# Convex Programming for Scheduling Unrelated Parallel Machines

Yossi Azar* Amir Epstein†

**Abstract**

We consider the classical problem of scheduling parallel unrelated machines. Each job is to be processed by exactly one machine. Processing job $j$ on machine $i$ requires time $p_{ij}$. The goal is to find a schedule that minimizes the $\ell_p$ norm. Previous work showed a 2-approximation algorithm for the problem with respect to the $\ell_\infty$ norm. For any fixed $\ell_p$ norm the previously known approximation algorithm has a performance of $\theta(p)$. We provide a 2-approximation algorithm for any fixed $\ell_p$ norm ($p > 1$). This algorithm uses convex programming relaxation. We also give a $\sqrt{2}$-approximation algorithm for the $\ell_2$ norm. This algorithm relies on convex quadratic programming relaxation. To the best of our knowledge, this is the first time that general convex programming techniques (apart from SDPs and CQPs) are used in the area of scheduling. We show for any given $\ell_p$ norm a PTAS for any fixed number of machines. We also consider the multidimensional generalization of the problem in which the jobs are $d$-dimensional. Here the goal is to minimize the $\ell_p$ norm of the generalized load vector, which is a matrix where the rows represent the machines and the columns represent the jobs dimension. For this problem we give a $(d+1)$-approximation algorithm for any fixed $\ell_p$ norm ($p > 1$).

## 1 Introduction

We consider the classical problem of scheduling jobs on parallel unrelated machines. Lenstra et. al [14] and Shmoys and Tardos [16] provided a 2-approximation algorithm for minimizing the makespan ($\ell_\infty$ norm). However, for the $\ell_p$ norm only $\theta(p)$-approximation algorithm was known (see [2]). We provide a 2-approximation algorithm for any $\ell_p$ norm. In addition we show a $\sqrt{2}$-approximation algorithm for the $\ell_2$ norm.

Our approximation algorithms are based on convex programming relaxations. To the best of our knowledge, this is the first time that general convex programming techniques (apart from SDPs and CQPs) are used in the area of scheduling. Semidefinite programming (SDP) and convex quadratic programming (CQP) are special cases of convex programming (CP). Given any $\epsilon > 0$ convex programs can be solved within an additive error of $\epsilon$ under some requirements on the convex objective function and on the feasible space. This can be done through the ellipsoid algorithm (Grötschel et. al [9]) and more efficiently using interior-point methods (see e.g, Nesterov and Nemirovsky [15]).

Linear programming (LP) relaxations have been proved to be a useful tool in the design and analysis of approximation algorithms for many graph and combinatorial problems. There are several applications of linear programming relaxations for machine scheduling problems, see, e.g., [14, 16, 10, 4, 5]. The importance of semidefinite programming is that sometimes it leads to tighter relaxations for many graph and combinatorial problems. Grötchel et. al [8] used semidefinite programming to design a polynomial time algorithm for finding the largest stable set in a perfect graph. Goemans and

*School of Computer Science, Tel-Aviv University. E-Mail: azar@tau.ac.il. Supported in part by German-Israeli Foundation and by the Israel Science Foundation.

†School of Computer Science, Tel Aviv University. E-Mail: amirep@tau.ac.il. Supported in part by the German-Israeli Foundation and by the Israel Science Foundation.

Williamson [7] were the first to use semidefinite programming relaxations in the design of approximation algorithms. They used semidefinite relaxations for the problems MAXCUT, MAXDICUT and MAX2SAT. Convex quadratic programming and semidefinite programming relaxations in the area of scheduling were first used by Skutella [17]. He used convex quadratic programming relaxations to design approximation algorithms for the problem of scheduling unrelated parallel machines so as to minimize total weighted completion time of jobs.

**Techniques:** In the design of the 2-approximation algorithm for scheduling jobs on unrelated machines we formulate the problem as a convex program, where the objective function is the $\ell_p$ norm ($p > 1$). The obvious CP relaxation yields a large integrality gap. To overcome this problem we modify the objective function of the CP. The new objective function yields a bounded gap. Our rounding technique is based on the rounding technique of Shmoys and Tardos for the generalized assignment problem [16]. They used this rounding technique for the case of $\ell_\infty$ norm. However, they did not need to use the extra property of not increasing the cost. Interestingly in our rounding method for the $\ell_p$ norm we need to use this extra property. We use their technique in an enhanced manner using the additional term in the modified objective function of the convex program as the cost of the fractional weighted matching of the bipartite graph they form. Then we exploit the property That there exists an integral matching with no greater cost. This matching defines the schedule of the jobs. For the $\ell_2$ norm we use the same convex program together with randomized rounding to obtain a $\sqrt{2}$-approximation algorithm. For any $\ell_p$ norm we also design a slightly better than a 2-approximation algorithm for the restricted assignment model by using a bootstrapping of a 2-approximation algorithm. Additionally we use convex programming relaxation in the design of a PTAS to the problem.

