# Task Distributions on Multiprocessor Systems

Evgeny V. Shchepin[1⋆] and Nodari N. Vakhania[2⋆⋆]

[1] Steklov Math. Inst., 117966, Gubkina 8, Moscow
scepin@matcuer.unam.mx
[2] Faculty of Sciences
State University of Morelos
Av. Universidad 1001, Cuernavaca 62210, Mor. Mexico
nodari@servm.fc.uaem.mx

**Abstract.** We consider the problem of scheduling of $n$ independent jobs on $m$ unrelated machines to minimize the $\max(t_1, t_2, ..., t_m)$, $t_i$ being the completion time of machine $i$. In [1] was suggested a polynomial 2-approximation algorithm for this problem. It was also proved that there can exist no polynomial 1.5-approximation algorithm unless $P = NP$. Here we improve this earlier performance bound 2 to $2 - \frac{1}{m}$. In [1] is also proved a general *rounding theorem*, which allows to construct in polynomial time 1-job approximations to the optimum, i.e. schedules with an absolute bound equal to the largest job processing time. We also improve this result and obtain $(1 - \frac{1}{m})$-job approximation to optimal.

**Keywords:** approximation algorithm, distribution, independent jobs, unrelated processors, makespan

## 1 Introduction

In this paper we consider one of the classical scheduling problems. We are given $n$ tasks and $m$ unrelated parallel processors. The processing time of a task on a processor is an arbitrary real number, quite independent from the processing time of any other task on that processor and from the processing time of this task on any other processor (this is in contrast with the situation with identical or uniform processors, when task processing times are more restricted). No task preemption is allowed, each machine can process at most one task at a time and we wish to minimize $\max(t_1, t_2, ..., t_m)$, $t_i$ being the completion time of machine $i$. Even is $m = 2$ and the processors are identical, the problem is $NP$-hard [2]. Hence, no polynomial algorithm can build an optimal schedule for $m \geq 2$ processors, unless $P \neq NP$, and we try to approximate the optimum in polynomial time. For a given schedule, the optimality (or performance) ratio is defined as the ratio of the makespan of this schedule to the optimal makespan.

⋆ Partially supported by CONACYT grant 980066 and Russian Foundation of Basic Researches grant 99-01-00009

⋆⋆ Partially supported by CONACYT grant #473100-5-28937A

A schedule with the optimality ratio $k$ is called *$k$-optimal.* An algorithm with the worst-case optimality ratio $k$ is called a *$k$-approximation algorithm.*

If the processors are identical, then a linear time list scheduling algorithm which works with arbitrary precedence relations, gives a worst-case ratio $2 - \frac{1}{m}$ ([3]). An $O(n \log n)$ MULTIFIT algorithm gives the optimality ratio for identical processors 13/11 and for uniform processors $\leq 7/5$ (see [4], [5], [6]). Polynomial approximation schemes for uniform processors (the family of polynomial algorithms with optimality ratios arbitrary close to 1) were first suggested in [7].

With unrelated processors, a much weaker approximability results are known. The approximation scheme proposed in [8] is polynomial by $n$ but non-polynomial by $m$. This algorithm with the optimality ratio $1 + \varepsilon$ has time complexity $O(n^{2m}/\varepsilon)$ and its space complexity is non-polynomial. For fixed $m$, i.e., when $m$ is not an input on the problem, there is a liner by $n$ polynomial approximation scheme by Jansen and Porkolab [17]. For non-fixed $m$, the first polynomial-time approximation algorithms for unrelated processors were proposed in [9] with the optimality ratio $m$. This result was essentially improved in [10] where polynomial-time algorithms with optimality ratio within $2\sqrt{m}$ were proposed. Breakthrough in the area was due to [1] in which a polynomial algorithm with optimality ratio 2 was proposed. It was also proved that there can exist no polynomial algorithm with optimality ratio 3/2 or less, unless $P = NP$. This work was preceded by the paper [11], in which first was brought into the play the linear programming for this problem and produced an efficient but still non-polynomial by $m$ algorithm with optimality ratio 2. For a more detailed survey of the approximability results see [18].

In this paper, relying on the results from [1], we present an improved polynomial algorithm for unrelated processors with the Graham's performance bound for identical processors, i.e., with the worst-case ratio $2 - \frac{1}{m}$. This is the best result so far for $m > 2$. For $m = 2$, a linear time algorithm from [11] gives the similar result.

An *absolute error* estimates the quality of a schedule in absolute terms and is the difference between the makespan of this schedule and an optimal one. The *rounding theorem* from [1] provides with polynomial algorithms which construct schedules with an absolute error equal to the maximal job processing time $p_{max}$. We improve this result as well presenting a polynomial algorithm with an absolute error $\frac{m-1}{m}p_{\max}$. A similar result for identical processors was obtained in [12] in Theorem 1.2.

## 2 Preliminaries

In this section, we introduce the basic concepts and notations. A *schedule* assigns each task a processor, and also starting time on that processor, while a *distribution* deals only with the assignment of tasks to processors, but doesn't care about starting times of tasks on the assigned processors. In fact, the literature contains a number of results on distributions, we will mention some of them. Explicitly this concept has appeared for non-preemptive case under different names. For example, in [12] it is used the term partition, and in [10] it is used assignment. For preemptive case we will use the term distribution which seems to us more adequate.

For job $J$ and processor $M$, let us denote by $M(J)$ the time, which takes the complete execution of $J$ on $M$, and by $M_\sigma(J)$ we denote the time, during which $J$ is processed by $M$ in the schedule $\sigma$. $\frac{M_\sigma(J)}{M(J)}$ is the part of $J$ scheduled in $\sigma$ on $M$ (if preemptions are allowed, then this ratio can be any real number from the interval [0,1], and without preemptions we can have only 0s and 1s).

