\leq \texttt{COST}^{S'} + \alpha \cdot \int_{t=0}^{\infty} \Big[\frac{\mathrm{d}}{\mathrm{d}t}\texttt{COST}^{S'(\delta)}(t) + \delta \cdot \frac{\mathrm{d}}{\mathrm{d}t}\texttt{COST}^{S}(t)\Big]\mathrm{d}t \\
&\leq \texttt{COST}^{S'} + \alpha \cdot \texttt{COST}^{S'(\delta)} + (\alpha\delta) \cdot \texttt{COST}^{S} \\
&= (1 + \alpha/\delta) \cdot \texttt{COST}^{S'} + (\alpha\delta) \cdot \texttt{COST}^{S}
\end{aligned}$$

The last equality follows from Fact 3.1.

Thus, setting $\lambda = (1+\alpha/\delta)$ and $\mu = \alpha\delta$, we get a robust price of anarchy bound of $(1+\alpha/\delta)/(1-\alpha\delta)$. This leads to Theorem 1.2. More specifically, by setting $\delta = 1/(2\alpha)$, we obtain a robust PoA bound of $2(1 + 2\alpha^2)$. The best bound we can obtain here is $\frac{1+\alpha(\sqrt{\alpha^2+1}+\alpha)}{1-\alpha(\sqrt{\alpha^2+1}-\alpha)}$ when $\delta = \sqrt{\alpha^2+1} - \alpha$. This bound becomes $(\sqrt{2}+1)/(\sqrt{2}-1) \simeq 5.8284$ when $\alpha = 1$.

Remark. When all the release dates are zero, the robust price of anarchy bound improves to $4\alpha^2$. This improvement follows from a better bound for Equation 4, namely, we can show that its left hand side is at most $\alpha \cdot \Phi(0)$. Note that this bound is tight [10] when $\alpha = 1$ (see Theorem 1.1).

4 Price of Anarchy using Dual Fitting

In this section, we improve the bound on the PoA of α-stretch scheduling policies to 4α using dual fitting. For simplicity of exposition, we only derive the PoA of pure Nash equilibria, and defer the extension to mixed Nash and correlated equilibria to Appendix G. We require that the scheduling policies satisfy two properties in addition to the bounded stretch condition.

Definition 4.1 (Myopic Policy)**.** *A scheduling policy is myopic iff its scheduling decision depends only on the status of the jobs available for processing at the present time instant. In particular, the decision is independent of the jobs that will be released in future.*

Assume for a while that all the jobs have unit weights, and consider a machine which follows the SRPT scheduling policy. At time t, this machine looks at the set of jobs available in its queue, and works on the job j with shortest remaining processing time $p_j(t)$. At time $t+1$, it repeats the same process. This policy is myopic, since its scheduling decision depends only on the jobs currently available in the machine's queue. The reader may find it helpful to keep this example in mind while going through the rest of this section.

Definition 4.2 (Monotone Policy)**.** *A scheduling policy is monotone iff it satisfies three properties. (1) Everything else remaining the same, the completion time of a job can never increase if it is released at an earlier date. (2) Everything else remaining the same, the completion time of a job j can never decrease if the machine is asked to process an extra job j'. (3) For any two jobs j and j' with $w_j = w_{j'}$, $r_j \leq r_{j'}$, and $p_j \leq p_{j'}$, the completion time of job j is at most the completion time of job j', i.e, $C_j \leq C_{j'}$.*

All scheduling policies that give $O(1)$-approximations on a single machine (see Appendix B), with the exception of WLAPS, are myopic and monotone, and have stretch $\alpha = 1$. We prove the following theorem.

Theorem 4.1. *Suppose that each machine declares a (possibly different) scheduling policy which is myopic, monotone and has stretch $\alpha \geq 1$. Then the price of anarchy of the induced game is at most 4α.*

First we derive a bound on the completion time of a job. The next lemma (whose proof appears in Appendix D) justifies our use of the term "bounded stretch", for traditionally the "stretch" of a job is defined as its completion time minus its release date divided by its processing length.

Lemma 4.2. *If a machine runs a scheduling policy with stretch $\alpha \geq 1$, then a job j (with weight w_j, release date r_j, processing length p_j, and completion time C_j) on the machine satisfies the following condition.*

$$C_j \leq r_j + \alpha \cdot (W(r_j)/w_j) \cdot p_j.$$

Here, the symbol $W(r_j)$ denotes the total weight of the unfinished jobs at time t.

LP-relaxation. Consider the LP PRIMAL described below [1]. It has a variable x_{ijt} for each machine $i \in \mathcal{M}$, each job $j \in \mathcal{J}$ and each unit time-slot $t \geq r_{ij}$. If the machine i processes the job j during the whole time-slot t, then this variable is set to 1. The first constraint says that every job has to be completely processed. The second constraint says that a machine cannot process more than one unit of the jobs during any time-slot. Note that the LP allows a job to be processed simultaneously across different machines.

In the objective function, the term $\sum_i \sum_{t \geq r_{ij}} h_{ij} \cdot (x_{ijt}/p_{ij})$ gives the assignment cost incurred by the job j. The term $\sum_i \sum_{t \geq r_{ij}} w_j \cdot x_{ijt} \cdot (t/p_{ij})$ is known as the *fractional weighted completion time* of the job j. In a feasible schedule this quantity is no more than its integral weighted completion time, minus half of its weighted processing time. Finally, the remaining term $\sum_i \sum_{t \geq r_{ij}} w_j \cdot x_{ijt} \cdot (1/2)$ equals half of the weighted processing time of the job. Thus, adding up these three terms, we see that the disutility of a job j is at least $\sum_i \sum_{t \geq r_{ij}} h_{ij} \cdot (x_{ijt}/p_{ij}) + \sum_i \sum_{t \geq r_{ij}} w_j \cdot x_{ijt} \cdot (t/p_{ij} + 1/2)$. Hence, the linear program PRIMAL is a valid relaxation of our problem.

$$\text{Min} \quad \sum_j \sum_i \sum_{t \geq r_{ij}} h_{ij} \cdot (x_{ijt}/p_{ij}) + \sum_j \sum_i \sum_{t \geq r_{ij}} w_j \cdot x_{ijt} \cdot (t/p_{ij} + 1/2) \qquad \text{(PRIMAL)}$$

$$\begin{array}{rcll} \sum_i \sum_{t \geq r_{ij}} x_{ijt}/p_{ij} & \geq & 1 & \forall j \\ \sum_{j\,:\,t \geq r_{ij}} x_{ijt} & \leq & 1 & \forall i, t \\ x_{ijt} & \geq & 0 & \forall i, j, t \,:\, t \geq r_{ij} \end{array}$$

Now, suppose that we constrain each machine to run at a reduced speed of $1/2\alpha$. In other words, each machine can process at most $1/2\alpha$ units of the jobs during one unit of time. It is easy to see that the modified LP described below is a valid relaxation under this new constraint. This transformation increases the objective by at most a factor of 2α.

$$\text{Min} \quad \sum_j \sum_i \sum_{t \geq r_{ij}} h_{ij} \cdot (x_{ijt}/p_{ij}) + \sum_j \sum_i \sum_{t \geq r_{ij}} w_j \cdot x_{ijt} \cdot (t/p_{ij} + \alpha) \qquad \text{PRIMAL}(\alpha)$$

$$\begin{array}{rcll} \sum_i \sum_{t \geq r_{ij}} x_{ijt}/p_{ij} & \geq & 1 & \forall j \\ \sum_{j\,:\,t \geq r_{ij}} x_{ijt} & \leq & 1/(2\alpha) & \forall i, t \\ x_{ijt} & \geq & 0 & \forall i, j, t \,:\, t \geq r_{ij} \end{array}$$

Lemma 4.3. *The optimal objective of the linear program* PRIMAL(α) *is at most* 2α *times the total disutility of the jobs in any feasible integral schedule.*

Finally, we write down the dual of the above linear program.

$$\text{Max} \quad \sum_j y_j - \sum_i \sum_t z_{it} \qquad \text{DUAL}(\alpha)$$

$$\begin{array}{rcll} y_j & \leq & h_{ij} + w_j \cdot (t + \alpha \cdot p_{ij} + (2\alpha) \cdot (z_{it}/w_j) \cdot p_{ij}) & \forall i, j, t \,:\, t \geq r_{ij} \\ y_j & \geq & 0 & \forall j \\ z_{it} & \geq & 0 & \forall i, t \end{array}$$

Analysis. Fix a profile of scheduling policies $A = (A_1, \ldots, A_{|\mathcal{M}|})$, where A_i gives the scheduling policy declared by the machine i. For all $i \in \mathcal{M}$, the scheduling policy A_i is myopic, monotone and has a stretch α. Recall the notations introduced in Section 2. For the rest of this section, we focus on any scenario $S = (A, Q, \theta)$ where the strategy-profile θ is a pure Nash equilibrium of GAME(A).

We will set the variables of the linear program DUAL(α) so as to get a feasible dual solution, with an objective that is at least $1/2$ times the total disutility of the jobs under the scenario S. This, combined with Lemma 4.3 and weak duality, will imply that the price of anarchy of GAME(A) is at most 4α.

Setting the dual variables: The variable y_j is set to be the disutility of the job j under the scenario S. Further, the variable z_{it} is set to be half of the total weight of the unfinished jobs on machine i at time t, under the scenario S.

$$y_j = \texttt{COST}_j^A(\theta) = h_{\theta_j, j} + w_j \cdot C_j^A(\theta) \tag{6}$$

$$z_{it} = (1/2) \cdot W_i(t) \tag{7}$$

Lemma 4.4. *If the dual variables are set as in the equations 6, 7, then the objective of the linear program* DUAL(α) *is at least* $(1/2) \cdot \sum_j \texttt{COST}_j^A(\theta)$.