## 1.1 Problem Definition

We have $m$ parallel machines and $n$ independent jobs, where job $j$ takes positive integral processing time $p_{ij}$ when processed by machine $i$. The load of machine is the total processing time required for the jobs assigned to it. The cost of an assignment for an input sequence of jobs is defined as the $\ell_p$ norm of the load vector. Specifically, the $\ell_\infty$ norm is the makespan (or maximum load) and the $\ell_2$ norm is the Euclidean norm, which is equivalent to the sum of the squares of the load vector. The goal of an assignment algorithm is to assign all the jobs so as to minimize the cost. Consider for example the case where the weight of a job corresponds to its machine disk access frequency. Each job may see a delay that is proportional to the load on the machine it is assigned to. Then the *average* delay over all disk accesses is proportional to the sum of squares of the machines loads (namely the $\ell_2$ norm of the corresponding machines load vector) whereas the *maximum* delay is proportional to the maximum load.

## 1.2 Our Results

We show the following results for scheduling jobs on unrelated machines:

- A 2-approximation polynomial algorithm for the general problem for any given $\ell_p$ norm ($p > 1$). This improves the previous $\theta(p)$ approximation algorithm given in [2].
- A $\sqrt{2}$-approximation polynomial algorithm for the $\ell_2$ norm for the problem, improving the previous $1 + \sqrt{2}$ approximation algorithm given in [2].
- A PTAS for fixed number of machines for any given $\ell_p$ norm with space which is polynomial in both $\frac{1}{\epsilon}$ and $m$ (and the input size).

We also consider a generalization of the problem and a special case of the problem and obtain the following results:

- We consider the case in which jobs are $d$-dimensional and show a $d+1$-approximation algorithm.
- We consider the restricted assignment model where each job has a size and should be assigned to one out of some subset of machines. For this problem we show a slightly better than a 2-approximation algorithm for any $\ell_p$ norm.

### 1.3 Previous Results

Lenstra et. al [14] and Shmoys and Tardos [16] presented a 2-approximation algorithm for the makespan, however their algorithm does not guarantee any constant approximation ratio to optimal solutions for any other norms (it is easy to come up with a concrete example to support that). For any given $\ell_p$ norm the only previous approximation algorithm for unrelated machines, presented by Awerbuch et al. [2], has a performance of $\theta(p)$ (this algorithm was presented as an on-line algorithm). Our main result is a 2-approximation polynomial algorithm for any given $\ell_p$ norm $(p > 1)$. It is also known that there is no approximation polynomial algorithm for the $\ell_\infty$ norm with ratio better then $3/2$ (see [14]). In addition the problem is $APX$-Hard for any fixed $p$ (see [3]). These hardness results hold for the restricted assignment model as well.

For the $\ell_2$ norm the only previous approximation algorithm for unrelated machines, presented by Awerbuch et al. [2], has a performance of $1+\sqrt{2}$ (this algorithm was presented as an on-line algorithm). We improve this result by providing a $\sqrt{2}$-approximation polynomial algorithm for the $\ell_2$ norm.

Recall that for the $\ell_p$ norm ($p > 1$) the problem of scheduling in the restricted assignment model is APX-hard. Thus, there is no PTAS for the problem unless ($P = NP$). However, if the number of machines is fixed a PTAS can be achieved. We present a PTAS for any given norm and any fixed number of machines with better space complexity then the FPTAS presented for the problem in [3]. Note that for minimizing the makespan Horowitz and Sahni [13] presented a FPTAS for any fixed number of machines. Lenstra *et al.* [14] suggested a PTAS for the same problem (i.e. minimizing the makespan) with better space complexity.

Our algorithm for the restricted assignment model has an approximation ratio of $2-\Omega(\frac{1}{2^p})$. Previously, Azar et al. [3] presented an all-norm 2-approximation polynomial algorithm which provides a 2-approximation to all norms simultaneously. Their algorithm gives a 2-approximation to any fixed $\ell_p$ norm. Our improved algorithm uses their algorithm recursively.

For $d$-dimensional jobs we present a $(d+1)$-approximation algorithm for any fixed dimension $d$ and any fixed $\ell_p$ norm ($p > 1$). This is in contrast to the $\theta(p)$ algorithm that can be obtained from [2].

**Other related results:** Other scheduling models have also been studied. For the identical machines model, where each job has an associated weight and can be assigned to any machine, Hochbaum and Shmoys [12] presented a PTAS for the case of minimizing the makespan. Later, Alon *et al.* [1] showed a PTAS for any $\ell_p$ norm in the identical machines model. For the related machines model, in which each machine has a speed and the machine load equals the sum of jobs weights assigned to it divided by its speed, Hochbaum and Shmoys [11] presented a PTAS for the case of minimizing the makespan. Epstein and Sgall [6] showed a PTAS for any $\ell_p$ norm in the same model.