Every (preemptive or non-preemptive) schedule $\sigma$ defines its corresponding distribution. The distribution $\delta_\sigma$ associated with schedule $\sigma$ is a function which assigns to each pair $J, M$ the value $\delta_\sigma(J,M) = \frac{M_\sigma(J)}{M(J)} \leq 1$. For every job $J$, scheduled in $\sigma$, we have $\sum\limits_M \delta_\sigma(J,M) = 1$, where the sum is taken over all machines in $\sigma$.

The concept of distribution may be introduced and investigated independently of the concept of schedule. This concept is simpler. We introduce more formal definitions. For the convenience, let us assume that all possible *jobs* constitute an universal set of jobs which we denote by $JOBS$. This set might be finite or infinite, but all schedules and distributions are defined only on finite subsets of $JOBS$. The *processors* or *machines* are defined as functions from $JOBS$ to nonnegative real numbers $R^+$. If $M$ is a processor and $J$ is a job, $M(J)$ is the time needed to execute $J$ on $M$. The set of all processors is denoted by $PROC$. A *multiprocessor* or a *processor system* is a finite linearly ordered set of processors denoted by $\mathcal{M} = \{M_1, M_2, M_3, \ldots M_m\}$. A multiprocessor consisting of $m$ processors is called *m-multiprocessor.* A *job system* is a linearly ordered finite set of jobs $\mathcal{J} = \{J_1, J_2, \ldots J_n\}$. A job system with $n$ jobs is called *n-job system.*

A *distribution* is a function $\delta\colon JOBS \times PROC \to R^+$, such that for every job $J$, $\sum\limits_{M\in\mathcal{M}} \delta(J,M)$ is 0 or 1. Again, $\delta(J,M)$ is the part of job $J$ assigned to machine $M$. The jobs, for which this sum takes value 1 are called *distributed* in $\delta$ or *$\delta$-distributed*, and the rest of the jobs are called *non-distributed.* The set of all $\delta$-distributed jobs is denoted by $JOBS(\delta)$. If $\delta(J,M) > 0$ we will say that job $J$ is *$\delta$-distributed* on the machine $M$ (or the machine $M$ is *$\delta$-occupied* by $J$). We denote by $\delta(J)$ the set of all machines on which job $J$ in $\delta$ is distributed. The set of all $\delta$-occupied machines is denoted by $PROC(\delta)$. We will say that a *distribution $\delta$ distributes a job system $\mathcal{J}$ on a multiprocessor $\mathcal{M}$* if $\mathcal{J} = JOBS(\delta)$ and $PROC(\delta) \subset \mathcal{M}$.

Let us note that a convex combination of two distributions, (i.e. $p\delta + (1-p)\delta'$, $1 \geq p \geq 0$) is a distribution. The sum of two distributions $\delta$ and $\delta'$ is a distribution iff $JOBS(\delta) \cap JOBS(\delta') = \emptyset$. We will call such distributions *disjoint.* If inequality $\delta(J,M) \leq \delta'(J,M)$ holds for all $J, M$, then we will write $\delta \subset \delta'$ and say that $\delta$ is a *sub-distribution* of $\delta'$, and $\delta'$ is an *extension* of distribution $\delta$. In this case, as it easy to see, $JOBS(\delta) \subset JOBS(\delta')$ and $\delta(J,M) = \delta'(J,M)$ for any $M$ and $\delta$-distributed job $J$.

$M_\delta(J) = \delta(J,M)M(J)$ is the time needed for machine $M$ to execute the assigned to it in $\delta$ part of job $J$. $\sum\limits_{J\in\mathcal{J}} M_\delta(J)$ is called the *load time* of machine $M$ in distribution $\delta$ and is denoted by $|\delta|_M$. The load time of machine $|\sigma|_M$ in a schedule $\sigma$ is its load time in the associated distribution, so $|\sigma|_M = \sum\limits_{J\in\mathcal{J}} M_\sigma(J)$. Let us call the maximal load time of a machines in a distribution $\delta$ its *makespan* and denote it by $|\delta|_{\max}$. The makespan of a schedule $\sigma$ will be also denoted by $|\sigma|_{\max}$.

The distribution $\delta$ is *optimal* if it has the minimal makespan among all distributions which distribute the same set of jobs on the same multiprocessor. The problem of constructing of an optimal distribution is equivalent to the following linear programming problem:

Minimize $D_{\text{opt}}$

$$\sum_{i=1}^{n} x_{i,j}t_{i,j} \le D_{\text{opt}}, \quad \sum_{j=1}^{m} x_{i,j} = 1, \quad i = 1,...,n \quad j = 1,\dots m, \quad x_{i,j} \ge 0$$

To see this equivalence, we let $t_{i,j} = M_j(J_i)$, $x_{i,j} = \delta(J_i, M_j)$. For further references, we abbreviate this linear programming problem by $LP(D_{\text{opt}})$.

Construction of an optimal schedule can be split into two stages. On the first stage we construct an optimal distribution, and on the second stage we construct an optimal schedule with this distribution. If the distribution is non-preemptive, the second stage is trivial. The problem of construction of an optimal schedule, associated with the given distribution, was thoroughly investigated in [13]. The concept of an *open shop*, introduced in this paper, is almost the same as that of a distribution. Let $\delta$ be a distribution of $\mathcal{J}$ on $\mathcal{M}$. Then one can interpret every $J \in \mathcal{J}$ as a job consisting of $m$ subtasks, where task number $i$ has to be performed on processor $M_i$ and its execution takes the time $(M_i)_\delta(J)$. In this way every distribution generate an open shop and vice-versa.

Unfortunately, not every preemptive distribution has an associated schedule with the same makespan. The *processing time* $|\delta|^J$ of a job $J$ in a distribution $\delta$ is defined as the total time during which this job is processed on all machines, i.e. $|\delta|^J = \sum\limits_{M} M_\delta(J)$. Let us denote by $|\delta|^{\max}$ the maximum of $|\delta|^J$. So $|\delta|^{\max}$ is the largest processing time in $\delta$. Since each job has to be performed *sequentially*, i.e., a job cannot be processed at any moment by more than one machine, the makespan of every schedule $\sigma$ associated with a given distribution $\delta$ cannot be less than $|\delta|^{\max}$. Let us call the *sequential makespan* of a distribution the maximum between $|\delta|_{\max}$ and $|\delta|^{\max}$. We see that the makespan of every schedule associated with a given distribution cannot exceed its sequential makespan. Now, a principal result from [13] can be formulated as follows.