Proof. We use the well-known fact that the total weighted completion time of all the jobs assigned to any specific machine i is equal to $\sum_t W_i(t)$. Thus, we infer that:

$$\sum_{i,t} z_{it} = (1/2) \cdot \sum_j w_j \cdot C_j^A(\theta) \leq (1/2) \cdot \sum_j \texttt{COST}_j^A(\theta).$$

The lemma follows from the above inequality and the fact that $\sum_j y_j = \sum_j \texttt{COST}_j^A(\theta)$. □

The proof of the next lemma appears in Appendix D.

Lemma 4.5. *If the dual variables are set as in the equations 6, 7, then all the constraints of the linear program* DUAL(α) *are satisfied.*

Theorem 4.1 now follows from Lemma 4.3, Lemma 4.4, and Lemma 4.5.

References

[1] S. Anand, N. Garg, and A. Kumar. Resource augmentation for weighted flow-time explained by dual fitting. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '12, pages 1228–1241. SIAM, 2012.

[2] Y. Azar, K. Jain, and V. Mirrokni. (almost) optimal coordination mechanisms for unrelated machine scheduling. In *Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA '08, pages 323–332, Philadelphia, PA, USA, 2008. Society for Industrial and Applied Mathematics.

[3] N. Bansal, R. Krishnaswamy, and V. Nagarajan. Better scalable algorithms for broadcast scheduling. In *Proceedings of the 37th international colloquium conference on Automata, languages and programming*, ICALP'10, pages 324–335, Berlin, Heidelberg, 2010. Springer-Verlag.

[4] N. Bansal and K. R. Pruhs. Server scheduling to balance priorities, fairness, and average quality of service. *SIAM J. Comput.*, 39(7):3311–3335, July 2010.

[5] I. Caragiannis. Better bounds for online load balancing on unrelated machines. In *SODA*, pages 972–981, 2008.

[6] I. Caragiannis. Efficient coordination mechanisms for unrelated machine scheduling. In *SODA*, pages 815–824, 2009.

[7] J. S. Chadha, N. Garg, A. Kumar, and V. N. Muralidhara. A competitive algorithm for minimizing weighted flow time on unrelatedmachines with speed augmentation. In *STOC*, 2009.

[8] G. Christodoulou, E. Koutsoupias, and A. Nanavati. Coordination mechanisms. *Theor. Comput. Sci.*, 410(36):3327–3336, Aug. 2009.

[9] J. Cohen, C. Dürr, and N. K. Thang. Smooth inequalities and equilibrium inefficiency in scheduling games. In *WINE*, pages 350–363, 2012.

[10] R. Cole, J. R. Correa, V. Gkatzelis, V. Mirrokni, and N. Olver. Inner product spaces for minsum coordination mechanisms. In *Proceedings of the 43rd annual ACM symposium on Theory of computing*, STOC '11, pages 539–548, New York, NY, USA, 2011. ACM.

[11] J. R. Correa and M. Queyranne. Efficiency of equilibria in restricted uniform machine scheduling with total weighted completion time as social cost. In *Naval Research Logistics*, pages 384–395, 2012.

[12] J. Edmonds and K. Pruhs. Scalably scheduling processes with arbitrary speedup curves. In *SODA*, pages 685–692, 2009.

[13] L. A. Hall, D. B. Shmoys, and J. Wein. Scheduling to minimize average completion time: Off-line and on-line algorithms. In *SODA*, 1996.

[14] S. Im and B. Moseley. Online scalable algorithm for minimizing ;k-norms of weighted flow time on unrelated machines. In *SODA*, pages 95–108, 2011.

[15] S. Im, B. Moseley, and K. Pruhs. A tutorial on amortized local competitiveness in online scheduling. *SIGACT News*, 42(2):83–97, 2011.

[16] N. Immorlica, L. E. Li, V. S. Mirrokni, and A. S. Schulz. Coordination mechanisms for selfish scheduling. *Theor. Comput. Sci.*, 410(17):1589–1598, Apr. 2009.

[17] B. Kalyanasundaram and K. Pruhs. Speed is as powerful as clairvoyance. *J. ACM*, 47(4):617–643, July 2000.

[18] E. Koutsoupias and C. H. Papadimitriou. Worst-case equilibria. In *STACS*, pages 404–413, 1999.

[19] T. Roughgarden. Intrinsic robustness of the price of anarchy. In *STOC*, pages 513–522, 2009.

[20] A. S. Schulz and M. Skutella. Scheduling unrelated machines by randomized rounding. *SIAM J. Discret. Math.*, 15(4):450–469, Apr. 2002.

[21] M. Skutella. Convex quadratic and semidefinite programming relaxations in scheduling. *J. ACM*, 48(2):206–242, Mar. 2001.

[22] M. Skutella and G. J. Woeginger. A ptas for minimizing the total weighted completion time on identical parallel machines, 1999.

A List of Symbols

α **:** Stretch of a scheduling policy.

$\mathcal{J}$ **:** Set of jobs.

$\mathcal{M}$ **:** Set of machines.

p_{ij} **:** Processing length of job $j \in \mathcal{J}$ on machine $i \in \mathcal{M}$.

r_{ij} **:** Communication delay (or release date) of job $j \in \mathcal{J}$ on machine $i \in \mathcal{M}$.

h_{ij} **:** Assignment cost of job $j \in \mathcal{J}$ on machine $i \in \mathcal{I}$.

w_j **:** Weight of job $j \in \mathcal{J}$.

A **:** This symbol denotes the profile of scheduling policies followed by the machines. This is a vector $(A_1, \ldots, A_{|\mathcal{M}|})$, where A_i denotes the scheduling policy of machine $i \in \mathcal{M}$.

GAME(A) **:** The game induced between the jobs by the profile of scheduling policies A.

θ **:** This denotes a strategy-profile $(\theta_1, \ldots, \theta_{|\mathcal{J}|})$, where $\theta_j \in \mathcal{M}$ is the machine selected by the job $j \in \mathcal{J}$.

Q **:** This is an assignment-vector $(Q_1, \ldots, Q_{|\mathcal{M}|})$, where $Q_i = \{j \in \mathcal{J} : \theta_j = i\}$ is the subset of jobs assigned to the machine $i \in \mathcal{M}$.

S **:** A *scenario* S is characterized by a triple $S = (A, Q, \theta)$. This specifies the profile of scheduling policies and an outcome of the induced game.

$p_{ij}(t)$ **:** The remaining processing length of job $j \in Q_i$ at time t.

$W_i(t)$ **:** At time t, the total weight of the unfinished jobs on machine i, i.e., $W_i(t) = \sum_{j:p_{ij}(t)>0} w_j$.

$C_j^A(\theta)$ **:** Completion time of job $j \in \mathcal{J}$ under scenario $S = (A, Q, \theta)$.

$\texttt{COST}_j^A(\theta)$: Disutility of job $j \in \mathcal{J}$ under scenario $S = (A, Q, \theta)$, i.e., $\texttt{COST}_j^A(\theta) = h_{\theta_j,j} + w_j \cdot C_j^A(\theta)$.

$Q_i(t)$: At time t, the set of unfinished jobs on machine i, i.e., $Q_i(t) = \{j \in Q_i : p_{ij}(t) > 0\}$.

$J_i(t)$: At time t, the set of unfinished jobs on machine i that are available for processing. Thus, we have:

$$J_i(t) = \{j \in Q_i : p_{ij}(t) > 0 \text{ and } r_{ij} \leq t\}$$

B Some Specific Scheduling Policies

Here, we describe some well known scheduling policies. Fix a machine i which has to process a set of jobs Q_i. The symbol $J_i(t)$ denotes the set of unfinished jobs that are available for processing at time t. All the policies described below are simple, in the sense that they can be implemented in an "online" environment. Thus, at any time t, the processing decision depends only on the jobs in $J_i(t)$, their weights, and remaining processing lengths. In particular, the decision is independent of the jobs that are going to arrive in future.

First we give a brief description of scheduling policies which has bounded stretch (see Definition 1.1).

HIGHEST DENSITY FIRST (HDF). At any time t, the machine works on the job $j \in J_i(t)$ which has the highest density w_j/p_{ij}. When the jobs are unweighted, this policy is known as "Shortest Processing Time First" (SPT).

HIGHEST RESIDUAL DENSITY FIRST (HRDF). At any time t, the machine works on the job $j \in J_i(t)$ which has the highest residual density $w_j/p_{ij}(t)$. When the jobs are unweighted, this policy is known as "Shortest Remaining Processing Time First" (SRPT).

WEIGHTED ROUND ROBIN (WRR). At any time t, the machine works on the jobs in $J_i(t)$ in proportion to their weights. Specifically, for all jobs $j \in J_i(t)$, we have:

$$\frac{d}{dt}\left(p_{ij}(t)\right) = -\frac{w_j}{\sum_{j' \in J_i(t)} w_{j'}}.$$

We note that WRR is an example of a *non-clairvoyant* scheduling policy. This refers to the property that a scheduling policy like WRR can be used even when the machine does not known the processing lengths in advance, and a job's processing length is revealed only when the machine finishes the job.

WEIGHTED SHORTEST ELAPSED TIME FIRST (WSETF). At any time t, the machine works on the job $j \in J_i(t)$ which maximizes the ratio $w_j/(p_{ij} - p_{ij}(t))$. Unweighted version of this scheduling policy is known as "Shortest Elapsed Time First", and here the machine works on the job which has been least processed so far. WSETF is a non-clairvoyant scheduling policy which has been extensively studied in scheduling literature [17, 4].

It is easy to verify that all the scheduling policies mentioned above satisfy the stretch condition with $\alpha = 1$, and are myopic and monotone. We skip the proof.

Next, we look at scheduling policies which do not satisfy our bounded stretch conditon. First we consider WLAPS, which gives $O(1)$ approximation on a single machine and has been extensively studied in scheduling theory [12, 3]. We show in Section E that WLAPS has $\alpha = \Omega(n)$, and it also has PoA $= \Omega(n)$.