### 1.4 Paper Structure

In Section 3 we present our approximation algorithm for any $\ell_p$ norm. In Section 4 we present an approximation algorithm for the $\ell_2$ norm. In section 5 we give for any given $\ell_p$ norm a slightly better approximation algorithm for the restricted assignment model. In section 6 we construct for any given $\ell_p$ norm a PTAS for any fixed number of machines. In Section 7 we consider the multidimensional generalization of the problem and for this problem we present an approximation algorithm for any $\ell_p$ norm.

## 2 The Approximation Algorithm ($p > 1$)

### 2.1 Convex Programming Formulation

We define the following minimization problem in the unrelated machines model, where there is a fixed processing time $p_{ij}$ associated with each machine $i = 1, \ldots, m$ and each job $j = 1, \ldots, n$. Integer solutions to the following convex program (CP1) give the optimal schedules.

$$\begin{array}{lll}
\min \sum_{i=1}^{m} t_i^p & & \\
\sum_{i=1}^{m} x_{ij} = 1 & \text{for} & j = 1, \ldots, n \\
\sum_{j=1}^{n} x_{ij} p_{ij} - t_i = 0 & \text{for} & i = 1, \ldots, m \\
x_{ij} \geq 0 & \text{for} & j = 1, \ldots, n \ , \ i = 1, \ldots, m,
\end{array}$$

where for each machine $i = 1, \ldots, m$ and each job $j = 1, \ldots, n$ the variable $x_{ij}$ denotes the relative fraction of job $j$ on machine $i$ and for each machine $i = 1, \ldots, m$ the variable $t_i$ denotes the load of machine $i$. We denote the optimum integer solution by $OPT^p$, which is the optimum $\ell_p$ norm to the power of $p$ and $OPT$ is the optimum $\ell_p$ norm. Because of solving convex program requirements, we change the linear equality constraints of the convex program to linear inequality constraints, which does not change the optimal solution value as follows

$$\begin{array}{lll}
\sum_{i=1}^{m} x_{ij} \geq 1 & \text{for} & j = 1, \ldots, n \\
\sum_{j=1}^{n} x_{ij} p_{ij} - t_i \leq 0 & \text{for} & i = 1, \ldots, m \\
x_{ij} \geq 0 & \text{for} & j = 1, \ldots, n \ , \ i = 1, \ldots, m.
\end{array}$$

This CP relaxation has large integrality gap. To overcome this problem we modify the objective function in a way which also helps us in our rounding technique as follows. Let $g(\underline{t}) = \sum_{i=1}^{m} t_i^p$ be the original objective function. We call it the $\ell_p$ norm function. Let $c(\underline{x}) = \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} p_{ij}^p$ be the cost function. Our modified objective function is $f(\underline{x}, \underline{t}) = g(\underline{t}) + c(\underline{x})$ and the objective is

$$\min f(\underline{x}, \underline{t}).$$

We call the modified convex program (CP2).

**Solvability of Convex Program:** Convex program can be solved within an additive error of $\epsilon$ in polynomial time for any given $\epsilon > 0$ when the feasible region is a convex compact set with non-empty interior (i.e. nonzero volume) under some requirements (see, e.g., [9, 15]). Specifically, it can be solved for minimizing an objective function which is convex and differentiable if we have a (polynomial time) separation oracle for the feasible set and the possibility to compute the objective and its subgradient at a given point (in polynomial time). When these requirements hold we can use for example the ellipsoid algorithm to solve the optimization minimization problem. To implement the ellipsoid algorithm we use the separation oracle for the feasible set and the fact that the objective function is convex and differentiable. Specifically, we can compute the subgradient, at any given point to obtain a separation half-space to exclude points that are not in the optimal set.

Since all the constraints in (CP2) are linear inequalities the feasible region is convex. Moreover, the separation oracle is easy to implement. Also it is easy to see that the feasible region has a positive volume. The objective function is sum of convex functions and hence convex. Moreover, it is easy to compute its value and its subgradient at a given point. Hence, (CP2) is solvable in polynomial time for any accuracy. We note that in the first proof we show how to overcome the additive error of $\epsilon$ that results from solving the convex program. In all other proofs we assume that we get the exact solution to the convex program, since we can easily overcome the additive error of $\epsilon$ as done in the first proof.

**Lemma 2.1** *Let $\underline{x}, \underline{t}$ be an optimum solution to (CP2), with value $f(\underline{x}, \underline{t})$. Then*

$$f(\underline{x}, \underline{t}) \leq 2OPT^p.$$

**Proof:** A possible feasible solution to (CP2) is the optimum integer solution to (CP1) denoted by $\underline{x}_0, \underline{t}_0$. For this solution $g(\underline{t}_0) = OPT^p$, $c(\underline{x}_0) \leq OPT^p$ and $f(\underline{x}, \underline{t}) \leq f(\underline{x}_0, \underline{t}_0)$, where the first inequality follows from the fact that for a feasible integer solution to (CP1) denoted by $\underline{x}', \underline{t}'$ holds

$$g(\underline{t}') = \sum_{i=1}^{m} (t_i')^p = \sum_{i=1}^{m} (\sum_{j=1}^{n} x'_{ij} p_{ij})^p \geq \sum_{i=1}^{m} \sum_{j=1}^{n} x'_{ij} p_{ij}^p = c(\underline{x}').$$