**Theorem Gonzales and Sahni** [13]) *There exists a polynomial algorithm which constructs for every distribution an associated schedule with makespan equal to the sequential makespan of this distribution.*

Based on this theorem, in [14] the problem of constructing of an optimal preemptive schedule is reduced to the following linear programming problem $LP(S_{\text{opt}})$:

Minimize $S_{\text{opt}}$,

$$\sum_{i=1}^{n} x_{i,j}t_{i,j} \leq S_{\text{opt}} \quad \sum_{j=1}^{m} x_{i,j}t_{i,j} \leq S_{\text{opt}},\ x_{i,j} \geq 0,$$

$$\sum_{i=1}^{n} x_{i,j} = 1, \quad i = 1, ..., n, \quad j = 1, ..., m$$

Any solution of this problem gives a distribution with the minimal sequential makespan.

## 3 Previous Results on Rounding

Thus the construction of an optimal preemptive distribution is polynomially solvable in opposite to the non-preemptive case, which is a subject of our study. Let us first give the *rounding approach*, first applied in [11] and later essentially improved in [1]. This approach allows us to produce good approximation to an optimal non-preemptive schedule.

For a real $x$, let $[x]$ and $\{x\}$ be its integral and fractional parts, respectively (we note that we use $\{.\}$ for representation of sets as well). So $x = [x] + \{x\}$, $[x]$ is integer and $0 \leq \{x\} < 1$. $[\delta(J, M)]$ and $\{\delta(J, M)\}$ define distributions $[\delta]$ and $\{\delta\}$, which we will call the *integral* and the *fractional* parts, of $\delta$, respectively; clearly, $[\delta]$ and $\{\delta\}$ are disjoint. A distribution is *integral* or *non-preemptive* if its fractional part is 0, and in this case it coincides with its integral part. Jobs distributed by $\{\delta\}$ are said to be *preempted* in $\delta$. An integral distribution $\delta$ is a *rounding* of another distribution $\delta'$ if it distributes the same jobs and $\delta = [\delta']$. So $\delta' = \delta + \delta_0$, where $\delta_0 = \{\delta'\}$.

To find a non-preemptive distribution, close to an optimal one, the rounding method looks for an optimal *extremal* preemptive distribution and *rounds* it. A distribution $\delta$ is *extremal* if it cannot be represented in the form $\frac{1}{2}(\delta' + \delta'')$, where $\delta'$ and $\delta''$ are different distributions, such that for every machine $M$, $|\delta'|_M = |\delta''|_M = |\delta|_M$. The importance of extremal distributions shows the following

**Extremality Principle**. *All distributions constructed by linear programming solution of $LP(D_{\text{opt}})$ are extremal.*

Proof. Denote by $\varDelta$ the set of all distributions of an $n$-job system $\mathcal{J}$ on an $m$-multiprocessor $\mathcal{M}$. This set represents a convex (possibly unbounded) polytope in space $R^{nm}$. In $LP(D_{\text{opt}})$, consider a subset $\varDelta'$ of product $R^{mn} \times R$ defined by conditions $\varDelta' = \{(\delta, D) \mid \delta \in \varDelta, D \in R, |\delta|_{\max} \leq D\}$.

If $\delta = \frac{1}{2}(\delta_1 + \delta_2)$ where $|\delta|_M = |\delta_1|_M = |\delta_2|_M$ for all machines $M$, then $(\delta, D) \in \varDelta'$ implies $(\delta_i, D) \in \varDelta'$ for $i = 1, 2$. Hence, $(\delta, D)$ is a middle point of $(\delta_1, D)$ and $(\delta_2, D)$, and therefore is not a vertex of $\Delta'$. But the linear programming works only with vertices of $\Delta'$. Hence they work only with extremal distributions.

The rounding method is based on the following known principle (see [15], [11], [1]), which will be proved the next section.

**The Preemption Bounding Principle**. *If $\delta$ is an extremal distribution on a multiprocessor $\mathcal{M} = \{M_1 \dots M_m\}$, then the number of jobs in $JOBS(\{\delta\})$ (i.e., preempted jobs in $\delta$) is less than $m$.*

Let us remark that for an integral distribution $\delta$, $\delta(J)$ represents a unique machine. Let us say that an integral distribution $\delta$ is $1-1$ if $\delta(J) \neq \delta(J')$ for every $\delta$-distributed jobs $J$ and $J'$. Since the number of preempted jobs in an extremal distribution does not exceed the number of machines (see the Preemption Bounding Principle), we can accomplish a $1-1$ *rounding*, i.e., to find a distribution $\delta'$, for which $\delta' - \delta$ is a $1-1$ distribution. In particular, applying $1-1$ rounding to an optimal extremal distribution, we immediately obtain the following result ($p_{max}$ below is the maximal task processing time):

**A 1-job Approximation Theorem**. *It is possible to construct a distribution with the makespan, exceeding the optimal makespan by no more than $p_{\max}$ in polynomial time.*

As we will see below, even an optimal $1-1$ rounding can be accomplished in polynomial time. Let us define the *selection problem* as follows. We say that a given family of subsets $\{X_i\}_{i=1,\dots,k}$ of a set $X$ is *selectable* if there exist such sequence of point $x_1, \dots x_k \in X$ (called *selection*), that $x_i \in X_i$ for all $i \leq k$ and all $x_i$ are distinct.

This selection problem can be easily solved via the *complete matching* problem in a bipartite graph $\{V_1, V_2, E\}$, with vertices $V_1 = X$ and $V_2 = \{1, 2, \dots k\}$, where pair $x, i$ is an edge iff $x \in X_i$. This matching problem is known to be polynomially solvable via the maximal flow algorithm.