WEIGHTED LATEST ARRIVAL PROCESSOR SHARING (WLAPS(ϵ)). This scheduling policy takes a parameter $\epsilon \in [0, 1]$ as input. Let $J_i^\epsilon(t)$ denote $\epsilon|J(t)|$ jobs in $J(t)$ with the highest release dates. At any time t, the machine works on the jobs $j \in J_i^\epsilon(t)$ in proportion to their weights, so that for all $j \in J_i^\epsilon(t)$ we have:

$$\frac{d}{dt}\left(p_{ij}(t)\right) = -\frac{w_j}{\sum_{j' \in J_i^\epsilon(t)} w_{j'}}.$$

This scheduling policy gives preference to the recently released jobs, and is non-clairvoyant. Note that if $\epsilon = 1$, then this policy reduces to WEIGHTED ROUND ROBIN. When the jobs are unweighted, this policy is known as LATEST ARRIVAL PROCESSOR SHARING (LAPS).

The following scheduling policies also fail to satisfy our bounded stretch condition.

- First in First Out (FIFO) – this schedules the jobs in the increasing order of their release dates.
- Longest Job First (LJF) – this schedules the longest job available for processing.
- Biggest Weight First (BWF) – this schedules the job with the highest weight.
- The policy which schedules a job uniformly at random.

However, unlike WLAPS, these polices are not $O(1)$ approximation to the weighted completion objective and hence have bad PoA.

Remark: The scheduling policy RAND considered in [10] does indeed satisfy our condition with $\alpha = 1$ in expectation. However, we do not analyze RAND, since it is not clear how to extend RAND in the presence of release dates.

C Proof of Theorem 3.1

Let $\Phi_i(t)$ denote the contribution towards $\Phi(t)$ by machine $i \in \mathcal{M}$, and let $\Phi_{ij'}(t)$ denote the contribution towards $\Phi_i(t)$ by job $j' \in Q_i'$. Thus, we have:

$$\Phi_{ij'}(t) = \sum_{j \in Q_i} \min\left(w_{j'} \cdot p_{ij}(t), w_j \cdot p^*_{ij'}(t)\right) \tag{8}$$

$$\Phi_i(t) = \sum_{j' \in Q_i'} \Phi_{ij'}(t) \tag{9}$$

$$\Phi(t) = \sum_{i \in \mathcal{M}} \Phi_i(t) \tag{10}$$

The next two lemmas show that $\Phi(t)$ satisfies the boundary conditions at $t = 0$ and $t = \infty$.

Lemma C.1. *The potential function $\Phi(t)$ satisfies Equation 3.*

Proof. Follows from the observation that each job has zero remaining processing length at time $t = \infty$. □

Lemma C.2. *The potential function $\Phi(t)$ satisfies Equation 4.*

Proof. Fix any machine $i \in \mathcal{M}$. For every job $j' \in Q'_i$, we have:

$$\begin{aligned} w_{j'} \cdot C^A_{j'}(i, \theta_{-j'}) &\leq w_{j'} \cdot r_{ij'} + w_{j'} \cdot p_{ij'} + \alpha \cdot \sum_{j \in Q_i} \min\left(w_{j'} \cdot p_{ij}, w_j \cdot p_{ij'}\right) \\ &= w_{j'} \cdot (r_{ij'} + p_{ij'}) + \alpha \cdot \sum_{j \in Q_i} \min\left(w_{j'} \cdot p_{ij}(0), w_j \cdot p^*_{ij'}(0)\right) \\ &= w_{j'} \cdot (r_{ij'} + p_{ij'}) + \alpha \cdot \Phi_{ij'}(0) \\ &\leq w_{j'} \cdot C^{A'}_{j'}(\theta') + \alpha \cdot \Phi_{ij'}(0) \end{aligned} \tag{11}$$

The first inequality holds since the scheduling policy A_i has stretch α (see Definition 1.1). The last inequality holds since $r_{ij'} + p_{ij'}$ is at most the completion time of the job j' under any feasible schedule. Finally, note that $w_{j'} \cdot C^A_{j'}(i, \theta_{-j'}) = \texttt{COST}^A_{j'}(i, \theta_{-j'})$ in the absence of assignment costs. So the lemma follows when we sum both sides of Equation 11 over all machines $i \in \mathcal{M}$ and jobs $j' \in Q'_i$. □

It remains to show that $\Phi(t)$ satisfies Equation 5. We will first make some simple observations.

Fact C.1. *The functions $p_{ij}(t), p^*_{ij'}(t)$, and $\Phi_{ij'}(t)$ are all continuous and non-increasing in t.*

The following facts hold since the machines operate at speed δ (resp. 1) under scenario $S'(\delta)$ (resp. S).

Fact C.2. *Fix any machine $i \in \mathcal{M}$, and any two jobs $j' \in Q'_i$ and $j \in Q_i$. At any time t, we have:*

$$0 \geq \frac{d}{dt}\Big(p^*_{ij'}(t)\Big) \geq -\delta, \quad \textit{and} \quad 0 \geq \frac{d}{dt}\Big(p_{ij}(t)\Big) \geq -1.$$

Fact C.3. *At any time t, on any machine $i \in \mathcal{M}$, we have:*

$$0 \geq \frac{d}{dt}\Big(\sum_{j' \in Q'_i} p^*_{ij'}(t)\Big) \geq -\delta, \quad \textit{and} \quad 0 \geq \frac{d}{dt}\Big(\sum_{j \in Q_i} p_{ij}(t)\Big) \geq -1.$$

We now bound the rate of change in $\Phi_i(t)$ due to any unfinished job under the scenario $S'(\delta)$.

Claim C.1. *For every job $j' \in Q'_i$ that completes after time t under the scenario $S'(\delta)$, we have:*

$$\frac{d}{dt}\left(\Phi_{ij'}(t)\right) \geq -w_{j'} + W_i(t) \cdot \frac{d}{dt}\Big(p^*_{ij'}(t)\Big).$$

Proof. Recall that $\Phi_{ij'}(t)$ is a summation over a set of terms, each corresponding to a job $j \in Q_i$ assigned to the same machine i, but under a different scenario S (see Equation 8). Each such term is the minimum of two functions: $w_{j'} \cdot p_{ij}(t)$ and $w_j \cdot p^*_{ij'}(t)$. We partition all the jobs in Q_i into two subsets Y and Z, depending on which of the two functions attain the minimum value:

$$Y = \{j \in Q_i \,:\, w_{j'} \cdot p_{ij}(t) \leq w_j \cdot p^*_{ij'}(t)\}, \text{ and } Z = \{j \in Q_i \,:\, w_{j'} \cdot p_{ij}(t) > w_j \cdot p^*_{ij'}(t)\}.$$

The functions $f^Y(t), f^Z(t)$ capture the respective contributions of the subsets Y and Z towards $\Phi_{ij'}(t)$.

$$f^Y(t) = w_{j'} \cdot \sum_{j \in Y} p_{ij}(t), \text{ and } f^Z(t) = \Big(\sum_{j \in Z} w_j\Big) \cdot p^*_{ij'}(t).$$

Note that $\Phi_{ij'}(t) = f^Y(t) + f^Z(t)$. Further, since the job j' completes after time t under the scenario $S'(\delta)$, we have $p^*_{ij'}(t) > 0$. It follows that every job $j \in Z$ has $p_{ij}(t) > 0$. In other words, every job in Z completes after time t under the scenario S, which implies that $\sum_{j \in Z} w_j \leq W_i(t)$. Hence, we conclude:

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\left(\Phi_{ij'}(t)\right) &= \frac{\mathrm{d}}{\mathrm{d}t} f^Y(t) + \frac{\mathrm{d}}{\mathrm{d}t} f^Z(t) \\
&= w_{j'} \cdot \frac{\mathrm{d}}{\mathrm{d}t} \sum_{j \in Y} p_{ij}(t) + \Big(\sum_{j \in Z} w_j\Big) \cdot \frac{\mathrm{d}}{\mathrm{d}t}\left(p^*_{ij'}(t)\right) \\
&\geq -w_{j'} + \Big(\sum_{j \in Z} w_j\Big) \cdot \frac{\mathrm{d}}{\mathrm{d}t}\left(p^*_{ij'}(t)\right) \qquad (12) \\
&\geq -w_{j'} + W_i(t) \cdot \frac{\mathrm{d}}{\mathrm{d}t}\left(p^*_{ij'}(t)\right) \qquad (13)
\end{aligned}$$

Equation 12 follows from Fact C.3. Equation 13 holds since $\sum_{j \in Z} w_j \leq W_i(t)$ and $\frac{\mathrm{d}}{\mathrm{d}t}(p^*_{ij'}(t)) \leq 0$. □

The next claim shows that we can ignore the jobs that finishes before time t under the scenario $S'(\delta)$.

Claim C.2. *For every job $j' \in Q'_i$ that completes before time t under the scenario $S'(\delta)$, we have:*

$$\frac{d}{dt}\left(\Phi_{ij'}(t)\right) = 0.$$

Proof. Follows from the observation that such a job j' has $p^*_{ij'}(t') = 0$ for all $t' \geq t$. □

The next claim bounds the overall rate of change of $\Phi_i(t)$.

Claim C.3. *For any machine $i \in \mathcal{M}$ and any time t, we have:*

$$\frac{d}{dt}\left(\Phi_i(t)\right) \geq -W^*_i(t) - \delta \cdot W_i(t).$$

Proof. We infer that:

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\Phi_i(t) &= \sum_{j' \in Q'_i} \frac{\mathrm{d}}{\mathrm{d}t}\Phi_{ij'}(t) \\
&\geq -W^*_i(t) + W_i(t) \cdot \sum_{j' \in Q'_i} \frac{\mathrm{d}}{\mathrm{d}t}\left(p^*_{ij'}(t)\right) \qquad (14) \\
&\geq -W^*_i(t) - \delta \cdot W_i(t) \qquad (15)
\end{aligned}$$

Equation 14 follows from Claim C.1 and Claim C.2. Equation 15 follows from Fact C.3. □

Now we are ready to bound the overall rate of change of $\Phi(t)$.