Hence $f(\underline{x}, \underline{t}) \leq f(\underline{x}_0, \underline{t}_0) = g(\underline{t}_0) + c(\underline{x}_0) \leq 2OPT^p$. This completes the proof. ■

## 2.2 Convex Programming Rounding

**Theorem 2.1** *The fractional solution to (CP2) can be rounded in polynomial time to an integral assignment which gives a value which is at most twice of the optimum for the $\ell_p$ norm.*

**Proof:** Given the fractional assignment $\{x_{ij}\}$ we will show how to construct the desired integral assignment $\{\hat{x}_{ij}\}$ in polynomial time. We use the same rounding algorithm used by Shmoys and Tardos for the Generalized Assignment Problem [16]. They showed how to convert a fractional solution $\{x_{ij}\}$ to an integer solution $\{\hat{x}_{ij}\}$ which satisfies the following

$$\begin{aligned} \hat{t}_i &\leq t_i + q_i \\ \sum_{i=1}^{m} \sum_{j=1}^{n} \hat{x}_{ij} c_{ij} &\leq \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} c_{ij}, \end{aligned}$$

where $q_i$ is the size the largest job assigned to machine $i$ by the integral assignment $\{\hat{x}_{ij}\}$ and $c_{ij}$ is the cost of job $j$ on machine $i$. We define $c_{ij} = p_{ij}^p$. Then we obtain $\sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} c_{ij} = \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} p_{ij}^p$ which is the cost term in the objective function of (CP2). Now we use the same rounding procedure defined above with the linear cost function defined by the coefficients $c_{ij}$. Next we show that the $\ell_p$ norm of the schedule constructed is at most twice of the optimum

$$\begin{aligned} g(\hat{\underline{t}}) &= \sum_{i=1}^{m} \hat{t}_i^p \leq \sum_{i=1}^{m} (t_i + q_i)^p \leq 2^{p-1} \sum_{i=1}^{m} (t_i^p + q_i^p) = 2^{p-1} (\sum_{i=1}^{m} t_i^p + \sum_{i=1}^{m} q_i^p) \\ &\leq 2^{p-1} (\sum_{i=1}^{m} t_i^p + \sum_{i=1}^{m} \sum_{j=1}^{n} \hat{x}_{ij} p_{ij}^p) \leq 2^{p-1} (\sum_{i=1}^{m} t_i^p + \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} p_{ij}^p) \\ &= 2^{p-1} f(\underline{x}, \underline{t}) \leq 2^{p-1} \cdot (2OPT^p + \epsilon) = 2^p OPT^p + 2^{p-1} \epsilon. \end{aligned}$$

The first inequality follows from the fact that the load on each machine $i$ is not greater then $t_i + q_i$. The second inequality follows from the inequality $(x+y)^p \leq 2^{p-1}(x^p + y^p)$. The fourth inequality follows from the fact that $\sum_{i=1}^{m}\sum_{j=1}^{n} \hat{x}_{ij}p_{ij}^p \leq \sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij}p_{ij}^p$. The last inequality follows from Lemma 2.1 and the fact that we get an optimal solution to (CP2) up to an additive error of $\epsilon$. By choosing $\epsilon = \frac{1}{2^p}$ we get

$$g(\hat{\underline{\mathrm{t}}}) \leq 2^p OPT^p + \frac{1}{2}.$$

Since $g(\hat{\underline{\mathrm{t}}})$ and $OPT^p$ are positive integral numbers we obtain

$$(g(\hat{\underline{\mathrm{t}}}))^{\frac{1}{p}} \leq 2OPT.$$

This completes the proof. ■

The proof of the following theorem appears in the Appendix.

**Theorem 2.2** *The approximation ratio provided by the approximation algorithm is lower bounded by* $2 - O\left(\frac{\ln p}{p}\right)$.

# 3 Approximation Algorithm ($p = 2$)

The modified objective function of (CP2) for the case $p = 2$ is

$$\min \sum_{i=1}^{m} t_i^2 + \sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij}p_{ij}^2,$$

which gives a convex quadratic program (CQP) that can be solved within an additive error of $\epsilon$ in polynomial time for any given $\epsilon > 0$. We use randomized rounding: each job $j$ is assigned independently at random to one of the machines with probabilities given through the values $\{x_{ij}\}$; notice that $\sum_{i=1}^{m} x_{ij} = 1$. We denote the obtained integral assignment by $\{\hat{x}_{ij}\}$.

