Stochastic Load Balancing on Unrelated Machines*

Anupam Gupta[†] Amit Kumar[‡] Viswanath Nagarajan[§] Xiangkun Shen[§]

July 12, 2017

Abstract

We consider the problem of makespan minimization: i.e., scheduling jobs on machines to minimize the maximum load. For the deterministic case, good approximations are known even when the machines are unrelated. However, the problem is not well-understood when there is uncertainty in the job sizes. In our setting the job sizes are stochastic, i.e., the size of a job j on machine i is a random variable X_{ij}, whose distribution is known. (Sizes of different jobs are independent of each other.) The goal is to find a fixed assignment of jobs to machines, to minimize the expected makespan—i.e., the expected value of the maximum load over the m machines. For the identical machines special case when the size of a job is the same across all machines, a constant-factor approximation algorithm has long been known. However, the problem has remained open even for the next-harder related machines case.

Our main result is a constant-factor approximation for the most general case of unrelated machines. The main technical challenge we overcome is obtaining an efficiently computable lower bound for the optimal solution. We give an exponential-sized LP that we argue gives a strong lower bound. Then we show how to round any fractional solution to satisfy only a small subset of the constraints, which are enough to bound the expected makespan of our solution.

We then consider two generalizations. The first is the *budgeted makespan minimization* problem, where the goal is to minimize the makespan subject to scheduling any subset of jobs whose reward is at least some target reward R. We extend our above result to a constant-factor approximation here using polyhedral properties of the bipartite matching polytope. The second problem is the *q-norm minimization problem*, where we want to minimize the expected ℓ_q-norm of the load vectors. (The case of $q = \infty$ is the makespan minimization case.) Here we give an $O(q/\log q)$-approximation algorithm using a reduction to the deterministic q-norm problem with side constraints.

*This work was done in part while the authors were visiting the Simons Institute for the Theory of Computing.

[†]Computer Science Department, Carnegie Mellon University, Pittsburgh, USA. Supported in part by NSF awards CCF-1536002, CCF-1540541, and CCF-1617790.

[‡]Dept. of Computer Science and Engg., IIT Delhi, India 110016.

[§]Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI 48109.

1 Introduction

We consider the problem of scheduling jobs on machines to minimize the maximum load (i.e., the problem of *makespan minimization*). This is a classic NP-hard problem, with Graham's list scheduling algorithm for the identical machines being one of the earliest approximation algorithms known. If the job sizes are deterministic, the problem is fairly well-understood, with PTASes for the identical [HS87] and related machines cases [HS88], and a constant-factor approximation and APX-hardness [LST90, ST93] for the unrelated machines case. Given we understand the basic problem well, it is natural to consider settings which are less stylized, and one step closer to modeling real-world scenarios: *what can we do if there is uncertainty in the job sizes?*

Our model is this: we are given n jobs and m machines. For each job j, its size on machine i is given by a random variable X_{ij}, whose distribution is known to the algorithm. (In this paper we assume that job sizes for different jobs $j \neq j'$ are independent of each other.) Given just this information, the algorithm has to assign these jobs to machines, say resulting in jobs J_i being assigned to machine i. The expected makespan of this assignment is

$$\mathbb{E}\left[\max_{i=1}^{m} \sum_{j \in J_i} X_{ij}\right]. \tag{1}$$

The goal for the algorithm is to minimize this expected makespan. Observe that the entire assignment of jobs to machines is done up-front without knowledge of the actual outcomes of the random variables, and hence there is no adaptivity in this problem.

Such stochastic load-balancing problems are common in real-world systems where the job sizes are indeed not known, but given the large amounts of data, one can generate reasonable estimates for the distribution. Moreover static (non-adaptive) assignments are preferable in many applications as they are easier to implement. Inspired by work on scheduling and routing problems in several communities, Kleinberg, Rabani, and Tardos first posed the problem of approximating the expected makespan in 1997 [KRT00]. They gave a constant-factor approximation for the identical machines case, i.e., for the case where for each job j, the sizes $X_{ij} = X_{i'j}$ for all $i, i' \in [m]$. (They also gave results for bin-packing and knapsack problems, which will not concern us here.) Goel and Indyk [GI99] gave better results for special classes of job size distributions.

Despite these improvements and refined understanding of the identical machines case, the above stochastic load-balancing problem has remained open, even for related machines setting. Recall that *related machines* refers to the special case where each machine has a *speed* s_i, and the random variables for each job j satisfy $X_{ij}\, s_i = X_{i'j}\, s_{i'}$. A moment's thought reveals the problem with extending the result of [KRT00]: *how do we give a lower bound on the optimal expected makespan?* In all these stochastic problems, the difficult case is when job sizes have a large variance and hence one cannot apply concentration bounds naively. Kleinberg et al. [KRT00] cleverly used a notion of *effective size* (due to Hui [Hui88]) of random variables to give such a lower bound for identical machines, but the effective size depends crucially on the number of machines in the system. So when the machines are not identical and different jobs might want to use different sets of machines, how should we define effective sizes? Furthermore, we need to utilize this lower bound in identifying a good assignment. Recall that even in the deterministic special case, achieving a constant-factor approximation for unrelated machines [LST90] is much harder than for identical machines (where a greedy assignment suffices).

1.1 Results and Techniques

We show how to circumvent these problems and give the following theorem:

Theorem 1.1. *There is an $O(1)$-approximation algorithm for the problem of finding an assignment to minimize the expected makespan on unrelated machines.*

Our work naturally builds on the edifice of [KRT00]. However, we need several new ideas to achieve this. As mentioned above, the result for identical machines uses the notion of effective job size. This is a function of the number of machines m and the random variable X_j (recall that for identical machines, all X_{ij} are same for a job j and different machines i, so we can use X_j to denote the size of job j), closely related to the moment generating function of X_j. If the optimal expected makespan is 1, one shows that the sum of the effective job sizes essentially remains $O(m)$. On the flip side, if the total effective size is $O(m)$ then one can get a schedule with expected makespan $O(1)$. As noted earlier, in the unrelated setting it does not make sense to talk of a fixed parameter m; see § B.1 for more detail.

Instead, we show that for every k-subset of machines, the effective sizes of jobs assigned to these machines in the optimal solution, with respect to parameter k, must satisfy a "volume" constraint (as in the case of identical machines). Since this condition must hold for every subset of machines, we must consider an exponential number of such conditions. We then show how to write a suitable LP relaxation which gives a fractional assignment satisfying all such constraints (fractionally). Finally, given a solution to this LP, we show how to carefully choose the right parameter for effective size of each job and use it to build an $O(1)$-approximate schedule. Although our LP relaxation has an exponential number of constraints, we show that it suffices to satisfy a small subset of these constraints in the final schedule: this is crucial in obtaining a rounding algorithm.

Extension: Achieving High Profit. We then consider two extensions: in the first problem, which we call *Budgeted Makespan Minimization*, each job has a *reward* r_j (having no relationship to other parameters such as its size), and we are given a target reward value R. The goal is to assign some subset $S \subseteq [n]$ of jobs whose total reward $\sum_{j \in S} r_j$ is at least R, and to minimize the expected makespan of this assignment. Clearly, this generalizes the basic makespan minimization problem.

Theorem 1.2. *There is an $O(1)$-approximation algorithm for the budgeted makespan minimization problem on unrelated machines.*

To solve this, we extend the ideas for expected makespan scheduling to include an extra constraint about high reward. We again write a similar LP relaxation. Rounding this LP requires some additional ideas on top of those in Theorem 1.1. The new ingredient is that we need to round solutions to an assignment LP with two linear constraints. To do this without violating the budget, we utilize a "reduction" from the Generalized Assignment Problem to bipartite matching [ST93] as well as certain adjacency properties of the bipartite matching polytope [BR74].

Extension: ℓ_q Norms. Finally, we consider the problem of stochastic load balancing under ℓ_q norms. Note that given some assignment, we can denote the "load" on machine i by $L_i := \sum_{j \in J_i} X_{ij}$, and the "load vector" by $\mathbf{L} = (L_1, L_2, \ldots, L_m)$. The expected makespan minimization problem is to minimize $\mathbb{E}[\|\mathbf{L}\|_\infty]$. The *q-norm minimization problem* is the following: find an assignment of jobs to machines to minimize

$$\mathbb{E}\big[\|\mathbf{L}\|_q\big] = \mathbb{E}\left[\left(\sum_{i=1}^{m}\Big(\sum_{j \in J_i} X_{ij}\Big)^q\right)^{1/q}\right].$$

Our result for this setting is the following:

Theorem 1.3. *There is an $O(\frac{q}{\log q})$-approximation algorithm for the stochastic q-norm minimization problem on unrelated machines.*

The main idea here is to reduce this problem to a suitable instance of deterministic q-norm minimization with additional side constraints. We then show that existing techniques for deterministic q-norm minimization [AE05] can be extended to our obtain a constant-factor approximation for our generalization as well. We also need to use/prove some probabilistic inequalities to relate the deterministic sub-problem to the stochastic problem. We note that using general polynomial concentration inequalities [KV00, SS12] only yields an approximation ratio that is exponential in q. We obtain a much better $O(q/\log q)$-approximation factor by utilizing the specific form of the norm function. Specifically, we use the convexity of norms, a second-moment calculation and a concentration bound [JSZ85] for the q^{th} moment of sums of independent random variables.

It is well-known that the ℓ_∞ norm on vectors of length m can be approximated within constants by setting $q = \log m$, and hence Theorem 1.3 gives an $O(\frac{\log m}{\log\log m})$-approximation for ℓ_∞, which is weaker than the $O(1)$-approximation result from Theorem 1.1. However, our techniques do not seem to be able to directly improve, and we leave the question of getting an $O(1)$-approximation for all q-norms as an interesting open problem.