Lemma C.3. *The potential function $\Phi(t)$ satisfies Equation 5.*

Proof. Follows from summing both sides of the inequality in Claim C.3 over all machines $i \in \mathcal{M}$, and recalling that $\frac{\mathrm{d}}{\mathrm{d}t}\mathtt{COST}^{S'(\delta)}(t) = \sum_i W^*_i(t)$, and $\frac{\mathrm{d}}{\mathrm{d}t}\mathtt{COST}^{S}(t) = \sum_i W_i(t)$. □

Theorem 3.1 follows from Lemma C.1, Lemma C.2, and Lemma C.3.

D Missing Proofs from Section 4

D.1 Proof of Lemma 4.2

Since the scheduling policy has stretch $\alpha \geq 1$, Definition 1.1 implies that:

$$\begin{aligned} C_j &\leq r_j + p_j + \sum_{j' \neq j : C_{j'} \geq r_j} \alpha \cdot \min(p_{j'}, (w_{j'}/w_j) \cdot p_j) \\ &\leq r_j + \sum_{j' : C_{j'} \geq r_j} \alpha \cdot (w_{j'}/w_j) \cdot p_j \\ &= r_j + \alpha \cdot (W(r_j)/w_j) \cdot p_j \end{aligned}$$

D.2 Proof of Lemma 4.5

Clearly, the dual variables are set to nonnegative values. For the rest of the proof, we fix a job j, a machine i, and a time $t \geq r_{ij}$, and show that the corresponding dual constraint is satisfied.

A Thought Experiment. We create a job j' with $p_{ij'} = p_{ij}$, $w_{j'} = w_j$, and $r_{ij'} = t$. The machine i is now asked to process the set of jobs $Q_i \cup \{j'\}$ using the scheduling policy A_i. Under this thought experiment, let $C^*_{j'}$ denote the completion time of the job j'. Recall that A_i is a myopic scheduling policy (Definition 4.1). Hence, under this thought experiment, the total weight of the unfinished jobs on machine i at time t is exactly $W_i(t) + w_{j'}$. Since the policy A_i also has stretch α, Lemma 4.2 gives:

$$C^*_{j'} \leq t + \alpha \cdot \left(\frac{W_i(t) + w_{j'}}{w_{j'}} \right) \cdot p_{ij'} = t + \alpha \cdot p_{ij'} + \alpha \cdot \left(W_i(t)/w_{j'} \right) \cdot p_{ij'}$$

Plugging in the equalities $W_i(t) = 2z_{it}$, $w_{j'} = w_j$, and $p_{ij'} = p_{ij}$, we get:

$$C^*_{j'} \leq t + \alpha \cdot p_{ij} + (2\alpha) \cdot (z_{it}/w_j) \cdot p_{ij} \tag{16}$$

We now consider two possible cases.

Case 1: Job j selects machine i under the outcome S, so that $\theta_j = i$.

Let C^*_j denote the completion time of the job j under the thought experiment. Recall that the machine follows a monotone scheduling policy. Hence, part 3 of Definition 4.2 implies that $C^*_j \leq C^*_{j'}$. Further, part 2 of Definition 4.2 implies that $C^A_j(\theta) \leq C^*_j$. Accordingly, we have $C^A_j(\theta) \leq C^*_{j'}$. Since $y_j = h_{ij} + w_j \cdot C^A_j(\theta)$, we infer that:

$$y_j \leq h_{ij} + w_j \cdot C^*_{j'} \tag{17}$$

Equation 16 and Equation 17 imply that the dual constraint is satisfied.

Case 2: Job j does not select machine i under the scenario S, so that $\theta_j \neq i$.

Consider the scenario S. We want to bound the completion time of the job j when it switches to the machine i, and everything else remains the same. This is denoted by $C^A_j(i, \theta_{-j})$. This also corresponds to a thought experiment, where the machine i is asked to schedule the jobs in $Q_i \cup \{j\}$ using the scheduling policy A_i. The only difference between this thought experiment and the previous one is that here the job being inserted

has an earlier release date ($r_{ij} \leq r_{ij'} = t$, $w_j = w_{j'}$, $p_{ij} = p_{ij'}$). Accordingly, part 1 of Definition 4.2 implies that $C_j^A(i, \theta_{-j}) \leq C_{j'}^*$. Since $\texttt{COST}_j^A(i, \theta_{-j}) = h_{ij} + w_j \cdot C_j^A(i, \theta_{-j})$, we get:

$$\texttt{COST}_j^A(i, \theta_{-j}) \leq h_{ij} + w_j \cdot C_{j'}^* \tag{18}$$

Finally, recall that the strategy-profile θ is a pure Nash equilibrium of $\textsc{Game}(A)$. Hence, we have:

$$y_j = \texttt{COST}_j^A(\theta) \leq \texttt{COST}_j^A(i, \theta_{-j}) \tag{19}$$

Equation 16, Equation 18, and Equation 19 imply that the dual constraint is satisfied.

E Price of Anarchy of a Scheduling Policy with Large Stretch

In this section, we consider a policy called Weighted Latest Arrival Processor Sharing (WLAPS) which has been extensively studied in scheduling theory [12, 3]. Although this policy gives $O(1)$-approximation to weighted completion time on a single machine, we show that its induced game has large PoA (see Lemma E.1). Further, we show that WLAPS fails our α-stretch condition (see Lemma E.2), hence making a case for bounded stretch policies.

Fix a machine i which has to process a set of jobs, and let $J_i(t)$ denote the set of unfinished jobs that are available for processing at time t. The scheduling policy WLAPS takes a parameter $\epsilon \in [0, 1]$ as input. Let $J_i^\epsilon(t)$ denote the $\epsilon|J_i(t)|$ jobs in $J_i(t)$ with the highest release dates. At any time t, the machine works on the jobs $j \in J_i^\epsilon(t)$ in proportion to their weights, so that for all $j \in J_i^\epsilon(t)$ we have:

$$\frac{d}{dt}\left(p_{ij}(t)\right) = -\frac{w_j}{\sum_{j' \in J_i^\epsilon(t)} w_{j'}}.$$

WLAPS is an example of a *non-clairvoyant* scheduling policy, and reduces to Weighted Round Robin (see Appendix B) when $\epsilon = 1$. For simplicity, we fix $\epsilon = 1/2$ in the following proof; however, the proof can be easily extended to any value of ϵ.

Lemma E.1. *If every machine follows the scheduling policy WLAPS(ϵ) with $\epsilon = 1/2$, then the PoA of the resulting game is $\Omega(n)$, where n denotes the total number of jobs.*

Proof. Consider the following instance. There are n jobs and n machines. Each job has unit weight. Among the n jobs, there is one *big* job j^* which has processing length $p_{ij^*} = n$ and release date $r_{ij^*} = \kappa$ on all the machines. Here, κ is an arbitrarily small positive value. The remaining $n - 1$ *small* jobs have processing lengths $p_{1j} = \kappa + \beta$ (which is an arbitrarily small positive value) on machine $i = 1$, and processing length $p_{ij} = n$ on the other machines $i \neq 1$. Further, these small jobs have release dates $r_{ij} = 0$ on all the machines.

In an optimal solution, all the small jobs are assigned to machine 1, and the big job is assigned to any other machine. Hence, the sum of the completion times of the jobs is at most $O(n)$. In contrast, there exists a bad equilibrium - the big job on machine 1, and each small job on a distinct machine. Here, the total completion time of all the jobs is at least $\Omega(n^2)$. It is easy to see that this is a Nash Equilibrium. In particular, no small job wants to deviate to machine 1, since that machine would make the small job wait till it finishes the big job.

□

Lemma E.2. *The scheduling policy $WLAPS(\epsilon)$ has stretch $\alpha = \Omega(n)$ when $\epsilon = 1/2$. Here, the symbol n denotes the number of jobs processed by the machine.*

Proof. Consider the following instance. A machine is processing a set of n jobs $X_1 \cup X_2$ using the scheduling policy WLAPS(1/2). Each job $j \in X_1 \cup X_2$ has weight $w_j = 1$. The subset X_1 consists of $n/2$ jobs. Each job $j \in X_1$ has a processing length $p_j = 1$ and release date $r_j = 0$. The subset X_2 also consists of $n/2$ jobs. Each job $j \in X_2$ has a processing length $p_j = n$ and release date $r_j = \kappa$, where κ is an arbitrarily small positive value.

It is easy to see that the machine starves the jobs in X_1 in favor of the jobs in X_2 from time $t = \kappa$ onwards. Hence, all the jobs in X_2 complete at time $t = n^2/2 + \kappa$, and the completion time of any job $j \in X_1$ is at least $\Omega(n^2)$. On the other hand, the bounded stretch condition (see Definition 1.1) requires the completion time of such a job to be at most αn. Thus, WLAPS(ϵ) has stretch $\alpha = \Omega(n)$ when $\epsilon = 1/2$. □

F Proof of Theorem 1.6

We show how to transform any preemptive policy into a non-preemptive policy in which the completion time of a job increases at most by a factor 2. This ensures that if the preemptive policy had a stretch α to begin with, then the resulting non-preemptive policy has a stretch 2α. This implies Theorem 1.6. Our transformation is similar in spirit to the one used in [13].

Consider any α-stretch scheduling policy A_i declared by machine i. Recall that Q_i denotes the set of jobs assigned to machine i. Let C_j be completion time of job j in the schedule produced by policy A_i on the set Q_i. Renumber the jobs in Q_i such that $C_{j-1} < C_j$. We modify the preemptive schedule produced by A_i into a non-preemptive schedule in the following manner.

- Consider the jobs in $j \in Q_i$ in increasing order of their completion time values C_j. Schedule the jobs non-preemptively in this order.

Let $\overline{C_j}$ denote the completion time of job j in this new schedule. The following theorem shows that, the completion time every job in the new schedule increases at most by a factor of 2.

Theorem F.1. $\overline{C_j} \leq 2C_j$

Proof. Fix a job j. The completion time of the job j in the non-preemptive schedule can be bounded by:

$$\overline{C_j} \leq \max_{k \in [j]} \{r_{ik}\} + \sum_{k \in [j]} p_{ik}$$

Recall that we renumbered the jobs in the set Q_i in the increasing order of C_j values. The above inequality holds since there is no idle period after the job with highest release date in the set $[1 \dots j]$ is released.