**Lemma 3.1** *For the solution* $\{\hat{x}_{ij}\}$ *obtained by randomized rounding:*

$$E[\hat{t}_i^2] - (E[\hat{t}_i])^2 = \sum_{j=1}^{n} x_{i,j}(1 - x_{i,j})p_{i,j}^2.$$

**Proof:** We have

$$E[\hat{t}_i^2] - (E[\hat{t}_i])^2 = Var[\hat{t}_i] = \sum_{j=1}^{n} x_{i,j}(1 - x_{i,j})p_{i,j}^2.$$

The first equality follows from the definition of the variance. The second equality follows from the linearity of expectation and the independence of the indicator random variables $\hat{x}_{ij}$ and $\hat{x}_{ik}$ for $j \neq k$. This completes the proof of the Lemma. ■

**Theorem 3.1** *The fractional solution to (CQP) can be rounded in polynomial time to an integral assignment which gives a value which is at most $\sqrt{2}$ of the optimum for the $\ell_p$ norm.*

**Proof:** We have

$$\begin{aligned} E[g(\underline{\hat{t}})] &= E[\sum_{i=1}^{m} \hat{t}_i^2] = \sum_{i=1}^{m} E[\hat{t}_i^2] = \sum_{i=1}^{m} (E[\hat{t}_i])^2 + \sum_{i=1}^{m}\sum_{j=1}^{n} x_{i,j}(1-x_{i,j})p_{i,j}^2 \\ &= \sum_{i=1}^{m} t_i^2 + \sum_{i=1}^{m}\sum_{j=1}^{n} x_{i,j}(1-x_{i,j})p_{i,j}^2 \leq \sum_{i=1}^{m} t_i^2 + \sum_{i=1}^{m}\sum_{j=1}^{n} x_{i,j}p_{i,j}^2 \\ &= f(\underline{x}, \underline{t}) \leq 2OPT^2. \end{aligned}$$

The second equality follows from the linearity of expectation. The third equality follows from Lemma 3.1 and the last inequality follows from Lemma 2.1. Hence

$$(E[g(\underline{\hat{t}})])^{\frac{1}{2}} \leq \sqrt{2}OPT.$$

This completes the proof. ■

The randomized rounding algorithm can easily be derandomized by the method of conditional probabilities.

# 4 Slightly Better Approximation Algorithm for the Restricted Assignment Model ($p > 1$)

Now we turn to the restricted assignment model. For improving the 2-Approximation algorithm for the restricted assignment model we can use either the approximation algorithm presented in the previous section for unrelated machines with its rounding scheme or the 2-Approximation algorithm presented by Azar et al. [3] with its rounding scheme. When we use the first algorithm we denote by $H_1$ the schedule that contains all the jobs assigned to each machine except the biggest job assigned to each machine. We denote by $H_2$ the schedule consisting of the big jobs that are not in $H_1$. When using the latter algorithm and its rounding scheme which has two phases, we denote by $H_1$ the schedule consisting of the jobs assigned in the first rounding phase and we denote by $H_2$ the schedule consisting of the jobs assigned in the second rounding phase. This schedule assigns only one job per machine. We denote by $OPT(H_1)$ the optimal schedule of the jobs in $H_1$. We denote by $OPT$ the optimal schedule. We have:

$$\begin{aligned} \|OPT\|^p &\geq \|H_1\|^p \,, && (1) \\ \|OPT\|^p &\geq \|H_2\|^p \,, && (2) \\ \|OPT\|^p &\geq \|OPT(H_1)\|^p + \|H_2\|^p \,. && (3) \end{aligned}$$

Now we apply the 2-approximation algorithm for the jobs in $H_1$, which returns a new schedule $H_3$. We define the schedule $H$ returned by the algorithm as follows:

$$H = \begin{cases} H_1 + H_2 & \|H_1 + H_2\|^p \leq \|H_2 + H_3\|^p \\ H_2 + H_3 & \text{otherwise} \end{cases}$$

Next we prove the approximation ratio of the approximation algorithm. Let $\epsilon > 0$ to be determined. We consider two cases: If $\|H_2\|^p \leq (1-\epsilon)\|OPT\|^p$ then

$$\begin{aligned}\|H\|^p &\leq \|H_1 + H_2\|^p \leq 2^{p-1}(\|H_1\|^p + \|H_2\|^p)\\ &\leq 2^{p-1}(\|OPT\|^p + (1-\epsilon)\|OPT\|^p)\\ &= 2^{p-1}(2-\epsilon)\|OPT\|^p \ ,\end{aligned}$$

where the first inequality follows from the inequality $(x+y)^p \leq 2^{p-1}(x^p + y^p)$. If $\|H_2\|^p > (1-\epsilon)\|OPT\|^p$ we obtain the following: It follows from (3) that

$$\|OPT(H_1)\|^p \leq \epsilon\|OPT\|^p \ ,$$

hence

$$\|H_3\|^p \leq 2^p\|OPT(H_1)\|^p \leq 2^p\epsilon\|OPT\|^p \ .$$

We obtain

$$\begin{aligned}\|H\|^p &\leq \|H_2 + H_3\|^p \leq 2^{p-1}(\|H_2\|^p + \|H_3\|^p)\\ &\leq 2^{p-1}(\|OPT\|^p + 2^p\epsilon\|OPT\|^p)\\ &= 2^{p-1}(1+2^p\epsilon)\|OPT\|^p \ .\end{aligned}$$

We choose $\epsilon = 2^{p+1}$, which gives

$$\|H\|^p \leq max\left\{2^{p-1}(2-\frac{1}{2^{p+1}}), 2^{p-1}\frac{3}{2}\right\}\|OPT\|^p = (2^p - \frac{1}{4})\|OPT\|^p.$$

Hence

$$\frac{\|H\|}{\|OPT\|} \leq 2(1-\frac{1}{4p2^p}) = 2 - \frac{1}{2p2^p}.$$

The following theorem summarize the result.