Due to the Hall's Marriage Theorem (see, for example [16]), our selection problem has no solution (i.e., there is no complete matching) iff there exists a subset $Y \subset X$ with the number of elements, less than the number of elements in the specially defined subset of $V_2$. In particular, this subset contains an element $l \in V_2$ iff there exists $y \in Y$, such that $y \in X_l$.

**Lemma 1.** *Let $\mathcal{J}$ be an n-job system, $\mathcal{M}$ be an m-multiprocessor, $m \geq n$ and $\{c_i\}_{i \leq n}$ be real numbers. Then it is possible to construct in polynomial time a $1-1$ distribution $\delta$ of $\mathcal{J}$ on $\mathcal{M}$, such that $|\delta|_{M_i} \leq c_i$ for all $i$, or to prove that such distribution does not exist.*

Proof. For every job $J$ let $\mathcal{M}(J) = \{M \in \mathcal{M} \mid M_i(J) \leq c_i\}$. The constriction of $1-1$ distribution with $|\delta|_{M_i} \leq c_i$ is equivalent to the selection problem for the family $\mathcal{M}(J)$.

This lemma with lemma 1 in [1] gives us the following:

**Theorem 1.** *Let $\mathcal{J}$ be an n-job system, and $\mathcal{M}$ be an m-multiprocessor, such that $n \leq m$. Then the optimal $1-1$ -rounding can be constructed in polynomial time.*

## 4 Acyclicity of Distributions

In [1] it is considered an optimization problem for the distributions with the additional restrictions, which forbid some jobs to be distributed on some machines. The linear programming problem corresponding to this *restricted optimization problem* can be obtained from $LP(D_{\text{opt}})$ if some inequalities of the type $x_{ij} \geq 0$ are changed to the equalities $x_{ij} = 0$ ($x_{ij} = 0$ eliminates the possibility of distribution of any part of $i$th job to $j$th machine). The Extremality Principle from Section 3 also holds for this problem; the proof of this fact is similar to that of Section 3.

To specify a restricted optimization problem, we assign to each job $J$ the set of machines $\mathcal{M}_J$, the ones, on which it is allowed to distribute $J$. We shall call such assignment a *job-machine configuration.* We will use $\mathcal{C}$ to denote the configurations. Formally, a job-machine configuration $\mathcal{C}$ is a multi-valued mapping $\mathcal{C}\colon JOBS \to PROC$. The set of machines, on which job $J$ is distributed in $\mathcal{C}$ is denoted by $\mathcal{C}(J)$. We will consider only *finite* configurations; for some $J$s, $\mathcal{C}(J)$ might be empty.

A distribution $\delta$ is called *restricted* by configuration $\mathcal{C}$ or $\mathcal{C}$*-restricted* if $\delta(J) \subset \mathcal{C}(J)$ for all $J \in JOBS$. The set of $\mathcal{C}$-restricted distributions is convex. A distribution of a job system $\mathcal{J}$ with the minimal makespan, among all $\mathcal{C}$-restricted distributions

For a distribution $\delta$, in [1] the so called *configuration graph* $G(\delta)$ is introduced. $G(\delta)$ is a bipartite graph $\{V_1, V_2, E\}$, such that $V_1 = JOBS(\delta)$, $V_2 = PROC(\delta)$ and there is an edge, corresponding to the pair $J, M$ in $G(\delta)$ iff $\delta(J, M) > 0$. We will call a distribution *connected* if its configuration graph is connected.

A sub-distribution $\delta'$ of a distribution $\delta$ is called its *component* if $G(\delta')$ is component of connectedness of $G(\delta)$. From the definitions immediately follows

**Lemma 2.** *For different components $\delta_i$ and $\delta_j$ the respective sets of occupied machines are disjoint, i.e., $PROC(\delta_i) \cap PROC(\delta_j) = \emptyset$.*

**Lemma 3.** *Let $\delta$, $\delta'$ and $\delta''$ be such distributions that $PROC(\delta) \cap PROC(\delta') = PROC(\delta) \cap PROC(\delta'') = \emptyset$ and $|\delta'|_M = |\delta''|_M$ for all machines. Then $|\delta' + \delta|_M = |\delta'' + \delta|_M$ for all $M$.*

**Lemma 4.** *If $\delta = \frac{1}{2}(\delta' + \delta'')$ then $\delta(J) = \delta'(J) \cup \delta''(J)$ for all $J$.*

The proofs are left to the reader.

**Lemma 5.** *A distribution is extremal iff all its components are extremal.*

Proof. Suppose $\delta$ has a non-extremal component $\delta_0$ such that $\delta_0 = \frac{1}{2}(\delta_0' + \delta_0'')$ and $|\delta_0|_M = |\delta_0'|_M = |\delta_0''|_M$. Then processors occupied in $\delta_0'$ and $\delta_0''$ are included in $PROC(\delta_0)$. Let $\delta_1 = \delta - \delta_0$. Then $PROC(\delta_0) \cap PROC(\delta_1) = \emptyset$. Further, let $\delta' = \delta_0' + \delta_1$ and $\delta'' = \delta_0'' + \delta_1$. Then $\delta = \frac{1}{2}(\delta' + \delta'')$ and $|\delta'|_M = |\delta''|_M = |\delta|_M$ for all $M$. Now the load times are equal because of lemma 3.