The proof of the theorem follows from the observations that $C_j \geq \max_{k \in [j]} r_{ik}$ and $C_j \geq \sum_{k \in [j]} p_{ik}$. □

G PoA for mixed Nash Equilibrium and Correlated Equilibrium

In this section, we give the complete proof of PoA bound for mixed Nash equilibrium using dual fitting. Reader can verify that, the same proof extends to Correlated Equilibrium. As a first step, we derive a (slightly) different bound on the completion time of a job when a scheduling policy satisfies the α-stretch (Def 1.1),is myopic (Def 4.1) and monotone (Def 4.2). Recall that for a scheduling policy A and an input set Q of jobs, C_j denotes the completion time of job $j \in Q$; $Q(t)$ denotes the set of unsatisfied jobs at time r_j and $W(t)$ denotes the total weight of unsatisfied jobs at time t.

Lemma G.1. *If a scheduling policy A which is myopic,monotone and has a stretch α, then the following inequalities hold for all input sets of jobs Q.*

$$C_j \leq r_j + \alpha \cdot \left(\frac{W(r_j)}{w_j}\right) \cdot p_j \quad \textit{for all } j \in Q.$$

Proof. Recall the definition of an α-stretch policy. We know from Definition 1.1,

$$\begin{aligned} C_j^A &\leq r_j + \alpha \cdot \left(\sum_{j' \in Q(r_j)} \min(p_{j'}, (w_{j'}/w_j) \cdot p_j)\right) \\ &\leq r_j + \alpha \cdot \left(\sum_{j' \in Q(r_j)} (w_{j'}/w_j) \cdot p_j\right) \\ &\leq r_j + \alpha \cdot W(r_j)/w_j \cdot p_j \end{aligned}$$

□

Consider the game induced by a profile of scheduling policies $A = (A_1, \ldots, A_{|\mathcal{M}|})$, where A_i is the scheduling policy followed by the machine i. Furthermore, for all $i \in \mathcal{M}$, the scheduling policy A_i is myopic, monotone and has a stretch α. Fix any (mixed) Nash equilibrium of this game and let the corresponding strategy-profile of jobs be denoted by $\theta = (\theta_1, \ldots, \theta_{|\mathcal{J}|})$, where $\theta_j = (\sigma_1, \ldots \sigma_i, \ldots \sigma_{|M|})$ and σ_i is the probability with which machine i selected by the job j. Let (A, θ) be denoted by scenario S.

We will use $E[\textsc{Cost}_j^S]$ to denote the expected total cost incurred by job j under the scenario S which includes the expected assignment cost + weighted expected completion time. For rest of the section, $E[C_j^S]$ and $E[H_j^S]$ will stand for the expected completion time of job j under the scenario S and $E[W_i^S(t)]$ denote the expected total weight of unsatisfied jobs on the machine i at time t.

Setting the dual variables The variable y_j is set to be the total penalty incurred by the job j, under the scenario S. On the other hand, the variable z_{it} is set to be half of the expected total weight of the unfinished jobs on machine i at time t.

$$y_j = E[\textsc{Cost}_j^S] = E[H_j^S] + w_j \cdot E[C_j^S] \tag{20}$$

$$z_{it} = (1/2) \cdot E[W_i^S(t)] \tag{21}$$

Lemma G.2. *If the dual variables are set as in the equations 20, 21, then the objective of the linear program* $\textsc{Dual}(\alpha)$ *is at least* $(1/2) \cdot E[\textsc{Cost}^S]$.

Proof. We invoke a standard characterization of completion time, which says that the total weighted completion time of all the jobs assigned to any specific machine i is equal to $\sum_t W_i^S(t)$. Thus, we infer that:

$$\sum_{i,t} z_{it} = (1/2) \cdot \sum_{j \in \mathcal{J}} w_j \cdot E[C_j^S] \leq (1/2) \cdot E[\textsc{Cost}^S].$$

The lemma follows from the above inequality and the fact that $\sum_j y_j = E[\textsc{Cost}^S]$. □

Lemma G.3. *If the dual variables are set as in the equations 20, 21, then all the constraints of the linear program* $\textsc{Dual}(\alpha)$ *are feasible.*

Proof. Clearly, the dual variables are set to nonnegative values. To complete the proof, we need to show that the dual constraint corresponding to any job j, any machine i, and any time $t \geq r_{ij}$, is satisfied. Let $E[\textsc{Cost}_j^S(i, \theta_{-j})]$ denote the expected cost incurred by job j if it unilaterally deviates to machine i, everything else remaining the same. Similarly, let $E[C_j^S(i, \theta_{-j})]$ denotes the expected completion time of job j if it deviates to machine i. Since, $S = (A, \theta)$ is in Nash equilibrium, we have

$$E[\textsc{Cost}_j^S] \leq h_{ij} + w_j \cdot E[C_j^S(i, \theta_{-j})] \tag{22}$$

To show that all the dual constraints are satisfied, fix a job j, machine i and some time instant $t \geq r_{ij}$.

Consider the following thought experiment. We create a job j^* with $p_{ij^*} = p_{ij}$, $w_{j^*} = w_j$, and $r_{ij^*} = t$. Machine i is asked to process this job $j*$. In otherwords, we take the scenario $S = (A, \theta)$ which is in Nash equilibrium and modify it into a new scenario $S' = (A, \theta')$ as follows. $\theta' = \theta \cup \theta_{j*}$ and $\theta_{j*} = (0, \ldots, \sigma_i = 1, \ldots 0)$. Let $E[C_{j^*}]$ denote the expected completion time of the job j^*. Since A_i is a myopic scheduling policy (Definition 4.1), under this thought experiment the expected total weight of the unfinished jobs on machine i at time t is exactly $E[W_i^S(t)] + w_{j^*}$.

Since the policy A_i also has stretch α, simple extension of Lemma G.1 gives:

$$E[C_{j^*}] \leq t + \alpha \cdot \left(\frac{E[W_i^S(t)] + w_{j^*}}{w_{j^*}} \right) \cdot p_{ij^*}$$

Plugging in the equalities $E[W_i^S(t)] = 2z_{it}$, $w_{j^*} = w_j$, and $p_{ij^*} = p_{ij}$, we get:

$$E[C_{j^*}] \leq t + \alpha \cdot (1 + 2z_{it}/w_j) \cdot p_{ij} \tag{23}$$

To complete the proof, we note that since A_i is a myopic and monotone scheduling policy,

$$E[C_j^S(i, \theta_{-j})] \leq E[C_{j^*}] \tag{24}$$

Therefore from Equations 22, 23 and 24 we have,

$$E[\textsc{Cost}_j^S] \leq h_{ij} + E[C_j^S(i, \theta_{-j})] \leq h_{ij} + w_j \left(t + \alpha \cdot (1 + 2z_{it}/w_j) \cdot p_{ij}\right) \tag{25}$$

Since $y_j = E[\textsc{Cost}_j^S]$ we get,

$$y_j \leq h_{ij} + w_j \left(t + \alpha \cdot p_{ij} + (2\alpha) \cdot (z_{it}/w_j) \cdot p_{ij}\right)$$

Therefore, dual constraints are feasible. This completes the proof. □

From Theorem 4.3 and Lemma G.2, it follows that PoA of α-stretch scheduling policies which are monotone and myopic is at most 4α.

H ℓ_k-norms of Completion Time

In this section, we extend our result for total completion time to the objective of ℓ_k-norms of complete time. The ℓ_k-norms of complete time of all jobs is defined as $(\sum_j (C_j - r_{\theta_j,j})^k)^{1/k}$ when job j is assigned to machine θ_j and is completed at time C_j. The kth power of completion time is defined as $\sum_j (C_j - r_{\theta_j,j})^k$. For the sake of analysis, we will first bound the smoothness of the game for the kth power of completion time, and will take the kth root at the end of analysis. The notation will remain the same unless specifically stated. We will use $\texttt{COST}$ to refer to kth power of completion time. For example, $\texttt{COST}^S$ denotes the total

kth power of completion time of all jobs under the schedule (scenario) S. We let $\texttt{COST}_j^{S,S'}$ denote the kth power of completion time of job j when the job j is scheduled on machine θ'_j with jobs $Q_{\theta'_j}$ by the scheduling policy $A_{\theta'_j}$. More intuitively, $\texttt{COST}_j^{S,S'}$ is job j's kth power of completion time when it moves to the machine on which S' scheduled the job. This section is devoted to proving the following theorem.

Theorem H.1. *For any integer $k \geq 2$, consider the objective of minimizing kth power of completion time. Also consider any two scenarios $S = (A, Q, \theta)$ and $S' = (A', Q', \theta')$ where all scheduling policies A_i (specified by A) have stretch α. Then it follows that*

$$\sum_{j \in \mathcal{J}} \mathit{COST}_j^{S,S'} \leq \lambda \cdot \mathit{COST}^S + \mu \cdot \mathit{COST}^{S'},$$

where $\lambda = O(\alpha^{k^2+k} \cdot (36k)^{k^2+k})$ and $\mu = 1/2$.

Hence we derive an upper bound of $O(\alpha^{k^2+k} \cdot (36k)^{k-1})$ on the robust PoA of the game for kth poet of completion time. By taking the kth root on this bound[8], we obtain an upper bound of $O(\alpha^{k+1}k)$ for the ℓ_k-norms of completion time, thereby proving Theorem 1.4 and 1.5. We note that there is almost a tight lower bound of $\Omega(k/\log k)$ on the price of anarchy [9].

H.1 Overview of the Analysis

In this section we give an overview of the analysis. We define the following potential function. Consider any two scenarios $S = (A, Q, \theta)$ and $S' = (A', Q', \theta')$.Suppose that in S, every machine is running at a reduced speed of $\delta \in [0, 1]$. This situation shall be denoted by $S(\delta)$. More precisely, a job j is processed in S at time t if and only if the same job j is processed in $S(\delta)$ at time t/δ; both Q and θ are the same in S and $S(\delta)$.