**Theorem 4.1** *For the restricted assignment model there is a $2 - \frac{1}{2p2^p}$ approximation algorithm for the $\ell_p$ norm that runs in polynomial time.*

# 5 PTAS for any fixed number of machines and a given $\ell_p$ norm

We describe a polynomial time approximation scheme for any fixed number of machines $m$ and a given $\ell_p$ norm, i.e. $(1+\epsilon)$-approximation algorithm for any $\epsilon > 0$ running in polynomial time. The running time of the algorithm will be bounded by a function that is the product of $(n+1)^{m^2/\epsilon}$ and a polynomial in the size of the input. By the hardness of approximation result presented in [3], there is no approximation scheme (PTAS or FPTAS) for a given norm and any number of machines unless $P = NP$. Azar et al. [3] showed a fully polynomial time approximation scheme which is a modification of the method presented initially by Horowitz and Sahni in [13]. Our PTAS is a modification of the algorithm constructed by Lenstra et. al [14]. The significance of the new algorithm is the improvement in space usage. The space required by the old scheme is $(n+1)^{m/\epsilon}$ whereas the new scheme uses space that is polynomial in both $\frac{1}{\epsilon}$ and $m$ (and the input size).

For any $\epsilon > 0$ our algorithm is as follows: We consider a scheduling problem in the unrelated machines model, when there is a fixed processing time $p_{ij}$ associated with each machine $i = 1, \ldots, m$ and each job $j = 1, \ldots, n$. We consider the decision version of the problem with $\ell_p$ norm at most $T$.

For any schedule for the instance with value $T$, we classify the assignment of a job to a machine as either long or short, depending on whether or not the processing time in question is greater then $\epsilon T/m$. No machine can handle $m/\epsilon$ or more long assignments before time $T$. Thus, for any instance there are less than $(n+1)^{m^2/\epsilon}$ schedules of long assignments.

Consider an instance with value $T$ that has a feasible schedule. Let $t_i$ be the total processing time on machine $i$ for that schedule. This schedule includes a partial schedule of long assignments. Suppose that for machine $i$ the long assignments amount to a total processing time $l_i$, and thus the remaining jobs are completed within time $d_i = t_i - l_i$. We define $t = \epsilon T/m$ For assigning the remaining jobs we solve the following convex program

$$
\begin{array}{lll}
\min \sum_{i=1}^{m} t_i^p & & \\
\sum_{i=1}^{m} x_{ij} \geq 1 & \text{for} & j = 1, \ldots, n \\
\sum_{j=1}^{n} x_{ij} p_{ij} - t_i + l_i \leq 0 & \text{for} & i = 1, \ldots, m \\
x_{ij} \geq 0 & \text{for} & j = 1, \ldots, n\ ,\ i = 1, \ldots, m \\
x_{ij} = 0 & \text{if } \mathrm{p_{ij}} > \mathrm{t} & j = 1, \ldots, n\ ,\ i = 1, \ldots, m
\end{array}
$$

We present the following theorem, which is similar to a theorem given in [16] and has the same proof.

**Theorem 5.1** *If the convex program has a feasible solution with value less then or equals to $T^p$, then there exists a schedule with $\ell_p$ norm at most $T$, such that each machine $i$ has load of at most $t_i + t$.*

We see that the convex program must have a feasible solution, so that we can apply theorem 5.1. The resulting integral solution yields a schedule of short assignments such that the total processing time taken by short assignments to machine $i$ is at most $t_i - l_i + \epsilon T/m$. Combining this with the schedule of long assignments, we get a schedule where the total time used by machine $i$ is at most $l_i + t_i - l_i + \epsilon T/m = t_i + \epsilon T/m$. For the $\ell_p$ norm we obtain

$$\|t_i + \epsilon T/m\| \leq \|t_i\| + \|\epsilon T/m\| \leq T + \epsilon T = (1+\epsilon)T.$$

We try all possible schedules of long assignments. For each schedule of long assignments we solve the convex program and apply theorem 5.1. If we obtained a schedule using theorem 5.1, we return this schedule which has $\ell_p$ norm at most $(1+\epsilon)T$. Otherwise we answer 'no'. The following theorem gives the result.

**Theorem 5.2** *The described algorithm is a PTAS that requires time bounded by a polynomial in $m$, $\log 1/\epsilon$, and the input size.*

## 6 Approximation Scheme for Multidimensional Jobs ($p > 1$)

We generalize the problem, by defining multidimensional jobs. In the new problem we consider scheduling parallel unrelated machines. Each job is $d$-dimensional and has to be processed by exactly one machine. Processing job $j$ on machine $i$ in dimension $k$ requires time $p_{ikj}$. The goal is to find a schedule that minimizes the $\ell_p$ norm of the generalized load vector, which is a matrix where the rows represent the machines and the columns represent the jobs dimension.