1.2 Other Related Work

Goel and Indyk [GI99] considered the stochastic load balancing problem on identical machines (same setting as [KRT00]) but for specific job-size distributions. For Poisson distributions they showed that Graham's algorithm achieves a 2-approximation, and for Exponential distributions they obtained a PTAS. Kleinberg et al. [KRT00] also considered stochastic versions of knapsack and bin-packing: given an overflow probability p, a feasible single-bin packing here corresponds to any subset of jobs such that their total size exceeds one with probability at most p. [GI99] gave better/simpler algorithms for these problems, under special distributions.

Recently, Deshpande and Li [LD11], and Li and Yuan [LY13] considered several combinatorial optimization problems including shortest paths, minimum spanning trees, where elements have weights (which are random variables), and one would like to find a solution (i.e. a subset of elements) whose expected utility is maximized. These results also apply to the stochastic versions of knapsack and bin-packing from [KRT00] and yield bicriteria approximations. The main technique here is a clever discretization of probability distributions. However, to the best of our knowledge, such an approach is not applicable to stochastic load balancing.

Stochastic scheduling has been studied in many different contexts, by different fields (see, e.g., [Pin04]). The work on approximation algorithms for these problems is more recent; see [MSU99] for some early work and many references. In this paper we consider the *(non-adaptive) fixed assignment model*, where jobs have to be assigned to machines up-front, and then the randomness is revealed. Hence, there is no element of adaptivity in these problems. This makes them suitable for settings where the decisions cannot be instantaneously implemented (e.g., for virtual circuit routing, or assigning customers to physically well-separated warehouses). A number of papers [MSU99, MUV06, IMP15, GMUX17] have considered scheduling problems in the *adaptive* setting, where assignments are done online and the assignment for a job may depend on the state of the system at the time of its assignment. See Appendix B.2 for a comparison of adaptive and non-adaptive settings in the load balancing problem.

2 Makespan Minimization

We consider the classical problem of scheduling jobs on unrelated machines with the objective of minimizing the makespan, where jobs have stochastic processing times. Formally, there are m machines and n jobs. For each job j and machine i, we are given a random variable $X_{i,j}$ which denotes the processing time of j on machine i. We assume that the random variables $X_{i,j}, X_{i',j'}$ are independent when $j \neq j'$ (the size of job j on two machines i and i may be related to each other). We assume access to the distribution of these random variables via some (succinct) representation.

A solution is an assignment of jobs to machines. Formally, a solution is a partition $\{J_i\}_{i=1}^m$ of the jobs among the machines, such that $J_i \subseteq [n]$ is the subset of jobs assigned to machine $i \in [m]$. The expected makespan of this solution (or schedule) is

$$\mathbb{E}\left[\max_{i=1}^{m} \sum_{j \in J_i} X_{i,j}\right].$$

Our goal is to find a solution which minimizes the expected makespan. We call this the *stochastic load balancing problem* (on unrelated machines), or StocMakespan for short. The main result of this section is:

Theorem 1.1. *There is an $O(1)$-approximation algorithm for the problem of finding an assignment to minimize the expected makespan on unrelated machines.*

Using a binary search scheme and scaling, it suffices to find one of the following:

(i) *upper bound:* a solution with expected makespan at most $O(1)$, or

(ii) *lower bound:* a certificate that the optimal expected makespan is more than one.

Hence, we assume that the optimal solution for the instance has unit expected makespan, and try to find a solution with expected makespan $b = O(1)$; if we fail we output a lower bound certificate.

2.1 Truncated and Exceptional Random Variables

The first step towards doing this is to divide each random variable $X_{i,j}$ into their *truncated* and *exceptional* parts.

- $X'_{i,j} := X_{i,j} \cdot \mathbf{I}_{(X_{i,j} \le 1)}$ (called the *truncated* part), and
- $X''_{i,j} := X_{i,j} \cdot \mathbf{I}_{(X_{i,j} > 1)}$ (called the *exceptional* part).

The reason for doing this is that these two kinds of random variables behave very differently w.r.t. the expected makespan. It turns out that expectation is a good measure of effective size for exceptional r.v.s, whereas one needs a more nuanced notion for truncated r.v.s (discussed below).

2.2 Effective Size and Intuition

As is often the case for stochastic optimization problems, we want to find some deterministic quantity that is a good surrogate for each random variable, and then use this deterministic surrogate instead of the actual random variable. For stochastic load balancing, a widely used quantity is the *effective size*, which is based on the logarithm of the (exponential) moment generating function [Hui88, Kel96, EM93]. While this will not solve our problem immediately, it will form a crucial ingredient in the mix.

Definition 2.1 (Effective Size)**.** *For any random variable X and parameter $0 < p < 1$, define*

$$\beta_p(X) \;:=\; \frac{1}{\log(1/p)} \cdot \log \mathbb{E}\left[e^{\log(1/p)\cdot X}\right]. \tag{2}$$

Also define $\beta_1(X) := \mathbb{E}[X]$.

To see the intuition for the effective size, consider a set of independent r.v.s $Y_1, \ldots, Y_k$ all assigned to the same machine. Then if $\sum_i \beta_p(Y_i) \leq b$,

$$\Pr[\sum_i Y_i \geq c] = \Pr[e^{\log(1/p)\sum_i Y_i} \geq e^{\log(1/p)c}] \leq \frac{\mathbb{E}[e^{\log(1/p)\sum_i Y_i}]}{e^{\log(1/p)c}} = \frac{\prod_i \mathbb{E}[e^{\log(1/p)Y_i}]}{e^{\log(1/p)c}}$$

Taking logarithms, we get

$$\log \Pr[\sum_i Y_i \geq c] \leq \log(1/p) \cdot \left[\sum_i \beta_p(Y_i) - c\right] \quad \Longrightarrow \quad \Pr[\sum_i Y_i \geq c] \leq p^{c-b}.$$

The above calculation, very reminiscent of the standard Chernoff bound argument, can be summarized by the following lemma (shown, e.g., in [Hui88]).

Lemma 2.2 (Upper Bound)**.** *For indep. r.v.s $Y_1, \ldots, Y_n$, if $\sum_i \beta_p(Y_i) \leq b$ then $\Pr[\sum_i Y_i \geq c] \leq p^{c-b}$.*

The usefulness of this definition comes from a partial converse, proved in [KRT00]:

Lemma 2.3 (Lower Bound)**.** *For indep. Bernoulli r.v.s $Y_1, \ldots, Y_n$ where each Y_i has non-zero size s_i being an inverse power of 2 such that $1/(\log(1/p)) \leq s_i \leq 1$, if $\sum_i \beta_p(Y_i) \geq 7$ then $\Pr[\sum_i Y_i \geq 1] \geq p$.*

In using the effective size, it is important to set the parameter p carefully. In its use by Kleinberg et al. [KRT00] for makespan minimization on identical machines, they set $p = \frac{1}{m}$. Loosely, if the effective sizes sum to at most 1 on each machine, the probability of some machine exceeding load 3 would be $m \cdot (1/m)^2$ by Lemma 2.2. On the other hand, if the effective sizes sum to more than $20m$ overall, then even if we assign them to balance out these evenly, each machine would have makespan at least 1 with probability $1/m$, and so with m machines this gives a certificate that the makespan is $\Omega(1)$.

This kind of argument breaks down if there are many different kinds of machines, and have jobs that can go on only some subset of machines, because we don't know what probability of success we want to aim for. For example, even if the machines had the same speed, but there were jobs that could go only on $\sqrt{m}$ of these machines, and others could go on the remaining $m - \sqrt{m}$ of them, we would want their effective sizes to be quite different. See Appendix B.1 for a detailed example. And once we go to the unrelated machines setting, it is not clear that any combinatorial argument suffices.

At a high level, the ideas we use are the following: first, in §2.3 we show a more involved lower bound based on the effective sizes of jobs assigned to every subset of machines. This is captured using an exponentially-sized LP which is solvable in polynomial time. Then, to show that this lower bound is a good one, we give a new rounding algorithm for this LP in §2.4 to get an expected makespan within a constant factor of the lower bound.

2.3 A New Lower Bound

Our starting point is a more general lower bound on the makespan. The (contrapositive of the) following lemma says that if the effective sizes are large then the expected makespan must be large too. This is much the same spirit as Lemma 2.3, but for the general setting of unrelated machines.

Lemma 2.4 (New Valid Inequalities)**.** *Consider any feasible solution that assigns jobs J_i to each machine $i \in [m]$. If the expected makespan $\mathbb{E}\left[\max_{i=1}^m \sum_{j \in J_i} X_{i,j}\right] \leq 1$, then*

$$\sum_{i=1}^m \sum_{j \in J_i} \mathbb{E}[X''_{i,j}] \leq 2, \qquad \textit{and} \tag{3}$$

$$\text{for all } K \subseteq [m], \quad \sum_{i\in K}\sum_{j\in J_i} \beta_{1/k}(X'_{i,j}) \;\le\; O(k), \quad \text{where } k = |K|. \tag{4}$$

Proof. The first inequality (3) focuses on the exceptional parts, and loosely follows from the intuition that if the sum of biases of a set of independent coin flips is large (exceeds 2 in this case) then you expect one of them to come up heads. Formally, the proof follows from Lemma A.3 applied to $\{X''_{ij} : j \in J_i, i \in [m]\}$.