For all machines $i \in \mathcal{M}$, time t, and $j \in \mathbf{Q}^{S(\delta)}$, define :

$$\Phi_{ij}^{S(\delta),S'}(t) := \Big(\sum_{j' \in Q'} \min(p_{ij}^{S(\delta)}(t), p_{ij'}^{S'}(t) \Big)^k$$

For all machines $i \in \mathcal{M}$, define:

$$\Phi_i^{S(\delta),S'}(t) := \sum_{j \in Q_i} \Phi_{i,j}^{S(\delta),S'}(t).$$

Our final potential function will be:

$$\Phi^{S(\delta),S'}(t) := \sum_{i \in \mathcal{M}} \Phi_i^{S(\delta),S'}(t).$$

Assuming that all algorithms in A' have a stretch of at most α, we have,

$$\begin{aligned}
\texttt{COST}^{S,S'} &\leq \sum_{i \in \mathcal{M}} \sum_{j \in Q_i} \Big(r_{ij} + p_{ij} + \alpha \cdot \sum_{j' \in Q'_i} \min\left(p_{ij}, p_{ij'}\right) \Big)^k \\
&\leq (k+1)^k \sum_{i \in \mathcal{M}} \sum_{j \in Q_i} \Big(r_{ij} + p_{ij} \Big)^k + 3\alpha^k \sum_{i \in \mathcal{M}} \sum_{j \in Q_i} \Big(\sum_{j' \in Q'_i} \min\left(p_{ij}, p_{ij'}\right) \Big)^k \\
&\leq (k+1)^k \texttt{COST}^S + 3\alpha^k \Phi^{S(\delta),S'}(0) \qquad (26)
\end{aligned}$$

[8]Note that this can be done since each job behaves in the same way for both objectives. For mixed and correlated Nash equilibria, this is not the case, and it should be assumed that each job's goal is to minimize its expected kth power of completion time, not its expected completion time.

The second inequality follows from a simple inequality that $\forall x, y \geq 0,\ \ (x+y)^k \leq (k+1)^k x^k + (1+1/k)^k y^k \leq (k+1)^k x^k + 3y^k$. The last inequality follows from the fact that $(r_{ij} + p_{ij})$ is a lower bound on the completion time of job j in any schedule where job j is assigned to machine i (which is the case in the assignment Q).

Hence our analysis will be focused on bounding $\Phi^{S(\delta),S'}(0)$. We will study $\frac{\mathrm{d}}{\mathrm{d}t}\Phi^{S(\delta),S'}(t)$, and to upper bound it, we define a very useful quantity

$$\texttt{COST}^S(t) := \sum_{j:t\leq C_j^S} (C_j^S - t)^k,$$

which will help keep track of how the cost of S changes. Likewise we define $\texttt{COST}^{S(\delta)}(t) := \sum_{j:t\leq C_j^{S(\delta)}} (C_j^S - t)^k$.

Note that $\texttt{COST}^S(0) = \texttt{COST}^S$ and $\texttt{COST}^S(\infty) = 0$. To this end, we show that for any $\delta \in (0,1]$, and $\epsilon \in (0,1)$, we will show that at all times t,

$$\frac{\mathrm{d}}{\mathrm{d}t}\Phi^{S(\delta),S'}(t) \leq -\Big(\frac{\delta^{k-1}}{\epsilon^{k-1}} + \frac{1}{\epsilon^{k-1}}\Big)\frac{\mathrm{d}}{\mathrm{d}t}\texttt{COST}^{S(\delta)}(t) - \Big(2\epsilon^{k-1} + 3\delta k\Big)\frac{\mathrm{d}}{\mathrm{d}t}\texttt{COST}^{S'}(t)$$

By integrating this inequality and combining it with (26), we will have

$$\begin{aligned}\texttt{COST}^{S,S'} &\leq (k+1)^k\texttt{COST}^S + \Big(\frac{3\alpha^k\cdot\delta^{k-1}}{\epsilon^{k-1}} + \frac{3\alpha^k}{\epsilon^{k-1}}\Big)\texttt{COST}^{S(\delta)} + 3\alpha^k\cdot\Big(2\epsilon^{k-1} + \delta k 2^{k+1}\Big)\texttt{COST}^{S'}\\ &\leq \Big((k+1)^k + \frac{3\alpha^k}{\epsilon^{k-1}} + \frac{3\alpha^k}{\epsilon^{k-1}\delta^{k-1}}\Big)\texttt{COST}^S + 3\alpha^k\cdot\Big(2\epsilon^{k-1} + \delta k 2^{k+1}\Big)\texttt{COST}^{S'}\end{aligned}$$

We set $\delta = \frac{1}{36k\alpha^k}$ and $\epsilon^{k-1} = \frac{1}{24\alpha^k}$, and derive:

$$\texttt{COST}^{S,S'} \leq O\Big(\alpha^{k^2+k}\cdot(36k)^{k-1}\Big)\texttt{COST}^S + \frac{1}{2}\texttt{COST}^{S'}$$

This will complete the proof of Theorem H.1.

As mentioned, our remaining task is to upper bound $\frac{\mathrm{d}}{\mathrm{d}t}\Phi^{S(\delta),S'}(t)$. We will first introduce several useful technical lemmas. Then we will proceed our analysis by considering the effect of the processing in $S(\delta)$ and S', separately. Throughout the analysis all ties are broken in an arbitrary but a consistent way. Particularly, the reader may read the analysis assuming that no two different jobs have the same size, nor the same remaining size.

H.2 Technical Lemmas

Lemma H.2. *Consider any single machine and the set Q of jobs scheduled on the machine. Consider any scenario (schedule) S. Then for any constant $\delta \in (0,1]$ and for all times t, it holds that*

$$\sum_{j\in Q} k\Big(\sum_{j'\in Q, p_{j'}^{S(\delta)}(t)\leq p_j^{S(\delta)}(t)} p_{j'}^{S(\delta)}(t)\Big)^{k-1} \leq -\delta^{k-1}\frac{d}{dt}COST^{S(\delta)}(t).$$

Proof. Fix the schedule S and time t. For notational convenience, we will use q_j to denote the remaining size $p_j^{S(\delta)}(t)$ of an alive job j at time t. We reindex jobs such that $0 < q_1 \leq q_2 \leq ... \leq q_\ell$; we can ignore completed jobs. Since the quantity $\frac{\mathrm{d}}{\mathrm{d}t}\texttt{COST}^{S(\delta)}(t)$ in the right-hand-side is defined only by completion times

of jobs, we will need to relate this quantity to the remaining sizes of jobs, $q_{\ell'}$. We claim that processing jobs in the order of shortest remaining sizes gives a lower bound to quantity. More formally,

$$(\frac{1}{\delta^{k-1}}) \sum_{\ell' \in [\ell]} k\bar{q}_{\ell'}^{k-1} \leq \sum_{\ell' \in [\ell]} k(C_{\ell'}^{S(\delta)} - t)^{k-1} \quad = -\frac{\mathrm{d}}{\mathrm{d}t}\mathtt{COST}^{S(\delta)}(t).$$

where $\bar{q}_{\ell'} := q_1 + q_2 + ... + q_{\ell'}$. To restate the claim in other words, we currently have ℓ unsatisfied jobs at time t with remaining sizes $q_{\ell'}$, $\ell' \in [\ell]$. Then we would like to show that in any valid schedule $\sigma(\delta)$ (which does not have to coincide with $S(\delta)$ since time t), $\sum_{\ell' \in [\ell]}(C_{\ell'}^{\sigma(\delta)} - t)^{k-1}$ cannot be smaller than $\sum_{\ell' \in [\ell]}(\frac{1}{\delta^{k-1}})\bar{q}_{\ell'}^{k-1}$. Here we can assume without loss of generality that all jobs $[\ell]$ are available for schedule right now at time t since it can only help minimize $\sum_{\ell' \in [\ell]}(C_{\ell'}^{\sigma(\delta)} - t)^{k-1}$. We can prove the claim by a simple 'swap' argument: Suppose a job $j' > j$ is processed before job j is completed. Then we modify the schedule restricted to the two jobs j and j'. More precisely, we modify the schedule $\sigma(\delta)$ only at the times when j or j' are processed, and simply schedule job j earlier than j' at those times. It is easy to see that this swap operation can only decrease $\sum_{\ell' \in [\ell]}(C_{\ell'}^{\sigma(\delta)} - t)^{k-1}$. Since $S(\delta)$ processes a job at a rate of δ, job ℓ' is completed in time $\frac{1}{\delta}\bar{q}'_{\ell}$. □

Corollary H.3. *Consider any single machine and the set Q of jobs scheduled on the machine. Consider any scenario (schedule) S. Then for any constant $\delta \in (0, 1]$ and for all times t, it holds that*

$$\sum_{j \in Q} k\Big(p_j^{S(\delta)}(t) \sum_{j' \in Q, p_{j'}^{S(\delta)}(t) \geq p_j^{S(\delta)}(t)} 1\Big)^{k-1} \leq -\delta^{k-1}\frac{d}{dt}COST^{S(\delta)}(t).$$

Proof. We borrow the same notation from the proof of Lemma H.2. By expressing the left-hand-side in this Corollary in terms of $q_{\ell'}$, by Lemma H.2, it suffices to show that:

$$\sum_{\ell' \in [\ell]} (q_{\ell'}(\ell - \ell' + 1))^{k-1} \leq \sum_{\ell' \in [\ell]} \bar{q}_{\ell'}^{k-1}$$

We fully expand the right-hand-side equation, and then have a sum of monomials of a unit coefficient. We charge each monomial $q_{\ell'}^{k-1}$ to a unique monomial in the right-hand-side. We will say that a monomial is in class ℓ' if it consists only of $q_{\ell'}, q_{\ell'+1}, ..., q_{\ell}$; class ℓ' is a superset of class $\ell' + 1$. Note that any monomial of degree $k-1$ in class ℓ' is no smaller than $q_{\ell'}^{k-1}$ due to $q_{\ell'}$ being non-decreasing. Note that all monomials appearing in both sides are of degree $k-1$. Observe that the right-hand-side has $1^{k-1} + 2^{k-1} + ... + (\ell - \ell' + 1)^{k-1}$ monomials in class ℓ'. Knowing that there are $(\ell - \ell' + 1)^{k-1}$ monomials of $q_{\ell'}^{k-1}$ in the left-hand-side, it is easy to see that we can charge each $q_{\ell'}^{k-1}$ in the left-hand-side to a unique monomial of class ℓ' in the right-hand-side. Hence the first inequality follows. □