In opposite direction, suppose $\delta$ is not extremal and consider its representation $\delta = \frac{1}{2}(\delta' + \delta'')$ where $|\delta|_M = |\delta'|_M = |\delta''|_M$ for all $M$. Let $J$ be a job for which $\delta'(J) \neq \delta''(J)$ and $\delta_0$ be a component of $\delta$ for which $J \in JOBS(\delta_0)$. Denote by $\delta_0'$ and $\delta_0''$ restrictions of $\delta'$ and $\delta''$ respectively, on $JOBS(\delta_0)$. Then $\delta_0 = \frac{1}{2}(\delta_0' + \delta_0'')$ and to prove the non-extremality of $\delta_0$ it is sufficient to check equality of the load times. Let $M \in PROC(\delta_0)$. As $\delta'(J) \subset \delta(J)$ for all $J \notin JOBS(\delta_0)$, we obtain $M \notin \delta'(J)$. Hence, all jobs distributed by $\delta'$ on $M$ belong to $JOBS(\delta_0)$ and are distributed by $\delta_0'$. Therefore, $|\delta_0'|_M = |\delta'|_M = |\delta_0|_M$. If $M \notin PROC(\delta_0)$, then for all $J \in JOBS(\delta_0)$ $M \notin \delta(J)$ and hence $M \notin \delta'(J)$. This implies that the load time of $M$ in $\delta_0'$, as well as in $\delta_0$, is 0. Thus we proved equality of the load times for $\delta_0'$. We use the similar reasoning for $\delta_0''$ and the lemma is proved.

Let us say that a distribution $\delta$ is *synchronous* over multiprocessor $\mathcal{M}$ if $|\delta|_M = |\delta|_{M'}$ for all $M, M' \in \mathcal{M}$.

**Lemma 6.** *A connected $\mathcal{C}$-optimal distribution $\delta$ is synchronous over $PROC(\mathcal{C})$.*

Proof. Suppose $\delta$ is not synchronous and let $M$ be a machine for which $|\delta|_{\max} > |\delta|_M$. Consider a machine $M'$ with the maximal load time. Since $G(\delta)$ is connected, there exists a path in it connecting $M$ and $M'$. Let $M = M_1, J_1, M_2, J_2, \ldots, M_{k+1} = M'$ be such a path and let $i$ be the maximal index, such that $|\delta|_{M_i} < |\delta|_{\max}$. $J_i$ occupies both $M_i$ and $M_{i+1}$. Let us choose $\varepsilon > 0$ so small that $\varepsilon < \delta(J_i, M_{i+1})$ and $\varepsilon M_i(J_i) + \delta(J_i, M_i) < 1$, and define a new distribution $\delta'$ as follows. $\delta'(J_i, M_i) = \delta(J_i, M_i) + \varepsilon$, if $\delta'(J_i, M_{i+1}) = \delta(J_i, M_i) - \varepsilon$ and $\delta'(J, M) = \delta(J, M)$ for any other job-machine pair.

The obtained distribution $\delta'$ contains less machines with the load time $|\delta|_{\max}$, and has the same configuration graph as $\delta$. Repeating the above procedure, we can construct a distribution with the same configuration as $\delta$ and with a smaller makespan. But this contradicts the optimality of $\delta$. The lemma is proved.

The number of machines in $\delta(J)$ minus 1 will be called the *number of preemptions* of $J$ in $\delta$ and will be denoted by $\pi_\delta(J)$. $\pi(\delta) = \sum\limits_{J \in \mathcal{J}} \pi_\delta(J)$ is the total number of preemptions in $\delta$.

**Lemma 7.** *The total number of preemptions in every connected $\mathcal{C}$-optimal extremal distribution $\delta$ is strictly less than the number occupied machines in $\delta$.*

Proof. Let $\delta$ occupy machines $M_1, \ldots M_m$ and let $J_1, J_2, \ldots J_k$ be the preempted jobs in $\delta$. For every $i \leq k$, let $j(i)$ be the first $j$ for which $M_j(J_i)$ is fractional. Denote by $P$ the set of pairs $i, j$, $j \neq j(i)$ and such that $0 < M_i(J_j) < 1$. The number of elements in $P$ is $\pi(\delta)$. Let $\varepsilon = \min\{M_i(J_j)\}_{(i,j) \in P}$. For every real function $f$ on $P$, such that $|f(i,j)| \leq \varepsilon/m$, let $\delta_f(J_i, M_{j(i)}) = \delta(J_i, M_{j(i)}) - \sum\limits_{j|(i,j) \in P} f(i,j)$, $\delta_f(J_i, M_j) = \delta(J_i, M_j) + f(i,j)$ if $(i,j) \in P$, and let $\delta_f(J, M) = \delta(J, M)$ otherwise. The distribution $\delta_f$, as it follows from its definition, is $\mathcal{C}$-restricted. To define a linear mapping $l$ of a $\varepsilon/m$-cube of Euclidean space $Q$ of dimension $\pi(\delta)$ into $(m-1)$ dimensional space, enumerate pairs in $P$. Then each point $x \in Q$ corresponds to a real function $f_x$ on $P$, and we can define

$l(x)$ as a vector in which $i$th component is $|\delta_{f_x}|_{M_i} - |\delta|_{M_i}$. If $\pi(\delta) \geq m$, then the kernel of the mapping $l$ is nontrivial. Let $y \in Q$ be a nonzero point for which $l(x) = 0$ and let $f$ be the function corresponding to $x$. If $|\delta_f|_{M_m} = |\delta|_{M_m}$, then $|\delta_{-f}|_{M_m} = |\delta|_{M_m}$ and we come to a contradiction with the extremality of $\delta$, because $\delta = \frac{1}{2}(\delta_{-f} + \delta_f)$. The optimality of $\delta$ implies that $|\delta_f|_{M_m} > |\delta|_{M_m}$. Indeed, if $|\delta_f|_{M_m} < |\delta|_{M_m}$, then $\delta_f$ is as well optimal, but not synchronous. But the same reasons applied to $\delta_{-f}$ gives us the similar inequality $|\delta_{-f}|_{M_m} > |\delta|_{M_m}$. Then we come to a contradiction with $\delta = \frac{1}{2}(\delta_{-f} + \delta_f)$. The lemma is proved.

We will call distribution *acyclic* if its configuration graph is acyclic. Let us say that a distribution $\delta$ is *componentwise $\mathcal{C}$-optimal* if each its component is $\mathcal{C}$-optimal. Our main result now can be formulated as follows.