For the second inequality (4), consider any subset $K \subseteq [m]$ of the machines. Then the total effective size of the jobs assigned to these machines must be small, where now the effective size can be measured with parameter $p = 1/|K|$. Formally applying Lemma A.4 only to the k machines in K and the truncated random variables $\{X'_{i,j} : i \in K, j \in J_i\}$ corresponding to jobs assigned to these machines, we obtain the desired inequality. ∎

Given these valid inequalities, our algorithm now seeks an assignment satisfying (3)–(4). If we fail, the lemma assures us that the expected makespan must be large. On the other hand, if we succeed, such a "good" assignment by itself is not sufficient. The challenge is to show the converse of Lemma 2.4, i.e., that any assignment satisfying (3)–(4) gives us an expected makespan of $O(1)$. Indeed, towards this goal, we first write an LP relaxation with an exponential number of constraints, corresponding to (4). We can solve this LP using the ellipsoid method. Then, instead of rounding the fractional solution to satisfy all constraints (which seems very hard), we show how to satisfy only a carefully chosen subset of the constraints (4) so that the expected makespan can still be bounded. Let us first give the LP relaxation.

In the ILP formulation of the above lower bound, we have binary variables $y_{i,j}$ to denote the assignment of job j to machine i, and fractional variables $z_i(k)$ denote the total load on machine i in terms of the deterministic effective sizes $\beta_{1/k}$. Lemma 2.4 shows that the following feasibility LP is a valid relaxation:

$$\sum_{i=1}^{m} y_{i,j} = 1 \qquad \forall j \in [n], \tag{5}$$

$$z_i(k) - \sum_{j=1}^{n} \beta_{1/k}(X'_{i,j}) \cdot y_{i,j} = 0 \qquad \forall i \in [m],\ k = 1, 2, \cdots m, \tag{6}$$

$$\sum_{i=1}^{m}\sum_{j=1}^{n} \mathbb{E}[X''_{i,j}] \cdot y_{i,j} \le 2 \tag{7}$$

$$\sum_{i\in K} z_i(k) \le b \cdot k, \qquad \forall K \subseteq [m] \text{ with } |K| = k,\ \forall k = 1, 2, \cdots m, \tag{8}$$

$$y_{i,j}, z_i(k) \ge 0 \qquad \forall i, j, k \tag{9}$$

In the above LP, $b = O(1)$ denotes the constant in the right-hand-side of (4).

Although this LP has an exponential number of constraints (because of (8)), we can give an efficient separation oracle. Indeed, consider a candidate solution $(y_{i,j}, z_i(k))$, and some integer k; suppose we want to verify (8) for sets K with $|K| = k$. We just need to look at the k machines with the highest $z_i(k)$ values and check that the sum of $z_i(k)$ for these machines is at most bk. So, using the Ellipsoid method we can assume that we have an optimal solution (y, z) for this LP in polynomial time. We can summarize this in the following proposition:

Proposition 2.5 (Lower Bound via LP)**.** *The linear program (5)–(9) can be solved in polynomial time. Moreover, if it is infeasible, then the optimal expected makespan is more than* 1.

2.4 The Rounding

Intuition. In order to get some intuition about the rounding algorithm, let us first consider the case when the assignment variables $y_{i,j}$ are either 0 or 1, i.e., the LP solution assigns each job integrally to a machine. In order to bound the expected makespan of this solution, let Z_j denote the variable X_{ij}, where j is assigned to i by this solution. First consider the exceptional parts Z''_j of the random variables. Constraint (7) implies that $\sum_j \mathbb{E}[Z''_j]$ is at most 2. Even if the solution assigns all of these jobs to the same machine, the contribution of these jobs to the expected makespan is at most $\sum_j \mathbb{E}[Z''_j]$, and hence at most 2. Thus, we need only worry about the truncated Z'_j variables.

Now for a machine i and integer $k \in [m]$, let $z_i(k)$ denote the sum of the effective sizes $\beta_{1/k}(Z'_j)$ for the truncated r.v.s assigned to i. We can use Lemma 2.2 to infer that if $z_i(m) = \sum_{j \text{ assigned to } i} \beta_{1/m}(Z_j) \leq b$, then the probability that these jobs have total size at most $b + 2$ is at least $1 - 1/m^2$. Therefore, if $z_i(m) \leq b$ for all machines $i \in [m]$, then by a trivial union bound the probability that makespan is more than $b + 2$ is at most $1/m$. Unfortunately, we are not done. All we know from constraint (8) is that the *average* value of $z_i(m)$ is at most b (the average being taken over the m machines). However, there is a clean solution. It follows that there is at least one machine i for which $z_i(m)$ is at most b, and so the expected load on such machines stays $O(1)$ with high probability. Now we can ignore such machines, and look at the residual problem. We are left with $k < m$ machines. We recurse on this sub-problem (and use the constraint (8) for the remaining set of machines). The overall probability that the load exceeds $O(1)$ on any machine can then be bounded by applying a union bound.

Next, we address the fact that $y_{i,j}$ may be not be integral. It seems very difficult to round a fractional solution while respecting all the (exponentially many) constraints in (8). Instead, we observe that the expected makespan analysis (outlined above) only utilizes a linear number of constraints in (8), although this subset is not known *a priori*. Moreover, for each machine i, the above analysis only uses $z_i(k)$ for a *single* value of k (say k_i). Therefore, it suffices to find an integral assignment that bounds the load of each machine i in terms of effective sizes β_{1/k_i}. It turns out that this problem is precisely an instance of the Generalized Assignment Problem (GAP), for which we utilize the algorithm from [ST93].

The Rounding Procedure. We now describe the iterative procedure formally. Assume we have an LP solution $\{y_{i,j}\}_{i\in[m],j\in[n]}, \{z_i(k)\}_{i,k\in[m]}$.

1. Initialize $\ell \leftarrow m$, $L \leftarrow [m]$, $c_{i,j} \leftarrow \mathbb{E}[X''_{i,j}]$.
2. While $(\ell > 0)$ do:
   (a) Set $L' \leftarrow \{i \in L : z_i(\ell) \leq b\}$. Machines in L' are said to be in *class* ℓ.
   (b) Set $p_{i,j} \leftarrow \beta_{1/\ell}(X'_{i,j})$ for all $i \in L'$ and $j \in [n]$.
   (c) Set $L \leftarrow L \setminus L'$ and $\ell = |L|$.
3. Define a deterministic instance $\mathcal{I}$ of the GAP as follows: the set of jobs and machines remains unchanged. For each job j and machine i, define $p_{i,j}$ and $c_{i,j}$ as above. The makespan bound is b. Use the algorithm of Shmoys and Tardos [ST93] to find an assignment of jobs to machines. Output this solution.

Recall that in an instance $\mathcal{I}$ of GAP, we are given a set of m machines and n jobs. For each job j and machine i, we are given two quantities: $p_{i,j}$ is the processing time of j on machine i, and $c_{i,j}$ is the cost of assigning j to i. We are also given a makespan bound b. Our goal is to assign jobs to machines to minimize the total cost of assignment, subject to the total processing time of jobs assigned to each machine being at most b. If the natural LP relaxation for this problem has optimal value $C^\star$ then the algorithm in [ST93] finds in polynomial-time an assignment with cost at most $C^\star$ and makespan is at most $B + \max_{i,j} p_{i,j}$.

2.5 The Analysis

We begin with some simple observations:

Observation 2.6. *The above rounding procedure terminates in at most m iterations. Furthermore, for any $1 \le \ell \le m$, there are at most ℓ machines of class at most ℓ.*

Proof. The first statement follows from the fact that $L' \neq \emptyset$ in each iteration. To see this, consider any iteration involving a set L of ℓ machines. The LP constraint (8) for L implies that $\sum_{i \in L} z_i(k) \le b \cdot \ell$, which means there is some $i \in L$ with $z_i(\ell) \le b$, i.e., $L' \neq \emptyset$. The second statement follows from the rounding procedure: the machine classes only decrease over the run of the algorithm, and the class assigned to any unclassified machine equals the current number of unclassified machines. ∎

Observation 2.7. *The solution y is a feasible fractional solution to the natural LP relaxation for the* GAP *instance $\mathcal{I}$. This solution has makespan at most b and fractional cost at most 2. The rounding algorithm of Shmoys and Tardos [ST93] yields an assignment with makespan at most $b + 1$ and cost at most 2 for the instance $\mathcal{I}$.*

Proof. Recall that the natural LP relaxation is the following:

$$\begin{aligned}
\min \quad & \textstyle\sum_{ij} c_{ij} y_{ij} & & \\
& \textstyle\sum_j p_{ij} y_{ij} \le b & \forall i & \qquad (10)\\
& \textstyle\sum_i y_{ij} = 1 & \forall j & \qquad (11)\\
& y_{ij} = 0 & \forall j \ s.t.\ p_{ij} > b & \qquad (12)\\
& y \ge 0 & &
\end{aligned}$$

Firstly, note that by (5), y is a valid fractional assignment that assigns each job to one machine, which satisfies (11).

Next we show (10), i.e., that $\max_{i=1}^m \sum_{j=1}^n p_{i,j} \cdot y_{i,j} \le b$. This follows from the definition of the deterministic processing times $p_{i,j}$. Indeed, consider any machine $i \in [m]$. Let ℓ be the class of machine i, and L be the subset of machines in the iteration when i is assigned class ℓ. This means that $p_{i,j} = \beta_{1/\ell}(X'_{i,j})$ for all $j \in [n]$. Also, because machine $i \in L'$, we have $z_i(\ell) = \sum_{j=1}^n \beta_{1/\ell}(X'_{i,j}) \cdot y_{i,j} \le b$. So we have $\sum_{j=1}^n p_{i,j} \cdot y_{i,j} \le b$ for each machine $i \in [m]$.