Corollary H.4. *Consider any single machine and the set Q of jobs scheduled on the machine. Consider any scenario (schedule) S. Then for any constant $\delta \in (0, 1]$, any $p' \geq 0$ and for all times t, it holds that,*

$$\Big(\sum_{j' \in Q : p_{j'}^{S(\delta)} \geq p'} 1\Big)^{k} \cdot (p')^{k-1} \leq -\delta^{k-1}\frac{d}{dt}COST^{S(\delta)}(t).$$

Proof. We use the same notation that was used in the proof of Lemma H.2. Let ℓ' be the smallest from $[\ell]$ such that $q^{\ell'} \geq p'$; if no such ℓ' exists, the corollary immediately follows. Then the left-hand-side can be expressed as $(\ell - \ell' + 1)^k q_{\ell'}^{k-1}$. By Lemma H.2, it suffices to show that,

$$(\ell - \ell' + 1)^k q_{\ell'}^{k-1} \leq k \sum_{\ell' \in [\ell]} \bar{q}_{\ell'}^{k-1}$$

Similar to the proof of Corollary H.3, we will charge $(\ell - \ell' + 1)^k$ copies of the monomial $q_{\ell'}^{k-1}$ to the monomials in class ℓ'. Observe that we can find $1^{k-1} + 2^{k-1} + ... + (\ell - \ell' + 1)^{k-1}$ monomials of class ℓ' in the right-hand-side, and it can be shown by at least $\int_0^{\ell-\ell'+1} h^{k-1} \mathrm{d}h \geq (\ell - \ell' + 1)^k$. □

H.3 Changes of $\Phi^{S(\delta),S'}(t)$

We will study the changes of $\Phi^{S(\delta),S'}(t)$ over time t. Note that $\Phi^{S(\delta),S'}(t)$ can change due to the processing in $S(\delta)$ and S'. We assume without loss of generality that there is a unique job $a_i(t)$ that is being processed at time t in $S(\delta)$'s schedule. Likewise we let $a'_i(t)$ denote the unique job being processed at time t in the schedule of S'. Throughout the analysis we simply assume that such jobs $a_i(t)$ an $a'_i(t)$ exist since otherwise it can only make our analysis easier. When time t is fixed, we may drop (t) and simply use a_i and a'_i. Recall that the job a_i and a'_i can be processed at a rate of at most δ and 1, respectively. When i is clear from the context we may omit i from a_i and a'_i.

For notational convenience, for any $p' \geq 0$, we let $V_{\leq p'}^{S',i}(t) := \sum_{j' \in Q'_i, p_{ij'}^{S'}(t) \leq p'} p_{ij'}^{S'}(t)$ denote the total remaining volume of jobs in Q'_i whose remaining size in S' are smaller than p'. Here the schedule and machine in consideration are specified in the superscript and the extra condition that jobs have to satisfy for consideration is specified in the subscript. In the same spirit, let $N_{\geq p'}^{S',i}(t) := |\{j' \in \mathbf{Q}_i^{S'} \mid p_{ij'}^{S'}(t) \geq p'\}|$. When time t is clear from the context, we amy simply use $V_{\leq p'}^{S',i}$ and $N_{\geq p'}^{S',i}$.

We will without loss of generality assume that jobs in consideration at the current time have distinct remaining sizes by breaking ties in an arbitrary but consistent way. We upper bound the changes of $\Phi^{S(\delta),S'}(t)$ due to the processing of $S(\delta)$'s S', separately.

H.3.1 Changes due to $S(\delta)$'s processing

Fix time t. We will focus on each machine i and bound $\frac{\mathrm{d}}{\mathrm{d}t}\Phi_i^{S(\delta),S'}(t)$. We observe that,

$$\frac{\mathrm{d}}{\mathrm{d}t}\Phi_i^{S(\delta),S'}(t) \quad \leq \quad \delta \cdot k(V_{\leq p_{ia_i}^{S(\delta)}(t)}^{S',i} + N_{\geq p_{ia_i}^{S(\delta)}(t)}^{S',i} \cdot p_{ia_i}^{S(\delta)}(t))^{k-1} \cdot N_{\geq p_{ia_i}^{S(\delta)}(t)}^{S',i}$$

This follows by observing that $\min(p_{ij}^{S(\delta)}(t), p_{ij'}^{S'}(t))$ changes only when $j = a$ and $p_{ij'}^{S'}(t) \geq p_{ia}^{S(\delta)}(t)$. Since we are focusing on machine i and considering only the jobs assigned to machine i (either in $S(\delta)$ or S'), we may drop i from the notation when the machine is clear in the context. Particularly we may omit i from V and N's superscript and from p's subscript.

We will show,

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\Phi_i^{S(\delta),S'}(t) \quad &\leq \quad \left(V_{\leq p_a^{S(\delta)}(t)}^{S'} + N_{\geq p_a^{S(\delta)}(t)}^{S'} p_a^{S(\delta)}(t)\right)^{k-1} N_{\geq p_a^{S(\delta)}(t)}^{S'} \\ &\leq \quad 3k \sum_{j \in Q^{S'}} \Big(\sum_{j' \in Q^{S'}, p_{j'}^{S'}(t) \leq p_j^{S'}(t)} p_{j'}^{S'}(t) \Big)^{k-1} \end{aligned} \tag{27}$$

This will imply by Lemma H.2 that,

$$\frac{\mathrm{d}}{\mathrm{d}t}\Phi_i^{S(\delta),S'}(t) \leq -3k\delta \frac{\mathrm{d}}{\mathrm{d}t}\mathtt{COST}_i^{S'}(t) \tag{28}$$

Here $\mathtt{COST}_i^{S'}(t)$ denotes machine i's contribution to the quantity $\mathtt{COST}^{S'}(t)$. We follow the same notation we used in the proof of Corollary H.3, let $q_{\ell'}$, $\ell' \in [\ell]$ denote the remaining sizes $p_{i,\ell'}^{S'}$ of unsatisfied jobs

ℓ' in S' at the current time t. Recall that $q_1 \le q_2 \le ... \le q_\ell$. Also let $\bar{q}_{\ell'} = q_1 + q_2 + ... + q_{\ell'}$. To show the desired inequality it suffices to show the following:

$$\sum_{\ell' \in [\ell]} (\bar{q}_{\ell'-1} + (\ell - \ell' + 1) q_{\ell'})^{k-1} \le \sum_{\ell \in [\ell']} \bar{q}_{\ell'}^{k-1}$$

To show this, for each $h \in [\lceil \ell/k \rceil]$, we charge $\sum_{\ell - kh < \ell' \le \ell - k(h-1)} (\bar{q}_{\ell'-1} + (\ell - \ell' + 1) q_{\ell'})^{k-1}$ to $\bar{q}_{\ell-h+1}^{k-1}$. Here we can show that $(\bar{q}_{\ell'-1} + (\ell - \ell' + 1) q_{\ell'}) \le \frac{k}{k-1} \bar{q}_{\ell-h+1}$. This can by shown by observing that all terms q terms in $\bar{q}_{\ell-h+1} - \bar{q}_{\ell'-1}$ are no smaller than $q_{\ell'}$ and there are at least $\frac{k-1}{k}(\ell - \ell' + 1)$ such terms. Then we derive

$$\sum_{\ell' \in [\ell]} (\bar{q}_{\ell'-1} + (\ell - \ell' + 1) q_{\ell'})^{k-1} \le k (\frac{k-1}{k})^{k-1} \sum_{\ell \in [\ell']} \bar{q}_{\ell'}^{k-1} \le 3k \sum_{\ell \in [\ell']} \bar{q}_{\ell'}^{k-1}.$$

This completes the proof of (28). By summing over all machines i, we derive

$$\frac{\mathrm{d}}{\mathrm{d}t} \Phi^{S(\delta), S'}(t) \Big|_{S(\delta)\text{' processing}} \le 3k\delta \cdot \frac{\mathrm{d}}{\mathrm{d}t} \mathtt{COST}^{S'}(t) \tag{29}$$

H.3.2 Changes due to the processing of S'

This is more challenging than bounding the changes to due to $S(\delta)$'s processing. As before we focus on each machine i, and will drop the notation i or (t) if they are clear from the context. Let a' denote the job that is being processed at the current time t by S'. Recall that

$$\Phi_i^{S(\delta), S'}(t) := \sum_{j \in Q_i(t)} \Big(\sum_{j' \in Q_i'(t)} \min(p_j^{S(\delta)}(t), p_{j'}^{S'}(t)) \Big)^k.$$

For any job $j \in Q_i(t)$, observe that $\sum_{j' \in \mathbf{Q}_i^{S'}(t)} \min(p_j^{S(\delta)}(t), p_{j'}^{S'}(t))$ decreases at a rate of at most 1 since a' can be processed at a rate of at most one. Hence $\frac{\mathrm{d}}{\mathrm{d}t}\Phi_i(t)$ due to S''s processing can be seen to be at most

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t} \Phi_i^{S(\delta), S'}(t) \Big|_{S'\text{'s processing}} &\le \sum_{j \in \mathbf{Q}_i^{S(\delta)}} k \Big(V^{S'}_{\le p_j^{S(\delta)}(t)} + N^{S'}_{\ge p_j^{S(\delta)}(t)} \cdot p_j^{S(\delta)}(t) \Big)^{k-1} \\
&\le 2^{k-1} \sum_{j \in \mathbf{Q}_i^{S(\delta)}} k \Big(V^{S'}_{\le p_j^{S(\delta)}(t)} \Big)^{k-1} \qquad (30) \\
&\quad + 2^{k-1} \sum_{j \in \mathbf{Q}_i^{S(\delta)}} k \Big(N^{S'}_{\ge p_j^{S(\delta)}(t)} \cdot p_j^{S(\delta)}(t) \Big)^{k-1} \qquad (31)
\end{aligned}$$

We bound (30) and (31) separately.