**Theorem 2.** *Every componentwise optimal and extremal distribution is acyclic.*

Proof. If a distribution is connected, then the number of its preemptions is less than the number of the corresponding occupied machines (by lemma 7). Hence its configuration graph has less edges than vertices. For a connected graph this implies the acyclicity. If the distribution is not connected, then consider its components. They are extremal by lemma 5. They are as well $\mathcal{C}$-optimal by our assumption. Hence the same argument shows their acyclicity.

## 5 Consolidation of Distributions

Let us say that a non-preemptive (integral) distribution $\delta$ is a *consolidation* of a preemptive distribution $\delta'$, if it has the same domain and $\delta(J, M) = \delta'(J, M)$, for all $J$ which are not preempted in $\delta'$.

The main result of this section is the following theorem

**Theorem 3.** *For every acyclic distribution $\delta$ on an $m$-processor, it is possible to construct in polynomial time a consolidation $\delta'$, such that* $|\delta'|_{\max} \leq |\delta|_{\max} + \frac{m-1}{m} p^{\delta}_{\max}$

The proof is based on some delicate considerations connected with graphs. Besides the introduced earlier configuration graph from [1], we shall consider another type of graph for presenting the preemptive structure of the distributions. We call it a *preemption graph* and denote it by $G_\delta$. $G_\delta$ has less nodes and edges than the corresponding configuration graph.

The nodes of $G_\delta$ represent the machines, and edges represent the jobs. There is an edge in joining a pair of nodes in $G_\delta$, if the job which represents this edge is shared by the machines which represent these nodes. The preemption graph, in general, is a multi-graph. But if the configuration graph is acyclic, then the corresponding preemption graph is simple. Indeed, it is sufficient to prove that $\delta(J_1) \cap \delta(J_2)$ cannot contain two machines. But if it contains two machines $M_1$ and $M_2$, then we will have a nontrivial cycle $M_1, J_1, M_2, J_2$ in $G(\delta)$.

Preemption graph may have cycles even if configuration graph is acyclic. For example, if $\delta(J)$ contains three machines $M_1$, $M_2$ and $M_3$, then the vertices corresponding to $M_i$, $i = 1, 2, 3$, form a cycle in $G_\delta$.

A *reduced preemption graph* $G'_\delta$ has less edges than $G_\delta$ has and this graph is acyclic iff $G(\delta)$ is acyclic. The structure of $G'_\delta$ (unlike that of $G_\delta$ and $G(\delta)$) depends not only on the distribution $\delta$, but also on the order, in which the machines are numbered. $G'_\delta$ is a subgraph of $G_\delta$, obtained by deleting the so called *redundant edges* in $G_\delta$. Again, whether an edge is redundant or not, depends on the order in $\mathcal{M}$. An edge $(x, y) \in G_\delta$ is redundant, if there exist two (or more) intermediated edges $(x, z)$ and $(z, y)$, such that the index of machine $z$ is more than that of machine $x$ and less than that of machine $y$, and all $x, y$ and $z$ share the same job.

**Proposition 1.** *$G'_\delta$ is acyclic iff $G(\delta)$ is acyclic. The number of edges in $G'_\delta$ is equal to the number of preemptions in the distribution $\delta$.*

**Lemma 8.** *Let $P_1, P_2, \ldots, P_k$ be connected subgraphs of an acyclic graph $G$, having in common at most one node. Then for some $i$, the intersection of $P_i$ with $\cup P_j, j = 1, 2, ..., k, j \neq i$, is a single node.*

Proof. Let $N_1, \ldots N_k$ be all nodes of $G$, which are common for some pairs of our connected subgraphs, and let $G'$ be the minimal connected subgraph of $G$ containing all $N_i$s. The acyclicity of $G$ implies the acyclicity of $G'$. Besides, $G'$ does not have any single degree node, different from some $N_i$. Indeed, if $N$ were a single degree node in $G'$ different from any $N_i$ then we would reduce $G'$ by eliminating $N$ and the corresponding edge.

Let $N$ be a single-degree node of $G'$ and $e$ be the edge in $G'$, corresponding to $N$. As $N = P_i \cap P_j$ for some $i, j$, either $P_i$ or $P_j$, suppose $P_i$, does not contain $e$. Suppose that $P_i$ contains a node $N_j \neq N$. In this case we will have two different paths between $N_j$ and $N$ in $G$. The first such path is contained in $P_i$ and does not pass through $e$, and the second one is in $G'$ and passes through $e$. But then we would have a cycle in $G$ which is a contradiction. Therefore, the intersection of $P_i$ with the union of the rest of our connected subgraphs is exactly $N$ and the lemma is proved.

A *weight function* on the graph $G$ is a function $w$ which assigns to each node $N$ of $G$ a nonnegative number $w(N)$, and such that the sum of all $w(N)$ is equal to 1. Let us say that a weight function $w$ is *supported* by a subgraph $P$ of $G$ if it takes value 0 for all nodes in $G \backslash P$.

**Lemma 9.** *Let $G$ be an acyclic graph with $m$ nodes, $P_1, P_2, \ldots, P_k$ its covering by connected subgraphs which intersect in more than in one node. Further, let $w_1, w_2, \ldots, w_k, k < m$ be a weighted function on $G$ such that $w_i$ is supported by $P_i$. Then it is possible in polynomial time to construct a sequence of nodes $s_1, \ldots, s_k$, such that $s_i \in P_i$ for all $i$, and $\sum\limits_{s_i = N} (1 - w_i(N)) \leq 1 - \frac{1}{m}$*

Proof. Applying the above lemma we can first order subgraphs $P_1, P_2, \ldots P_k$ of our decomposition in such a way that for all $i < k$, $P_i$ intersect the union $P^i = \cup_{j>i} P_j$ in exactly one node, which we denote by $N_i$.