Finally, since the random variable $X'_{i,j}$ is at most 1, we get that for any parameter p, $\beta_p(X'_{i,j}) \le 1 \le b$; this implies that $p_{ij} \le b$ and hence the constraints (12) are vacuously true. Finally, by (7), the objective is $\sum_{i=1}^m \sum_{j=1}^n c_{i,j} \cdot y_{i,j} = \sum_{i=1}^m \sum_{j=1}^n \mathbb{E}[X''_{i,j}] \cdot y_{i,j} \le 2$. Therefore the rounding algorithm [ST93] yields an assignment of makespan at most $b + p_{\max} \le b + 1$, and of cost at most 2. ∎

In other words, if J_i be the set of jobs assigned to machine i by our algorithm, Observation 2.7 shows that this assignment has the following properties (let ℓ_i denote the class of machine i):

$$\sum_{i=1}^m \sum_{j \in J_i} \mathbb{E}[X''_{i,j}] \ \le\ 2, \quad \textit{and} \qquad (13)$$

$$\sum_{j \in J_i} \beta_{1/\ell_i}(X'_{i,j}) \ \le\ b + 1, \qquad \forall i \in [m]. \qquad (14)$$

Note that we ideally wanted to give an assignment that satisfied (3)–(4), but instead of giving a bound for all sets of machines, we give just the bound on the β_{1/ℓ_i} values of the jobs for each machine i. The next lemma shows this is enough.

Lemma 2.8 (Bounding the Makespan)**.** *The expected makespan of the assignment $\{J_i\}_{i\in[m]}$ is at most $4b + 10$.*

Proof. Let I^{hi} denote the index set of machines of class 3 or higher. Observation 2.6 shows that there are at most 3 machines which are not in I^{hi}. For a machine i, let $T_i = \sum_{j\in J_i} X'_{i,j}$ denote the total load due to truncated sizes of jobs assigned to it. Clearly, the makespan is bounded by

$$\max_{i\in I^{\text{hi}}} T_i + \sum_{i\notin I^{\text{hi}}} T_i + \sum_{i=1}^{m}\sum_{j\in J_i} X''_{i,j}.$$

The expectation of third term is at most two, using (13). We now bound the expectation of the second term above. A direct application of Jensen's inequality (Theorem A.1) for concave functions shows that $\beta_p(X) \geq \mathbb{E}[X]$ for any random variable X and any $p \in (0, 1)$. Then applying inequality (14) shows that $\mathbb{E}[T_i] \leq b + 1$ for any machine i. Therefore, the expected makespan of our solution is at most

$$\mathbb{E}\left[\max_{i\in I^{\text{hi}}} T_i\right] + 3(b + 1) + 2. \tag{15}$$

It remains to bound the first term above.

Observation 2.9. *For any machine i, $\Pr\left[\sum_{j\in J_i} X'_{i,j} > b + 1 + \alpha\right] \leq \ell_i^{-\alpha}$ for all $\alpha \geq 0$.*

Proof. Inequality (14) for machine i shows that $\sum_{j\in J_i} \beta_{1/\ell_i}(X'_{i,j}) \leq b+1$. But recalling the definition of the effective size (as the log-MGF), the result follows from Lemma 2.2. ■

Now we can bound the probability of any machine in I^{hi} having a high makespan.

Lemma 2.10. *For any $\alpha > 2$, $\Pr\left[\max_{i\in I^{hi}} T_i > b + 1 + \alpha\right] \leq 2^{2-\alpha}/(\alpha - 2)$.*

Proof. Using a union bound, we get

$$\begin{aligned}
\Pr\left[\max_{i\in I^{\text{hi}}} T_i > b + 1 + \alpha\right] &\leq \sum_{\ell=3}^{m}\sum_{i:\ell_i=\ell} \Pr\left[T_i > b + 1 + \alpha\right] \\
&\leq \sum_{\ell=3}^{m} \ell^{-\alpha} \cdot (\text{number of class } \ell \text{ machines}) \\
&\leq \sum_{\ell=3}^{m} \ell^{-\alpha+1} \quad \leq \quad \int_{x=2}^{\infty} x^{-\alpha+1} dx \quad = \quad \frac{2^{-\alpha+2}}{\alpha - 2}.
\end{aligned}$$

The first inequality uses a trivial union bound, the second uses Observation 2.9 above, and the third inequality is by Observation 2.6. ■

Using the above lemma, we get

$$\mathbb{E}[\max_{i\in I^{\text{hi}}} T_i] = (b + 4) + \int_{\alpha=3}^{\infty} \Pr[\max_{i\in I^{\text{hi}}} T_i > b + 1 + \alpha] d\alpha \leq (b + 4) + \int_{\alpha=3}^{\infty} 2^{2-\alpha} d\alpha \leq b + 5.$$

Inequality (15) now shows that the expected makespan is at most $(b + 5) + 3(b + 1) + 2$. ■

This completes the proof of Theorem 1.1.

3 Budgeted Makespan Minimization

We now consider a generalization of the StocMakespan problem, called BudgetStocMakespan, where each job j also has *reward* $r_j \geq 0$. We are required to schedule some subset of jobs whose total reward is at least some target value R. The objective, again, is to minimize the expected makespan. If the target $R = \sum_{j\in[n]} r_j$ then we recover the StocMakespan problem. We show:

Theorem 1.2. *There is an $O(1)$-approximation algorithm for the budgeted makespan minimization problem on unrelated machines.*

Naturally, our algorithm/analysis will build on the ideas developed in §2, but we will need some new ideas to handle the fact that only a subset of jobs need to be scheduled. As in the case of StocMakespan problem, we can formulate a suitable LP relaxation. A similar rounding procedure reduces the stochastic problem to a deterministic problem, which we call Budgeted GAP. An instance of Budgeted GAP is similar to that of GAP, besides the additional requirement that jobs have rewards and we are required to assign jobs of total reward at least some target R. Rounding the natural LP relaxation for Budgeted GAP turns out to be non-trivial. Indeed, using ideas from [ST93], we reduce this rounding problem to rounding a fractional matching solution with additional constraints, and solve the latter using polyhedral properties of bipartite matching polyhedra. Details in §C.

4 ℓ_q-norm Objectives

In this section, we prove Theorem 1.3. Given an assignment $\{J_i\}_{i=1}^m$, the load L_i on machine i is the r.v. $L_i := \sum_{j\in J_i} X_{ij}$. Our goal is to find an assignment to minimize the expected q-norm of the load vector $\mathbf{L} := (L_1, L_2, \dots, L_m)$. Recall that the makespan is $\|\mathbf{L}\|_\infty$, so the q-norm problem is a generalization of StocMakespan. Our main result here is:

Theorem 1.3. *There is an $O(\frac{q}{\log q})$-approximation algorithm for the stochastic q-norm minimization problem on unrelated machines.*

We begin by assuming that we know the optimal value M of the q-norm. Our approach parallels that for the case of minimizing the expected makespan, with some changes. In particular, the main steps are: (i) find valid inequalities satisfied by any assignment for which $\mathbb{E}[\|\mathbf{L}\|_q] \leq M$, (ii) reduce the problem to a deterministic assignment problem for which any feasible solution satisfies the valid inequalities above, (iii) solve the deterministic problem by writing a convex programming relaxation, and give a rounding procedure for a fractional solution to this convex program, and (iv) prove that the resulting assignment of jobs to machines has small q-norm of the load vector.

4.1 Useful Bounds

We start with stating some valid inequalities satisfied by any assignment $\{J_i\}_{i=1}^m$.

For each $j \in [n]$ define $Y_j = X_{ij}$ where $j \in J_i$. By definition of M, we know that

$$\mathbb{E}\left[\left(\sum_{i=1}^m (\sum_{j\in J_i} Y_j)^q\right)^{1/q}\right] \leq M. \tag{16}$$

As in Section 2, we split each random variable Y_j into two parts: truncated $Y'_j = Y_j \cdot \mathbf{I}_{Y_j \leq M}$ and exceptional $Y''_j = Y_j \cdot \mathbf{I}_{Y_j > M}$. The claim below is analogous to (3), and states that the total expected size of the exceptional parts cannot be too large.

Claim 4.1. *For any schedule satisfying* (16)*, we have* $\sum_{j=1}^{n} \mathbb{E}[Y_j''] \leq 2M$*.*

Proof. Suppose for a contradiction that $\sum_{j=1}^{n} \mathbb{E}[Y_j''] > 2M$. Lemma A.3 implies $\mathbb{E}[\max_{j=1}^{n} Y_j''] > M$. Now using the monotonicity of norms and the fact that $Y_j'' \leq Y_j$, we have

$$\max_{j=1}^{n} Y_j'' \leq \|(Y_1'', \cdots, Y_n'')\|_q \leq \left(\sum_{i=1}^{m} (\sum_{j \in J_i} Y_j)^q \right)^{1/q},$$

which contradicts (16). ∎

Our next two bounds deal with the truncated r.v. Y_j'. The first one states that if replace Y_j' by its expectation $\mathbb{E}[Y_j']$, the q-norm of this load vector of expectations cannot exceed M. The second bound states that the expected q-moments of the vector $(Y_j')_{j=1}^{n}$ is bounded by a constant times M^q.