Bounding (30): We partition jobs in Q_i in two groups Q_1 and Q_2: We put all jobs j in Q_i such that $V^{S'}_{\le p_j^{S(\delta)}(t)} \le (1/\epsilon) V^{S(\delta)}_{\le p_j^{S(\delta)}(t)}$, into Q_1. All other jobs are placed in Q_2. We first bound Q_1's contribution to (30). By Lemma H.2,

$$\sum_{j \in Q_1} k \Big(V^{S'}_{\le p_j^S(t)} \Big)^{k-1} \le \sum_{j \in Q_1} k \Big(\frac{1}{\epsilon} V^{S(\delta)}_{\le p_j^S(t)} \Big)^{k-1} \le -\frac{\delta^{k-1}}{\epsilon^{k-1}} \frac{\mathrm{d}}{\mathrm{d}t} \mathtt{COST}(S(\delta), t) \tag{32}$$

Now we bound Q_2's contribution to (30) by only a small fraction of $\frac{\mathrm{d}}{\mathrm{d}t}\mathtt{COST}^{S'}(t)$. To this end we will associate each job $j \in Q_2$ with a distinct job $\pi'(j)$ in Q'_i such that $V^{S'}_{\leq p^{S'}_{\leq p^{i'}(j)}(t)} \geq (1/\epsilon) V^{S(\delta)}_{\leq p^{S(\delta)}_j(t)}$. This is possible due to the definition of Q_2. We will then charge $k(V^{S'}_{\leq p^{S(\delta)}_j(t)})^{k-1}$ to $k(V^{S'}_{\leq p^{S'}_{\pi'(j)}})^{k-1}$. Observe that the first quantity is at most ϵ^{k-1} times the second quantity. By summing over all $j \in Q_2$ and by Lemma H.2, we will have

$$\begin{aligned}\sum_{j\in Q_2} k(V^{S'}_{\leq p^{S(\delta)}_j(t)})^{k-1} &\leq \epsilon^{k-1} \sum_{j\in Q_2} k(V^{S'}_{\leq p^{S'}_{\pi'(j)}})^{k-1} \\ \leq \quad \epsilon^{k-1} \sum_{j'\in \mathbf{Q}^{S'}_i} k(V^{S'}_{\leq p^{S'}_{j'}})^{k-1} &\leq -\epsilon^{k-1} \frac{\mathrm{d}}{\mathrm{d}t}\mathtt{COST}^{S'}(t) \end{aligned} \tag{33}$$

By combining this with (32), we will have

$$(30) \leq -\frac{\delta^{k-1}}{\epsilon^{k-1}} \frac{\mathrm{d}}{\mathrm{d}t}\mathtt{COST}^{S(\delta)}(t) - \epsilon^{k-1}\frac{\mathrm{d}}{\mathrm{d}t}\mathtt{COST}^{S'}(t) \tag{34}$$

It now remains to show the following lemma.

Lemma H.5. *Suppose $0 < \epsilon < 1$. Then there exists a mapping from each $j \in Q_2$ to a distinct job $\pi'(j) \in Q'_i$ such that $V^{S'}_{\leq p^{S'}_{\leq p^{i'}(j)}(t)} \geq (1/\epsilon) V^{S(\delta)}_{\leq p^{S(\delta)}_j(t)}$.*

Proof. To show the lemma, we partition jobs in Q' into disjoint sets Q'^j, $j \in Q_2$. The job j' in Q'^j with the largest remaining size $p^{S'}_{j'}(t)$ will be $\pi'(j)$. For notational convenience, we relabel the unsatisfied jobs in Q_2 as $1, 2, ..., \ell$ such that $p^{S(\delta)}_{l'}(t)$ is non-decreasing in l'. Let q_l denote $p^{S(\delta)}_l(t)$. We consider jobs l in $Q_2 = [\ell]$ in increasing order, i.e. $1, 2, ..., \ell$. Our goal is to associate each job l with a disjoint set of jobs $Q'^l \subseteq Q'^j$. Jobs in Q' will be considered in increasing order of $p^{S'}_{j'}(t)$. We let Q'^l be a minimal set of jobs from Q' that were not associated yet with other l such that $V^{S'}_{\geq q_l} \geq (1/\epsilon)\sum_{h\in[l]} V^{Q'^l}$. Here $V^{Q'^l}$ denotes $\sum_{j'\in Q'^l} p^{S(\delta)}_{j'}$. it follows from definition of Q_2. that for all jobs $j' \in Q'^j$, $p^{S(\delta)}_j(t) \geq p^{S'}_{j'}(t)$.

Now to complete the proof it suffices to show that for all $l \in [\ell]$, $Q'^l \neq \emptyset$. For the sake of contradiction suppose that $Q'^l = \emptyset$ for some l. Since all jobs $j' \in Q'^l$ have remaining size $p^{S'}_{j'}(t)$ smaller than q_l, it must be the case that

$$(1/\epsilon)(q_1 + q_2 + ... + q_l) \leq \sum_{h\in[l]} V^{Q'^l} \leq (1/\epsilon)(q_1 + q_2 + ... + q_l) + q_l,$$

due to the minimality of Q'^l. Similarly, we have

$$(1/\epsilon)(q_1 + q_2 + ... + q_{l-1}) \leq \sum_{h\in[l-1]} V^{Q'^{l-1}} \leq (1/\epsilon)(q_1 + q_2 + ... + q_{l-1}) + q_{l-1}.$$

Since $\sum_{h\in[l]} V^{Q'^l} = \sum_{h\in[l-1]} V^{Q'^{l-1}}$, it must be the case that

$$(1/\epsilon)(q_1 + q_2 + ... + q_l) \leq (1/\epsilon)(q_1 + q_2 + ... + q_{l-1}) + q_{l-1}$$

This simplifies to $q_l \leq \epsilon q_{l-1}$, which is a contradiction. □

Bounding (31): We partition jobs in $Q^{S(\delta)}(t)$ in two groups Q_1 and Q_2: We place all jobs j in $Q^{S(\delta)}(t)$ such that $N^{S'}_{\geq p_j^{S(\delta)}(t)}(t) \leq (1/\epsilon) N^{S(\delta)}_{\geq p_j^S(t)}(t)$, into Q_1. All other jobs are placed in Q_2. We first bound Q_1's contribution to (31).

$$\sum_{j \in Q_1(t)} k\Big(N^{S'}_{\geq p_j^S(t)}(t) \cdot p_j^S(t)\Big)^{k-1} \leq \sum_{j \in Q_1(t)} k\Big(N^{S'}_{\geq p_j^S(t)}(t) \cdot p_j^{S(\delta)}(t)\Big)^{k-1} \leq -\frac{1}{\epsilon^{k-1}} \frac{\mathrm{d}}{\mathrm{d}t} C(S(\delta), t) \tag{35}$$

The last inequality is due to Corollary H.4.

Now we bound Q_2's contribution to (31) by a small fraction of $\frac{\mathrm{d}}{\mathrm{d}t}\mathtt{COST}^{S'}(t)$. Intuitively this is possible since $N^{S'}_{\geq p_j^{S(\delta)}(t)}(t) \geq (1/\epsilon) N^{S(\delta)}_{\geq p_j^S(t)}(t)$ implies that there are more jobs in S' than $S(\delta)$. To establish such a relationship formally, we associate each job j in Q_2 with a distinct job $\pi'(j)$ such that $N^{S'}_{\geq p_{\pi'(j)}^{S(\delta)}(t)} \geq (1/\epsilon) N^{S(\delta)}_{\geq p_j^S(t)}$ and $p^{S'}_{\pi' i'(j)}(t) \geq p_j^{S(\delta)}(t)$. To obtain such a map π', we let $\pi'(j)$ be the job $j' \in Q'$ that minimizes $N^{S'}_{\geq p_{j'}^{S(\delta)}}$, and such that $N^{S'}_{\geq p_{j'}^{S(\delta)}} \geq (1/\epsilon) N^{S(\delta)}_{\geq p_j^S(t)}$. One can show that $\pi'(j)$ are distinct when $0 < \epsilon < 1$.

For each job $j \in Q_2$, we will charge the quantity $k(N^{S'}_{\geq p_j^S(t)}(t) \cdot p_j^S(t))^{k-1}$ to $k(N^{S'}_{\geq p^{S'}_{\pi'(j)}(t)} \cdot p^{S'}_{\pi'(j)}(t))^{k-1}$. Note that the first quantity is no bigger than ϵ^{k-1} times the latter quantity. Hence we have

$$\sum_{j \in Q_2} k(N^{S'}_{\geq p_j^{S(\delta)}(t)}(t) \cdot p_j^S(t))^{k-1} \leq -\epsilon^{k-1} \frac{\mathrm{d}}{\mathrm{d}t} \mathtt{COST}^{S'}(t) \tag{36}$$

Combining this with (35), we obtain

$$(31) \leq -\frac{1}{\epsilon^{k-1}} \frac{\mathrm{d}}{\mathrm{d}t} \mathtt{COST}^{S(\delta)}(t) - \epsilon^{k-1} \frac{\mathrm{d}}{\mathrm{d}t} \mathtt{COST}^{S'}(t) \tag{37}$$

From (30), (31), (34), and (37), by summing over all machines i, we have

$$\frac{\mathrm{d}}{\mathrm{d}t} \Phi^{S(\delta), S'}(t)\Big|_{S'\text{'s processing}} \leq -\Big(\frac{\delta^{k-1}}{\epsilon^{k-1}} + \frac{1}{\epsilon^{k-1}}\Big) \frac{\mathrm{d}}{\mathrm{d}t} \mathtt{COST}^{S(\delta)}(t) - 2\epsilon^{k-1} \frac{\mathrm{d}}{\mathrm{d}t} \mathtt{COST}^{S'}(t) \tag{38}$$