### 6.1 Convex Programming Formulation

Integer solutions to the following convex program (CP3) give the optimal schedules, where the value of the $\ell_p$ norm of the optimal schedule is $T^{\frac{1}{p}}$.

$$\begin{array}{ll}
\min f(\underline{t}) = \sum_{i=1}^{m} \sum_{k=1}^{d} t_{ik}^p & \\
\sum_{i=1}^{m} x_{ij} \geq 1 & \text{for} \quad j = 1, \ldots, n \\
\sum_{j=1}^{n} x_{ij} p_{ikj} - t_{ik} \leq 0 & \text{for} \quad i = 1, \ldots, m \ , \ k = 1, \ldots, d \\
c(\underline{x}) = \sum_{i=1}^{m} \sum_{k=1}^{d} \sum_{j=1}^{n} x_{ij} p_{ikj}^p \leq T & \\
x_{ij} \geq 0 & \text{for} \quad j = 1, \ldots, n \ , \ i = 1, \ldots, m
\end{array}$$

We call the function $c(\underline{x})$ the cost function.

### 6.2 Convex Programming Rounding

**Theorem 6.1** *If (CP3) has a feasible solution with $\ell_p$ norm value at most $T^{\frac{1}{p}}$, then the fractional solution to (CP3) can be rounded in polynomial time to an integral assignment which gives a value which is at most $(d+1)T^{\frac{1}{p}}$ for the $\ell_p$ norm.*

**Proof:** Given the fractional assignment $\{x_{ij}\}$ we will show how to construct the desired integral assignment $\{\hat{x}_{ij}\}$ in polynomial time. We move to the 1-dimensional problem by defining 1-dimensional jobs $p_{ij} = \sum_{k=1}^{d} p_{ikj}$. The load of machine $i$ denoted by $t_i$ is $t_i = \sum_{k=1}^{d} t_{ik}$. Now for the new 1-dimensional problem instance, we use the fractional solution obtained for the d-dimensional instance and we perform the same rounding procedure we used for the 1-dimensional case, but now We define $c_{ij} = \sum_{k=1}^{d} p_{ikj}^p$. Then we obtain $\sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} c_{ij} = \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} \sum_{k=1}^{d} p_{ikj}^p$ which is the cost function $c(\underline{x})$. After performing the rounding we return to the original $d$-dimensional jobs. Let $\{\hat{t}_{ik}\}$ be the load of machine $i$ in dimension $k$ after the rounding, let $\{\hat{q}_{ik}\}$ be the processing time in dimension $k$ of the largest job that was matched to machine $i$ by the integral assignment $\{\hat{x}_{ij}\}$ and let $t'_{ik} = \hat{t}_{ik} - \hat{q}_{ik}$. We obtain the following:

$$\begin{aligned}
\|\hat{\underline{t}}\|_p &= \|\underline{t}' + \hat{\underline{q}}\|_p \leq \|\underline{t}'\|_p + \|\hat{\underline{q}}\|_p \leq \|d \cdot \underline{t}\|_p + \|\hat{\underline{q}}\|_p \\
&= d\|\underline{t}\|_p + \|\hat{\underline{q}}\|_p \leq dT^{\frac{1}{p}} + T^{\frac{1}{p}} = (d+1)T^{\frac{1}{p}}.
\end{aligned}$$

Let $t'_i = \sum_{k=1}^{d} t'_{ik}$. The second inequality follows from the fact that after the rounding $t'_i \leq t_i$, the pigeonhole principle and the convexity of the $\ell_p$ norm. The last inequality follows from the fact that $f(\underline{t}) \leq T$ , $c(\underline{x}) \leq T$ and the fact that $c(\hat{\underline{x}}) \leq c(\underline{x})$. This completes the proof. ■

## References


[1] N. Alon, Y. Azar, G. Woeginger, and T. Yadid. Approximation schemes for scheduling on parallel machines. *Journal of Scheduling*, 1(1):55–66, 1998.

[2] B. Awerbuch, Y. Azar, E. Grove, M. Kao, P. Krishnan, and J. Vitter. Load balancing in the $l_p$ norm. In *Proc. of the 36th Annual Symposium on Foundations of Computer Science (FOCS95)*, pages 383–391, 1995.

[3] Y. Azar, L. Epstein, Y. Richter, and G. Woeginger. All-norm approximation algorithms. In *Proc. 8th Scandinavian Workshop on Algorithm Theory (SWAT)*, pages 288–297, 2002.

[4] S. Chakrabarti, C. A. Phillips, A. S. Schulz, D. B. Shmoys, C. Stein, and J. Wein. Improved scheduling algorithms for minsum criteria. In *Proc. 23rd International Colloquium on Automata, Languages and Programming*, pages 646–657, 1996.