Claim 4.2. *For any schedule satisfying* (16) *we have*

$$\sum_{i=1}^{m} \left(\sum_{j \in J_i} \mathbb{E}[Y_j'] \right)^q \leq M^q.$$

Proof. Since the function $f(Y_1', \cdots Y_n') := \left(\sum_{i=1}^{m} (\sum_{j \in J_i} Y_j')^q \right)^{1/q}$ is a norm and hence convex, Jensen's inequality (Theorem A.1) implies $\mathbb{E}[f(Y_1', \cdots Y_n')] \geq f(\mathbb{E}[Y_1'], \cdots \mathbb{E}[Y_n'])$. Raising both sides to the q^{th} power and using (16), the claim follows. ∎

Claim 4.3. *Let* $\alpha = 2^{q+1} + 8$*. For any schedule satisfying* (16) *we have*

$$\sum_{j=1}^{n} \mathbb{E}[(Y_j')^q] \leq \alpha \cdot M^q.$$

Proof. Define $Z := \sum_{j=1}^{n} (Y_j')^q$ as the quantity of interest. Observe that it is the sum of independent $[0, M^q]$ bounded random variables. Since $q \geq 1$ and the r.v.s are non-negative, $Z \leq \sum_{i=1}^{m} (\sum_{j \in J_i} Y_j')^q$. Thus (16) implies $\mathbb{E}[Z^{1/q}] \leq M$. However, now Jensen's inequality cannot help upper-bound $\mathbb{E}[Z]$.

Instead we use a second-moment calculation. To reduce notation let $Z_j := (Y_j')^q$, so $Z = \sum_{j=1}^{n} Z_j$. The variance of Z is $var(Z) = \mathbb{E}[Z^2] - \mathbb{E}[Z]^2 \leq \sum_{j=1}^{n} \mathbb{E}[Z_j^2] \leq M^q \cdot \mathbb{E}[Z]$ as each Z_j is $[0, M^q]$ bounded. By Chebyshev's inequality,

$$\Pr\left[Z < \frac{\mathbb{E}[Z]}{2} - 4M^q \right] \leq \frac{var(Z)}{(\mathbb{E}[Z]/2 + 4M^q)^2} \leq \frac{var(Z)}{(\mathbb{E}[Z]/2) \cdot 4M^q} \leq \frac{2M^q \cdot E[Z]}{\mathbb{E}[Z] \cdot 4M^q} \leq \frac{1}{2}.$$

This implies

$$E[Z^{1/q}] \geq \frac{1}{2} \left(\frac{\mathbb{E}[Z]}{2} - 4M^q \right)^{1/q}$$

Using the bound $\mathbb{E}[Z^{1/q}] \leq M$ from above, we now obtain $\mathbb{E}[Z] \leq 2 \cdot ((2M)^q + 4M^q)$ as desired. ∎

In the next sections, we show that the three bounds above are enough to get a meaningful lower bound on the optimal q-norm of load.

4.2 Reduction to a Deterministic Scheduling Problem

We now formulate a surrogate deterministic scheduling problem, which we call q-DetSched. An instance of this problem has n jobs and m machines. For each job j and machine i, there is a processing time p_{ij} and two costs c_{ij} and d_{ij}. There are also bounds C and D on the two cost functions respectively. The goal is to find an assignment of jobs to machines that minimizes the q-norm of the machine loads subject to the constraint that the total c-cost and d-cost of the assignments are at most C and D respectively. We now show how to convert an instance $\mathcal{I}^{stoc}$ of the (stochastic) expected q-norm minimization problem to an instance $\mathcal{I}^{det}$ of the (deterministic) q-DetSched problem.

Suppose $\mathcal{I}^{stoc}$ has m machines and n jobs, with random variables X_{ij} for each machine i and job j. As before, let $X'_{ij} = X_{ij} \cdot \mathbf{I}_{X_{ij} \le M}$ and $X''_{ij} = X_{ij} \cdot \mathbf{I}_{X_{ij} > M}$ denote the truncated and exceptional parts of each random variable X_{ij} respectively. Then instance $\mathcal{I}^{det}$ has the same set of jobs and machines as those in $\mathcal{I}$. Furthermore, define

- the processing time $p_{ij} = \mathbb{E}[X'_{ij}]$,
- the c-cost $c_{ij} = \mathbb{E}[X''_{ij}]$ with bound $C = 2M$, and
- the d-cost $d_{ij} = \mathbb{E}[(X'_{ij})^q]$ with bound $D = \alpha \cdot M^q$.

Observation 4.4. *If there is any schedule of expected q-norm at most M in the instance $\mathcal{I}^{stoc}$, then optimal value of the instance $\mathcal{I}^{det}$ is at most M.*

Proof. This follows directly from Claims 4.1, 4.2 and 4.3. ∎

4.3 Approximation Algorithm for q-DetSched

Our approximation algorithm for the q-DetSched problem is closely based on the algorithm for unrelated machine scheduling to minimize ℓ_q-norms [AE05]. We show:

Theorem 4.5. *There is a polynomial-time algorithm that given any instance $\mathcal{I}^{det}$ of q-DetSched, finds a schedule with (i) q-norm of processing times at most $2^{1+2/q} \cdot OPT(\mathcal{I}^{det})$, (ii) c-cost at most $3C$ and (iii) d-cost at most $3D$.*

Proof Sketch. We only provide a sketch as many of these ideas parallel those from [AE05]. Start with a convex programming "relaxation" with variables x_{ij} (for assigning job j to machine i).

$$\min \left\{ \sum_{i=1}^{m} \ell_i^q + \sum_{ij} p_{ij}^q \cdot x_{ij} \;\middle|\; \ell_i = \sum_j p_{ij} \cdot x_{ij} \,\forall i,\, \sum_i x_{ij} = 1 \,\forall j,\, \sum_{ij} c_{ij} \cdot x_{ij} \le C,\, \sum_{ij} d_{ij} \cdot x_{ij} \le D \right\}.$$

This convex program can be solved to arbitrary accuracy and its optimal value is $V \le 2 \cdot OPT(\mathcal{I}^{det})^q$. Let (x, ℓ) denote the optimal fractional solution below.

We now further reduce this q-norm problem to GAP. The GAP instance $\mathcal{I}^{gap}$ has the same set of jobs and machines as those in $\mathcal{I}^{det}$. For a job j and machine i, the processing time remains p_{ij}. However, the cost of assigning j to i is now $\gamma_{ij} := \frac{c_{ij}}{C} + \frac{d_{ij}}{D} + \frac{p_{ij}^q}{V}$. Furthermore, we impose a bound of ℓ_i on the total processing time of jobs assigned to each machine i (i.e., the makespan on i is constrained to be at most ℓ_i). Note that the solution x to the convex program is also a feasible fractional solution to the natural LP-relaxation for GAP with an objective function value of $\sum_{ij} \gamma_{ij} \cdot x_{ij} \le 3$. The rounding algorithm in [ST93] can now be used to round x into an integral assignment $\{A_{ij}\}$ with γ-cost also at most 3, and load on each machine i being $L_i \le \ell_i + m_i$, where m_i denotes the maximum processing time of any job assigned to machine i by this algorithm. The definition of γ and the bound on the

γ-cost implies that the c-cost and d-cost of this assignment are at most $3C$ and $3D$ respectively. To bound the q-norm of processing times,

$$\sum_{i=1}^{m} L_i^q \leq 2^{q-1}\left(\sum_i \ell_i^q + \sum_i m_i^q\right) \leq 2^{q-1}\left(V + \sum_{ij} p_{ij}^q \cdot A_{ij}\right) \leq 2^{q-1}(V+3V) = 2^{q+1}\cdot V.$$

Above, the first inequality uses $(a+b)^q \leq 2^{q-1}(a^q+b^q)$, and the third inequality uses the fact that $p_{ij}^q A_{ij} \leq V \cdot \gamma_{ij} A_{ij} \leq 3V$ by the bound on the γ cost. The proof is now completed by using $V \leq 2 \cdot OPT(\mathcal{I}^{det})^q$. ■

4.4 Interpreting the Rounded Solution

Starting from an instance $\mathcal{I}^{stoc}$ of expected q-norm minimization problem, we first constructed an instance $\mathcal{I}^{det}$ of q-DetSched. Let $\mathcal{J} = (J_1, \dots, J_m)$ denote the solution found by applying Theorem 4.5 to the instance $\mathcal{I}^{det}$. If the q-norm of processing times of this assignment (as a solution for $\mathcal{I}^{det}$) is more than $2^{1+2/q}M$ then using Observation 4.4 and Theorem 4.5, we obtain a certificate that the optimal value of $\mathcal{I}^{stoc}$ is more than M. So we assume that $\mathcal{J}$ has objective at most $2^{1+2/q}M$ (as a solution to $\mathcal{I}^{det}$). We use this assignment as a solution for the stochastic problem as well. It remains to bound the expected q-norm of this assignment.

By the reduction from $\mathcal{I}^{stoc}$ to $\mathcal{I}^{det}$, and the statement of Theorem 4.5, we know that

$$\sum_{i=1}^{m}\sum_{j\in J_i} \mathbb{E}[X''_{ij}] = \sum_{i=1}^{m}\sum_{j\in J_i} c_{ij} \leq 6M \tag{17}$$

$$\sum_{i=1}^{m}\left(\sum_{j\in J_i} \mathbb{E}[X'_{ij}]\right)^q = \sum_{i=1}^{m}(\sum_{j\in J_i} p_{ij})^q \leq 2^{q+2}\cdot M^q \tag{18}$$

$$\sum_{i=1}^{m}\sum_{j\in J_i} \mathbb{E}[(X'_{ij})^q] = \sum_{i=1}^{m}\sum_{j\in J_i} d_{ij} \leq 3\alpha M^q \tag{19}$$

We now derive properties of this assignment as a solution for $\mathcal{I}^{stoc}$.