In the process of construction, we will need to keep an additional information for every node. An array of nonnegative integers $v(N)$. The algorithm is the following.

Step 1. (initialization) $v(N) := 0$ for all $N$, $j := 0$ (counter of cycles)

Step 2. Search for a node $M$ in $P_i$, different from $N_i$ and such that

$$w_i(M) \geq v(M)/m + 1/m,$$

(case 1): if such a node is founded then $s_i := M$

(case 2): otherwise, $s_i = N_i$, $v(N_i) := v(N_i) + \sum_{M \in P_i \setminus N_i} (v(M) + 1)$

Step 3. if $j < k$ then j:=j+1; goto step 2; else STOP

To prove correctness of this algorithm, first note that $\sum_{M \in P^i} v(M)$ is no more than the number of nodes in complement to $P^i$, for all stages in our construction. Indeed, from $P^{i-1}$ to $P^i$ we increase $\sum_{M \in G'} v(M)$ exactly by the number of the deleted nodes from $P_i$, or we do not change it at all.

The algorithm works without stopping up to the last step. When we choose $s_k$ there is no $N_k$. So we have to find an $M$ satisfying inequality $w_k(M) \geq v(M)/m + 1/m$. Suppose that there is no such $M$. In this case, for all $M$ we have $w_k(M) < v(M)/m + 1/m$. If we sum these inequalities, for all $M \in P_k$, we obtain $1 < \frac{1}{m}\sum(v(M)+1)$. But $\sum(v(M)+1)$ as already noted, does not exceed $m$ and we came to a contradiction.

Let us note that if $s_i$ is different from $N_i$, then $w_i(M) \geq v(M)/m + 1/m$ and this point cannot be chosen in the sequel. On the other hand, during the whole process, for all $i$ we have that

$$\sum_{j:s_j=N} (1 - w_j(N)) \leq v(N)/m,$$

because $v(N)$ is increased only when $N = N_i$ is selected, for some $P_j$, and it is increased by $\sum_{M \in P_i \setminus N} (v(M) + 1)$. But the condition of selection of $N_i$ is that $w_i(M) < v(M)/m + 1/m$ for all $M \in P_j$. And sums of these inequalities provide desired results. To finish our prove it is sufficient to note that $v(N) \leq m - 1$ for all $N$.

**Proof of Theorem 3** Let $G$ be a reduced preemption graph of fractional part of our distribution. This graph is subgraph of $G'_\delta$ and therefore is acyclic. Let $J_1, \ldots J_k$ be all jobs preempted in $\delta$. For every job $J_i$ denote by $P_i$ the subgraph which edges correspond to $J_i$. For every node $M$ of this subgraph, let $w_i(M) = \delta(J_i, M)$. Then we obtain decomposition $\{P_i\}$ of $G$ and the system of weights. By lemma 9 we choose for each job $J_i$, a machine $M(i)$. Now we define $\delta'(J_i, M(i)) = 1$ for all $i$. This completely defines this consolidation. The increase of the load time on machine $M$ in $\delta'$ compared with that in $\delta$ is equal to $\sum_{i:M(i)=M} (1 - \delta(J_i, M))M(J_i)$. As $M(J_i) \leq p^\delta_{\max}$ for all $i$, this sum does not exceed $(1 - \frac{1}{m})p^\delta_{\max}$. The theorem is proved.

## 6 The Worst-Case Bounds

As a simple consequence of the acyclicity and consolidation theorems, we build the approximation algorithms in this section.

Given a configuration $\mathcal{C}$ and job system $\mathcal{J}$, construct by the linear programming a $\mathcal{C}$-optimal distribution $\delta$ of $\mathcal{J}$. Decompose it into components $\delta = \sum \delta_i$. For all $i$, construct by linear programming extremal $\mathcal{C}$-optimal distributions $\delta'_i$ of $JOBS(\delta_i)$. Then the sum $\sum \delta'_i$ represents an extremal componentwise $\mathcal{C}$-optimal distribution of $\mathcal{J}$. This distribution will be acyclic owing to the Acyclicity Theorem. Now applying consolidation algorithm of Theorem 3, we obtain a distribution which properties are given in the next theorem. Denote by $p^{\mathcal{C}}_{\max}$ maximum of $M(J)$, where $M \in \mathcal{C}(J)$, and denote by $D^{\mathcal{C}}_{\text{opt}}$ the makespan of $\mathcal{C}$-optimal distribution of $\mathcal{J}$.

**Theorem 4.** *For every job system $\mathcal{J}$ and every configuration $\mathcal{C}$ it is possible to construct in polynomial time an integral distribution $\delta$, such that*

$$|\delta|_{\max} \leq D^{\mathcal{C}}_{\text{opt}} + \frac{m-1}{m} p^{\mathcal{C}}_{\max}$$

If we consider a *constant configuration*, i.e., a configuration $\mathcal{C}$, such that $\mathcal{C}(J) = \mathcal{M}$ for all $J \in \mathcal{J}$, then, from the above theorem, we immediately obtain the following improved version of the 1-job approximation theorem.

**Corollary 1.** *There is a polynomial algorithm which constructs an integral distribution of a job system $\mathcal{J}$ on a multiprocessor $\mathcal{M}$ with a makespan, exceeding the optimal one by no more than $\frac{m-1}{m} p_{\max}$.*

Now we find it useful to recall some basic ideas behind the earlier mentioned polynomial 2-approximation algorithm from reference [1]. One of the useful concepts, introduced in [1] was what we call the *balanced distribution*, a distribution, in which the jobs cannot be distributed on the machines, on which their execution time is "sufficiently large". More precisely, for a distribution $\delta$, let us denote by $p^{\delta}_{\max}$ the maximum of $\{M(J) \mid \delta(J,M) > 0\}$ and call it the *$\delta$-largest processing time.* Let us say that for some $B \in R^+$ $\delta$ is $B$-balanced, if $|\delta|_{\max} \leq B$ and $p^{\delta}_{\max} \leq B$. Denote by $B_{\text{opt}}$ the minimum $B$ for which there exist a $B$-balanced distribution. Note that $D_{\text{opt}} \leq B_{\text{opt}} \leq D^{\text{opt}}$. Indeed, every non-preemptive distribution $\delta$ is *auto-balanced* (that is $|\delta|_{\max}$-balanced) and this implies that $B_{\text{opt}} \leq D^{\text{opt}}$. We call an *optimally balanced* distribution a distribution which is $B_{\text{opt}}$-balanced. $B_{\text{opt}}$ can be found in polynomial time as it is shown in [1].