[5] F.A. Chudak and D.B. Shmoys. Approximation algorithms for precedence constrained scheduling problems on parallel machines that run at different speeds. *Journal of Algorithms*, 30:323–343, 1999.

[6] L. Epstein and J. Sgall. Approximation schemes for scheduling on uniformly related and identical parallel machines. In *Proc. 7th Annual European Symposium on Algorithms*, pages 151–162, 1999.

[7] M.X. Goemans and D.P Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the Association for Computing Machinery*, 42:1115–1145, 1995.

[8] M. Grötschel, L. Lovasz, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1:169–197, 1981. (Corrigendum:4(1984),291-295).

[9] M. Grötschel, L. Lovasz, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, 1988.

[10] L. Hall, A. Schulz, D. Shmoys, and J. Wein. Scheduling to minimize average completion time: Off-line and on-line approximation algorithms. *Mathematics of Operations Research*, 22:513–544, 1997.

[11] D. Hochbaum and D. Shmoys. A polynomial approximation scheme for scheduling on uniform processors: Using the dual approximation approach. *SIAM Journal on Computing*, 17(3):539–551, 1988.

[12] D. S. Hochbaum and D. B. Shmoys. Using dual approximation algorithms for scheduling problems: Theoretical and practical results. *J. Assoc. Comput. Mach.*, 34(1):144–162, January 1987.

[13] E. Horowitz and S. Sahni. Exact and approximate algorithms for scheduling non-identical processors. *Journal of the Association for Computing Machinery*, 23:317–327, 1976.

[14] J.K. Lenstra, D.B. Shmoys, and E. Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Math. Prog.*, 46:259–271, 1990.

[15] Y. Nesterov and A. Nemirovskii. Interior-point polynomial algorithms in convex programming. SIAM Studies in Applied Mathematics. SIAM, 1994.

[16] D. Shmoys and E. Tardos. An approximation algorithm for the generalized assignment problem. *Mathematical Programming A*, 62:461–474, 1993. Also in the proceeding of the 4th Annual ACM-SIAM Symposium on Discrete Algorithms, 1993.

[17] M. Skutella. Convex quadratic and semidefinite programming relaxations in scheduling. *Journal of the Association for Computing Machinery*, 48(2):206–242, 2001.

# A Appendix

## A.1 Proof of Theorem 2.2

Let $m \to \infty$. We construct problem instance for the identical machines model. We consider the machines as points in the interval $(0, 1]$, each machine is represented by a point $t \in (0, 1]$, and the load of the machines is represented as a function $f(t)$ in that interval. Let $0 < \alpha < 1$. We consider the following instance. There are infinitesimally small jobs of total volume $1 - \alpha$ and unit jobs of total volume $\alpha$. The optimal schedule has $\ell_p$ norm 1 and there is no better fractional assignment. The optimal algorithm assigns the unit jobs of total volume $\alpha$ evenly to $\alpha$ machines and assigns the infinitesimally small jobs of total volume $1 - \alpha$ evenly to the other $1 - \alpha$ machines. Suppose that the fractional schedule is as follows: The infinitesimally small jobs of total volume $1 - \alpha$ are assigned evenly to all the machines and the unit jobs of total volume $\alpha$ are also assigned evenly to all the machines ($\alpha$ fraction of a unit job is assigned to each machine). Rounding this fractional solution gives the following schedule: The infinitesimally small jobs of total volume $1 - \alpha$ are assigned evenly to all the machines and the unit jobs of total volume $\alpha$ are assigned evenly to $\alpha$ machines (one unit job is assigned to each of these machines). Let $A$ and Opt be the $\ell_p$ norms of the approximation and off-line algorithms respectively and let $C$ be the approximation ratio of the algorithm.

$$\begin{aligned} A^p &= \alpha(2-\alpha)^p + (1-\alpha)(1-\alpha)^p = \alpha(2-\alpha)^p + (1-\alpha)^{p+1} \\ Opt^p &= \alpha 1^p + (1-\alpha)1^p = 1 \\ C^p &\geq \left(\frac{A}{Opt}\right)^p = \alpha(2-\alpha)^p + (1-\alpha)^{p+1} \end{aligned}$$

We choose $\alpha = \frac{1}{p}$ and we obtain

$$C^p \geq \frac{1}{p}\left(2-\frac{1}{p}\right)^p + \left(1-\frac{1}{p}\right)^{p+1} \geq \frac{1}{p}\left(2-\frac{1}{p}\right)^p.$$

Hence

$$C \geq \left(\frac{1}{p}\right)^{\frac{1}{p}}\left(2-\frac{1}{p}\right) = e^{-\frac{\ln p}{p}}\left(2-\frac{1}{p}\right) \geq \left(1-\frac{\ln p}{p}\right)\left(2-\frac{1}{p}\right) = 2 - O\left(\frac{\ln p}{p}\right),$$

where the second inequality follows from the inequality $e^{-x} \geq 1 - x$. This completes the proof.