Claim 4.6. *The expected q-norm of exceptional jobs $\mathbb{E}[(\sum_{i=1}^m(\sum_{j\in J_i} X''_{ij})^q)^{1/q}] \leq 6M$.*

Proof. This follows from (17), since the ℓ_q-norm of a vector is at most its ℓ_1-norm. ■

Claim 4.7. *The expected q-norm of truncated jobs $\mathbb{E}[(\sum_{i=1}^m(\sum_{j\in J_i} X'_{ij})^q)^{1/q}] \leq O(\frac{q}{\log q})M$.*

Proof. Define random variables $Q_i := (\sum_{j\in J_i} X'_{ij})^q$, so that the q-norm of the loads is

$$Q := (\textstyle\sum_{i=1}^m Q_i)^{1/q} = (\sum_{i=1}^m(\sum_{j\in J_i} X'_{ij})^q)^{1/q}.$$

Since $f(Q_1,\cdots Q_m) = (\sum_{i=1}^m Q_i)^{1/q}$ is a concave function for $q \geq 1$, using Jensen's inequality (Theorem A.1) again,

$$\mathbb{E}[Q] \leq \left(\sum_{i=1}^{m}\mathbb{E}[Q_i]\right)^{1/q}. \tag{20}$$

We can bound each $\mathbb{E}[Q_i]$ separately using Rosenthal's inequality (Theorem A.2):

$$\mathbb{E}[Q_i] = \mathbb{E}\left[(\sum_{j\in J_i} X'_{ij})^q\right] \leq K^q \cdot \left((\sum_{j\in J_i}\mathbb{E}[X'_{ij}])^q + \sum_{j\in J_i}\mathbb{E}[(X'_{ij})^q]\right),$$

where $K = O(q/\log q)$. Summing this over all $i = 1, \ldots, m$ and using (18) and (19), we get

$$\sum_{i=1}^{m} \mathbb{E}[Q_i] \leq K^q \cdot (2^{q+2} + 3\alpha) M^q \tag{21}$$

Recall from Claim 4.3 that $\alpha = 2^{q+1} + 8$. Now plugging this into (20) we obtain $\mathbb{E}[Q] \leq O(K) \cdot M$. ■

Finally, using Claims 4.6 and 4.7 and the triangle inequality, the expected q-norm of solution $\mathcal{J}$ is $O(\frac{q}{\log q}) \cdot M$, which completes the proof of Theorem 1.3.

5 Conclusions

In this paper we considered the problem of assigning jobs to machines to minimize the expected makespan. While the case of identical machines has been understood for some time, our results give the first results for the considerably harder unrelated machines case. We also give results to settings where we are not required to schedule all the jobs, and for the case of minimizing the q-norms of machine loads (cf. makespan is the ℓ_∞ norm). Apart from tightening the factors of our results further, an intriguing next direction is that of virtual circuit routing, where we must route paths to minimize the maximum congestion. While some of our ideas extend, we must now deal with the load of different edges being correlated to each other (if there are paths that share these edges).

References

[AE05] Yossi Azar and Amir Epstein, *Convex programming for scheduling unrelated parallel machines*, Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005, 2005, pp. 331–337.

[BR74] M. L. Balinski and Andrew Russakoff, *On the assignment polytope*, SIAM Rev. **16** (1974), no. 4, 516 – 525.

[EM93] A. I. Elwalid and D. Mitra, *Effective bandwidth of general markovian traffic sources and admission control of high speed networks*, IEEE/ACM Transactions on Networking **1** (1993), no. 3, 329–343.

[GI99] Ashish Goel and Piotr Indyk, *Stochastic load balancing and related problems*, 40th Annual Symposium on Foundations of Computer Science, FOCS '99, 17-18 October, 1999, New York, NY, USA, 1999, pp. 579–586.

[GMUX17] Varun Gupta, Benjamin Moseley, Marc Uetz, and Qiaomin Xie, *Stochastic online scheduling on unrelated machines*, Integer Programming and Combinatorial Optimization, Springer International Publishing, 2017, pp. 228–240.

[HS87] Dorit S. Hochbaum and David B. Shmoys, *Using dual approximation algorithms for scheduling problems theoretical and practical results*, J. ACM **34** (1987), no. 1, 144–162.

[HS88] ______, *A polynomial approximation scheme for scheduling on uniform processors: Using the dual approximation approach*, SIAM J. Comput. **17** (1988), no. 3, 539–551.

[Hui88] J. Y. Hui, *Resource allocation for broadband networks*, IEEE J. Selected Areas in Comm. **6** (1988), no. 3, 1598–1608.

[IMP15] Sungjin Im, Benjamin Moseley, and Kirk Pruhs, *Stochastic scheduling of heavy-tailed jobs*, 32nd International Symposium on Theoretical Aspects of Computer Science, LIPIcs. Leibniz Int. Proc. Inform., vol. 30, Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2015, pp. 474–486. MR 3356434

[JSZ85] W. B. Johnson, G. Schechtman, and J. Zinn, *Best constants in moment inequalities for linear combinations of independent and exchangeable random variables*, Ann. Probab. **13** (1985), no. 1, 234–253. MR 770640 (86i:60054)

[Kel96] Frank P. Kelly, *Notes on effective bandwidths*, Stochastic Networks: Theory and Applications, Oxford University Press, 1996, pp. 141–168.

[KRT00] Jon Kleinberg, Yuval Rabani, and Éva Tardos, *Allocating bandwidth for bursty connections*, SIAM J. Comput. **30** (2000), no. 1, 191–217. MR 1762711

[KV00] Jeong Han Kim and Van H. Vu, *Concentration of multivariate polynomials and its applications*, Combinatorica **20** (2000), no. 3, 417–434.

[Lat97] Rafał Latała, *Estimation of moments of sums of independent real random variables*, Ann. Probab. **25** (1997), no. 3, 1502–1513. MR 1457628 (98h:60021)

[LD11] Jian Li and Amol Deshpande, *Maximizing expected utility for stochastic combinatorial optimization problems*, FOCS, 2011, pp. 797–806.

[LST90] Jan Karel Lenstra, David B. Shmoys, and Éva Tardos, *Approximation algorithms for scheduling unrelated parallel machines*, Math. Program. **46** (1990), 259–271.

[LY13] Jian Li and Wen Yuan, *Stochastic combinatorial optimization via poisson approximation*, STOC, 2013, pp. 971–980.

[MSU99] Rolf H. Möhring, Andreas S. Schulz, and Marc Uetz, *Approximation in stochastic scheduling: the power of LP-based priority policies*, J. ACM **46** (1999), no. 6, 924–942. MR 1753037

[MUV06] Nicole Megow, Marc Uetz, and Tjark Vredeveld, *Models and algorithms for stochastic online scheduling*, Math. Oper. Res. **31** (2006), no. 3, 513–525.

[Pin04] Michael Pinedo, *Offline deterministic scheduling, stochastic scheduling, and online deterministic scheduling*, Handbook of Scheduling - Algorithms, Models, and Performance Analysis., 2004.

[Ros70] Haskell P. Rosenthal, *On the subspaces of L^p $(p > 2)$ spanned by sequences of independent random variables*, Israel J. Math. **8** (1970), 273–303. MR 0271721 (42 #6602)

[SS12] Warren Schudy and Maxim Sviridenko, *Concentration and moment inequalities for polynomials of independent random variables*, Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012, 2012, pp. 437–446.

[ST93] David B. Shmoys and Éva Tardos, *An approximation algorithm for the generalized assignment problem*, Math. Program. **62** (1993), 461–474.

A Useful Facts and Lemmas

Theorem A.1 (Jensen's Inequality)**.** *Let $X_1, X_2, \ldots, X_t$ be random variables and $f(x_1, \cdots, x_t)$ any convex function. Then $\mathbb{E}[f(X_1, \cdots, X_t)] \geq f(\mathbb{E}[X_1], \cdots, \mathbb{E}[X_t])$.*

Theorem A.2 (Rosenthal Inequality [Ros70, JSZ85, Lat97])**.** *Let $X_1, X_2, \ldots, X_t$ be independent non-negative random variables. Let $q \geq 1$ and $K = \Theta(q/\log q)$. Then it is the case that*

$$\mathbb{E}\left[\left(\sum_j X_j\right)^q\right] \leq K^q \cdot \max\left\{\left(\sum_j \mathbb{E}[X_j]\right)^q, \sum_j \mathbb{E}[X_j^q]\right\}.$$

Lemma A.3 (Exceptional Items Lower Bound)**.** *Let $X_1, X_2, \ldots, X_t$ be non-negative discrete random variables each taking value zero or at least L. If $\sum_j \mathbb{E}[X_j] \geq L$ then $\mathbb{E}[\max_j X_j] \geq L/2$.*

Proof. The Bernoulli case of this lemma appears as [KRT00, Lemma 3.3]. The extension to the general case is easy. For each X_j, introduce independent *Bernoulli* random variables $\{X_{jk}\}$ where each X_{jk} corresponds to a particular instantiation s_{jk} of X_j, i.e. $\Pr[X_{jk} = s_{jk}] = \Pr[X_j = s_{jk}]$. Note that $\max_k X_{jk}$ is stochastically dominated by X_j: so $\mathbb{E}[\max_j X_j] \geq \mathbb{E}[\max_{jk} X_{jk}]$. Moreover, $\sum_{jk} \mathbb{E}[X_{jk}] = \sum_j \mathbb{E}[X_j] \geq L$. So the lemma follows from the Bernoulli case. ∎

Lemma A.4 (Truncated Items Lower Bound)**.** *Let $X_1, X_2, \cdots X_n$ be independent $[0,1]$ r.v.s, and $\{J_i\}_{i=1}^m$ be any partition of $[n]$. If $\sum_{j=1}^n \beta_{1/m}(X_j) \geq 17m$ then $\mathbb{E}\left[\max_{i=1}^m \sum_{j\in J_i} X_j\right] = \Omega(1)$.*

Proof. This is a slight extension of [KRT00, Lemma 3.4], with two main differences. Firstly, we want to consider arbitrary instead of just Bernoulli r.v.s. Secondly, we use a different definition of effective size than they do. We provide the details below for completeness.