In the polynomial 2-approximation algorithm from [1], first an optimally balanced extremal distribution $\delta$ is constructed and then *consolidated.* An integral distribution $\delta$ is a *consolidation* of a distribution $\delta'$, if $\delta$ and $\delta'$ distribute the same job system $\mathcal{J}$ on the same multiprocessor $\mathcal{M}$ and $\delta(J,M) = \delta'(J,M)$ provided by integrality of $\delta(J,M)$. Consolidation is a special sort of rounding. To build a consolidation from a given distribution, for each preempted job in this distribution a single machine is determined and the job is completely placed (scheduled) on that machine. If $\delta'$ is a consolidation of a $B$-balanced distribution $\delta$, then $|\delta'|_{\max} \leq |\delta|_{\max} + B$. Hence, a consolidation of an optimally balanced distribution has the makespan $\leq 2B_{\text{opt}} \leq 2D^{\text{opt}}$.

An ingenious trick is developed in [1] to consolidate an $B$-balanced distribution obtained by linear programming. This consolidation is produced by a matching in the corresponding configuration graph and is $1-1$. That is different preempted jobs occupy in consolidation different machines. This trick is based on the analysis of structure of the consolidation graph of the above distribution. It is proved that this graph is a *pseudo-forest* i.e., a graph with its all components having the edges no more than the nodes.

Let us denote by $B^{\text{opt}}$ the minimal possible makespan of auto-balanced distributions. Auto-balanced distribution with such makespan call optimal auto-balanced. This distribution is extremal and hence acyclic. That is for such distributions the configuration graph is a forest. As easy to see $B_{\text{opt}} \leq B^{\text{opt}} \leq D^{\text{opt}}$. The same argument as presented in [1] show that $B^{\text{opt}}$ is calculable in polynomial time. This $B^{\text{opt}}$ represents a best known polynomially calculable lower estimation for $D^{\text{opt}}$.

The consolidation applied to the optimal auto-balanced distribution (which is not in general 1-1) gives a $2 - \frac{1}{m}$-approximation to optimum.

**Theorem 5.** *For every given configuration, a non-preemptive distribution with optimality ratio $2 - 1/m$ can be constructed in polynomial time.*

Given a configuration $\mathcal{C}$, construct by the linear programming a $\mathcal{C}$-optimal extremal distribution $\delta$. This distribution is acyclic

**Theorem 6.** *There is a polynomial algorithm which constructs a schedule $\sigma$ with makespan exceeding an optimal schedule by no more than $\frac{m-1}{m}p_{\max}$.*

## References

1. J.K. Lenstra, D. B. Shmoys, E. Tardos "Approximation algorithms for scheduling unrelated parallel machines" *Mathematical programming*, 46, 259-271, 1990
2. Karp, R. M. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, R.E. Miller and J.W. Thatcher, Eds., Plenum Press, New York,1972,pp.85-103
3. Graham R. L. "Bounds for certain multiprocessing anomalies" *Bell. Syst. Tech. J*, 45 (1966), 1563-1581
4. D. K. Friesen "Tighter bound for the MULTIFIT processor scheduling algorithm" *SIAM J. Comput.*, 13, n.1, Febr.1984,170-181.
5. Yue M. "On the exact upper bound for the multifit processors scheduling algorithm", *Ann. Oper. Res.*, 24 (1990), 233-259
6. D. K. Friesen, M. A. Langston "Bounds for MULTIFIT scheduling on uniform processors" *SIAM J. Comput.*, 12, n.1, Febr. 1983, 60-70.
7. D. S. Hochbaum and D. B. Shmoys "A polynomial approximation scheme for scheduling on uniform processors: using the dual approximation approach", SIAM J. Comput., 17, n. 3 (1988), 539-551

8. Horowitz E. and Sahni S., "Exact and approximate algorithms for scheduling nonidentical processors", J. ACM, 23,n.5, (apr. 1976),317-327
9. Ibarra O.H., Kim C. E. "Heuristic algorithms for scheduling independent tasks on nonidentical processors",J. ACM, 24,2 (April 1977) 280-289
10. E. Davis, J. M. Jaffe *Algorithms for Scheduling Tasks on Unrelated Processors*, Journal of the ACM 28, 721-736 (1981).
11. C. N. Potts "Analysis of a linear programming heuristic for scheduling unrelated parallel machines" *Discrete Appl. Math.*, 10, 155-164 (1985)
12. G. Dobson, *Scheduling independent tasks on uniform processors*, SIAM J. Comput., Vol 13,No 4, November 1984,pp. 705–716
13. Gonzalez T. and S. Sahni, "Open Shop Scheduling to Minimize Finish time", *Journal of the ACM* 23, 665-679 (1976).
14. Lawler E.L. and J.Labetoulle, "On preemptive scheduling of unrelated parallel processors by linear programming", *J. of the ACM* 25, 612-619 (1978).
15. G. B. Dantzig " Linear programming and extensions" Princeton University Press, Princeton, NJ, 1963.
16. Berge C. "Graphs and Hypergraphs", American Elsevier, New York, 1973, p. 134
17. Jansen K. and L.Porkolab. "Improved approximation schemes for scheduling unrelated parallel machines". *Proc. STOC99*, 1999.
18. Hall L.A. "Approximation algorithms for scheduling". In *Approximation algorithms for NP-hard problems*, PWS Pub., 1997.