At the loss of factor two in the makespan, we may assume (by rounding down) that the only values taken by the X_j r.v.s are inverse powers of 2. For each r.v. X_j, applying [KRT00, Lemma 3.10] yields independent Bernoulli random variables $\{Y_{jk}\}$ so that for each power-of-2 value s we have

$$\Pr[X_j = s] = \Pr[s \leq \sum_k Y_{jk} < 2s].$$

Let $\overline{X}_j = \sum_k Y_{jk}$, so $X_j \leq \overline{X}_j < 2\cdot X_j$ and $\beta_{1/m}(\overline{X}_j) = \sum_k \beta_{1/m}(Y_{jk})$. Note also that $\beta_{1/m}(\overline{X}_j) \geq \beta_{1/m}(X_j)$. Hence $\sum_{jk} \beta_{1/m}(Y_{jk}) \geq \sum_{j=1}^n \beta_{1/m}(X_j) \geq 17m$. Now, consider the assignment of the Y_{jk} r.v.s corresponding to $\{J_i\}_{i=1}^m$, i.e. for each $i \in [m]$ and $j \in J_i$, all the $\{Y_{jk}\}$ r.v.s are assigned to part i. Then applying [KRT00, Lemma 3.4] which works for Bernoulli r.v.s, we obtain $\mathbb{E}\left[\max_{i=1}^m \sum_{j\in J_i}\sum_k Y_{jk}\right] = \Omega(1)$. Observe that the above lemma used a different notion of effective size: $\beta'_{1/m}(X) := \min\{s, sqm^s\}$ for any Bernoulli r.v. X taking value s with probability q. However, as shown in [KRT00, Prop 2.5], $\beta_{1/m}(X) \leq \beta'_{1/m}(X)$ which implies the version that we use here.

Finally, using $X_j > \frac{1}{2}\overline{X}_j$ we obtain

$$\mathbb{E}\left[\max_{i=1}^m \sum_{j\in J_i} X_j\right] \geq \frac{1}{2}\mathbb{E}\left[\max_{i=1}^m \sum_{j\in J_i} \overline{X}_j\right] = \frac{1}{2}\mathbb{E}\left[\max_{i=1}^m \sum_{j\in J_i}\sum_k Y_{jk}\right] = \Omega(1).$$

■

B Useful Examples

B.1 Bad Example for Simpler Effective Sizes

For stochastic load balancing on identical machines [KRT00] showed that any algorithm which maps each r.v. to a single real value and performs load balancing on these (deterministic) values incurs an $\Omega(\frac{\log m}{\log\log m})$ approximation ratio. This is precisely the reason they introduced the notion of truncated and exceptional r.v.s. For truncated r.v.s, their algorithm showed that it suffices to use $\beta_{1/m}(X_j)$ as the effective size (i.e. deterministic value) and perform load balancing with respect to these. Exceptional r.v.s are handled separately (in a simpler manner).

Here we consider the restricted assignment special case of unrelated machine scheduling. The size of each job j is a r.v. X_j which may be assigned to only a subset $S_j \subseteq [m]$ of machines. We provide an example which shows that even when all r.v.s are truncated, any algorithm which maps each r.v. to a single real value must incur approximation ratio at least $\Omega(\frac{\log m}{\log\log m})$. This suggests that more work is needed to define the "right" effective sizes in the unrelated machine setting.

There are m machines and $m + \sqrt{m}$ jobs. Each r.v. X_j takes value 1 with probability $\frac{1}{\sqrt{m}}$ (and 0 otherwise). The first $\sqrt{m}$ jobs have $S_j = \{1\}$, i.e. they can only be assigned to machine 1. The remaining m jobs have $S_j = [m]$, i.e. they can be assigned to any machine. Note that $OPT \approx 1 + 1/e$ which is obtained by assigning the first $\sqrt{m}$ jobs to machine 1, and each of the remaining m jobs in a one-to-one manner.

Given any fixed mapping of r.v.s to reals, note that all the X_j get the same value (say θ) as they are identically distributed. So the optimal value of the corresponding (deterministic) load balancing instance is $\sqrt{m} \cdot \theta$. Hence the solution which maps $\sqrt{m}$ jobs to each of the first $1 + \sqrt{m}$ machines is an optimal solution to the deterministic instance. It is easy to see that the expected makespan of this assignment is $\Omega(\frac{\log m}{\log \log m})$.

B.2 Adaptivity Gap for Identical Machines

Since we consider approximation algorithms and compare to the best solution given by an algorithm of the same nature, the adaptive and non-adaptive models are mutually incomparable, and depends heavily on the problem at hand. E.g., for makespan minimization on identical machines, Graham's list scheduling gives a trivial 2-approximation in the *adaptive* case (in fact, it is 2-approximate on an per-instance basis), whereas the NON-ADAPTIVE case is quite non-trivial where the Kleinberg et al. [KRT00] result was the first constant-factor approximation. Conversely, for the case of minimizing the weighted completion time (and other problems with linear objectives), the non-adaptive problems reduce immediately to their deterministic variants using linearity of expectations, whereas the adaptive problems are often difficult. Recall that all results in this paper were for the non-adaptive model.

Here we provide an instance of stochastic load balancing on identical machines where there is an $\Omega(\frac{\log m}{\log \log m})$ gap between the best static assignment (the setting of this paper) and the best adaptive assignment. In an adaptive assignment, a solution (or policy) decides to assign jobs to machines sequentially: the policy also gets to know the instantiated size of each job immediately after it is assigned. The instance consists of m machines and $n = m^2$ jobs each of which is identically distributed taking size 1 with probability $\frac{1}{m}$ (and 0 otherwise). Recall that Graham's algorithm considers jobs in any order and places each job on the least loaded machine. It follows that the expected makespan of this adaptive policy is at most $1 + \frac{1}{m} \cdot \sum_{j=1}^{n} \mathbb{E}[X_j] = 2$. On the other hand, the best static assignment has expected makespan $\Omega(\frac{\log m}{\log \log m})$, which is obtained by assigning m jobs to each machine.

B.3 Integrality Gap for Budgeted Matching LP

Here we show that the LP (28)-(32) used in the algorithm for budgeted GAP has an unbounded integrality gap, even if we assume that $c_{max} \ll OPT$. The instance consists of n jobs and $m = n - 1$ machines. For each machine $i \in [m]$, there are two incident edges in E: one to job i (with cost 1) and the other to job $i + 1$ (with cost n). So E is the disjoint union of two machine-perfect matchings M_1 (of total cost m) and M_2 (of total cost mn). The rewards are

$$r_j = \begin{cases} 1 & \text{if } j = 1 \\ 4 & \text{if } 2 \le j \le n-1 \\ 2 & \text{if } j = n \end{cases} .$$

and the target $R = 4(n-2) + 1 + \epsilon$ where $\epsilon \to 0$. Note that the only (minimal) integral solution involves assigning the jobs $\{2, 3, \cdots n\}$ which has total reward $4(n-2) + 2$. This solution has cost $OPT = mn$ and corresponds to matching M_2. On the other hand, consider the fractional solution $z = \epsilon \mathbf{1}_{M_2} + (1-\epsilon)\mathbf{1}_{M_1}$. This is clearly feasible for the matching constraints, and its reward is $\epsilon(4(n-2)+2) + (1-\epsilon)(4(n-2)+1) = R$. So z is a feasible fractional solution. The cost of this fractional solution is at most $m + \epsilon(mn) \ll OPT$.

C Budgeted Makespan Minimization

We now give the full details about our solution to the Budgeted Makespan Minimization problem, which we abbreviate as BUDGETSTOCMAKESPAN. Recall this is a generalization of the STOC-

Makespan problem, where now each job j also has *reward* $r_j \geq 0$. We are required to schedule some subset of jobs whose total reward at least some target value R. The objective, again, is to minimize the expected makespan. If the target $R = \sum_{j \in [n]} r_j$ then we recover the StocMakespan problem. In this section, we show:

Theorem 1.2. *There is an $O(1)$-approximation algorithm for the budgeted makespan minimization problem on unrelated machines.*

As before, using a binary-search scheme (and by scaling down the sizes), we can assume that we need to either (i) find a solution of expected makespan $O(1)$, or (ii) to prove that the optimal value is more than 1. We use a natural LP relaxation which has variables $y_{i,j}$ for each job j and machine i. The LP includes the constraints (6)-(9) for the base problem, and in addition it has the following two constraints:

$$\sum_{i=1}^{m} y_{ij} \leq 1 \qquad \forall j = 1, \cdots n, \tag{22}$$

$$\sum_{j=1}^{n} r_j \cdot \sum_{i=1}^{m} y_{ij} \geq R. \tag{23}$$

The first constraint (22) replaces constraint (5) and says that not all jobs need to be assigned. The second constraint (23) ensures that the assigned jobs have total reward at least the target R. We also perform a preprocessing step that will be useful later: for i, j pairs where $\mathbb{E}[X''_{ij}] > 2$, we force the associated y_{ij} variable to zero. (This is a valid operation, since Lemma 2.4 guarantees that any assignment with expected makespan at most 1 in fact has $\sum_i \sum_{j \in J_i} \mathbb{E}[X''_{ij}] \leq 2$.) As in §2.3 this LP can be solved in polynomial time via the ellipsoid method. If the LP is infeasible we get a proof (using Lemma 2.4) that the optimal expected makespan is more than one. Hence we may assume the LP is feasible, and proceed to round the solution along the lines of §2.4.

Recall that the rounding algorithm in §2.4 reduces the fractional LP solution to an instance of the generalized assignment problem (GAP). Here, we will use a further generalization of GAP, which we call Budgeted GAP. An instance of this problem is similar to an instance of GAP. We are given m machines and n jobs, and for each job j and machine i, we are given the processing time $p_{i,j}$ and the associated assignment cost $c_{i,j}$. Now each job j has a reward r_j, and there are two "target" parameters: the reward target R and the makespan target B. A solution must assign a subset of jobs to machines such that the total reward of assigned jobs is at least R. Moreover, as in the case of GAP, the goal is to minimize the total assignment cost subject to the condition that the makespan is at most B. Our main technical theorem of this section shows how to round an LP relaxation of this Budgeted GAP problem.

Theorem C.1. *There is a polynomial-time rounding algorithm for* Budgeted GAP *that given any fractional solution to the natural LP relaxation of cost C^*, produces an integer solution having total cost at most $C^* + c_{max}$ and makespan at most $B + 2p_{max}$.*

Before we prove this theorem, let us use it to solve the BudgetStocMakespan, and prove Theorem 1.2. Proceeding as in §2.4, we perform Steps 1-2 from the rounding procedure. This rounding gives us values $p_{i,j}$ and $c_{i,j}$ for each job/machine pair. Now, instead of reducing to an instance of GAP, we reduce to an instance $\mathcal{I}'$ of Budgeted GAP. The instance $\mathcal{I}'$ has the same set of jobs and machines as in the original BudgetStocMakespan instance $\mathcal{I}$. For each job j and machine i, the processing time and the assignment cost are given by $p_{i,j}$ and $c_{i,j}$ respectively. Furthermore, the reward r_j for job j, and the reward target R are same as those in $\mathcal{I}$. The makespan bound $B = O(1)$ (as in (8)). It is easy to check that the fractional solution $y_{i,j}$ is a feasible fractional solution to the natural LP relaxation for $\mathcal{I}'$ (given below), and the assignment cost of this fractional solution is at most 2. Applying Theorem C.1 yields an assignment $\{J_i\}_{i=1}^m$, which has the following properties:

- The makespan is at most $B + 2 = O(1)$; i.e., $\sum_{j \in J_i} p_{ij} \leq B + 2p_{max} \leq B + 2$ for each machine i. Here we used the fact that $p_{max} \leq 1$.
- The cost of the solution, $\sum_{i=1}^m \sum_{j \in J_i} c_{ij}$, is at most 4. This uses the fact $C^* \leq 2$ (from the LP solution) and $c_{max} \leq 2$ by the preprocessing on the $\mathbb{E}[X''_{ij}]$ values.
- The total reward for the assigned jobs, $\sum_{j \in \cup_i J_i} r_j$, is at least R.

Now arguing as in §2.4, we can show that the first two properties imply that the expected makespan is $O(1)$. The third property implies the total reward of assigned jobs is at least R, and completes the proof of Theorem 1.2.

C.1 Proof of Theorem C.1

Proof of Theorem C.1. Let $\mathcal{I}$ be an instance of Budgeted GAP as described above. The natural LP relaxation for this problem is as follows:

$$
\begin{aligned}
\min \quad & \textstyle\sum_{ij} c_{ij} y_{i,j} & & \\
& \textstyle\sum_j p_{ij} y_{i,j} \leq B & \forall i \quad & (24)\\
& \textstyle\sum_i y_{i,j} \leq 1 & \forall j \quad & (25)\\
& \textstyle\sum_{i,j} y_{i,j} r_j \geq R & & (26)\\
& y_{i,j} = 0 & \forall j \ s.t.\ p_{ij} > b \quad & (27)\\
& y \geq 0 & &
\end{aligned}
$$

Let $\{y_{i,j}\}$ denote an optimal fractional solution to this LP. For each machine i, let $t_i := \lceil \sum_j y_{i,j} \rceil$ be the (rounded) fractional assignment to machine i. Closely following [ST93], we construct a bipartite graph $G = (V_1 \cup V_2, E)$, where V_1 is the set of jobs (indexed by $j = 1, \ldots, n$), and V_2 consists of t_i copies for each machine i (indexed by $i' = 1, \ldots, m'$). We now transform y to a fractional matching $y'_{i',j}$ in G as follows: for machine i we sort the jobs in decreasing order of p_{ij}, assign the k^{th} unit of $\sum_j y_{i,j}$ to the k^{th} machine-copy for each $k \in [1, t_i]$. Let $y'_{i',j}$ be the resulting fractional assignment. The edges of G are (i', j) pairs in the support of y', and they inherit their costs $c_{i',j}$ from the corresponding (i, j) edge in the Budgeted GAP instance. One can check that the following properties hold:

- $\sum_{i=1}^{m'} y'_{i,j} \leq 1$ for all $j \in [n]$.
- $\sum_{j=1}^{n} y'_{i,j} \leq 1$ for all $i \in [m']$.
- The total reward of jobs assigned by y', i.e., $\sum_{j=1}^n r_j \sum_{i=1}^{m'} y'_{i,j}$ is at least R, and the total cost of this solution, $c^\intercal y' := \sum_{i,j} x_{i,j} y'_{i,j}$ is same as $c^\intercal y$.
- Any integral assignment that places at most one job on each machine-copy has makespan at most $B + p_{max}$ in the original instance $\mathcal{I}$ (where for every machine i, we assign to it all the jobs which are assigned to a copy of i in this integral assignment).

We use the following (simple) extension of the last property: if the integral assignment places *two* jobs on one machine-copy and at most one job on all other machine-copies, then it has makespan at most $B + 2p_{max}$ in the instance $\mathcal{I}$. Finally, we observe that the solution $y'_{i,j}$ is a feasible solution to the following LP (where the variables are $\{z_{i,j}\}_{(i,j)\in E}$):

$$\min \quad \sum_{ij} c_{ij} z_{i,j} \tag{28}$$

$$\sum_{i\in[m']:(ij)\in E} z_{i,j} \le 1 \; \forall j \in [n] \tag{29}$$

$$\sum_{j\in[n]:(ij)\in E} z_{i,j} \le 1 \; \forall i \in [m'] \tag{30}$$

$$\sum_{(ij)\in E} r_j \cdot z_{i,j} \ge R \tag{31}$$

$$z \ge 0 \tag{32}$$

The optimal value of this auxiliary LP is at most $c \cdot y$. We note that its integrality gap is unbounded even when c_{max} is small; see Appendix B.3. So this differs from the usual GAP where the corresponding LP (i.e. without (31)) is actually integral. However, we show how to obtain a good integral solution that violates the matching constraint for just a *single* machine-copy in V_2. Indeed, let z be an optimal solution to this LP. Note that the feasible region of this LP is just the bipartite-matching polytope on G intersected with one extra linear constraint (31) that corresponds to the total reward being at least R. So z must be a convex combination of two adjacent extreme points of the bipartite-matching polytope. Using the integrality and adjacency properties (see [BR74]) of the bipartite-matching polytope, it follows that $z = \lambda_1 \cdot \mathbf{1}_{M_1} + \lambda_2 \cdot \mathbf{1}_{M_2}$ where:

- $\lambda_1 + \lambda_2 = 1$ and $\lambda_1, \lambda_2 \ge 0$.
- M_1 and M_2 are integral matchings.
- The symmetric difference $M_1 \oplus M_2$ is a single cycle or path.

W.l.o.g. assume that $r(M_1) \ge r(M_2)$ and $c(M_1) \ge c(M_2)$, where $c(M)$ and $r(M)$ denote the total cost and reward of matching M. (Note that $r(M_1) \le r(M_2)$ and $c(M_1) \ge c(M_2)$, then M_2 would be a better solution than M_1, and we would have an integral solution.) If $M_1 \oplus M_2$ is a cycle then we output M_2 as the solution. Note that the cycle must be an even cycle: so the set of jobs assigned by M_1 and M_2 is identical. As the reward function is only dependent on the assigned jobs (and not the machines used in the assignment) it follows that $r(M_2) = r(M_1) \ge R$. So M_2 is indeed a feasible solution. Moreover, $c(M_2) \le c \cdot z$.

Now consider the case that $M_1 \oplus M_2$ is a path. If the set of jobs assigned by M_1 and M_2 are the same then M_2 is an optimal integral solution (as above). The only remaining case is that M_1 assigns one additional job (say j^* to i^*) over the jobs in M_2. Then we return the solution M_2 along with the assignment (j^*, i^*). Note that this may not be a feasible matching. But the only infeasibility is at machine-copy i^* which may have two jobs assigned; all other machine-copies have at most one job. The reward of this solution is $r(M_1) \ge R$. Moreover, its cost is at most $c(M_2) + c_{i^*j^*} \le c \cdot z + c_{max}$.

Now using this (near-feasible) assignment gives us the desired cost and makespan bounds, and completes the proof of Theorem C.1. ■