Multiresource Malleable Task Scheduling

Hadas Shachnai *
The Department of Computer Science, Technion, Haifa 32000, Israel
and
John J. Turek
IBM T.J. Watson, P.O. Box 704, Yorktown Heights, NY 10598

Abstract

We present a technique for generalizing previously known results for single resource task systems to multiresource *malleable* task systems: Assuming a system with s resources, we apply a transformation which maps the system into a single resource task system. For a large class of heuristics it is shown, that given a heuristic H which achieves a suboptimality bound of c_H with respect to the optimal makespan/average response time in a single resource system, the proposed transformation yields an algorithm with a suboptimality bound of $c_H \cdot s$ with respect to the optimal makespan/average response time for the multiresource system.

Key words. multiresource systems, scheduling, malleable, response time, makespan

*Corresponding author. present address: IBM T.J. Watson, P.O. Box 704, Yorktown Heights, NY 10598. e-mail:hadas@watson.ibm.com

1 Introduction

Consider a task designed to run on a computer system having several different types of resources (e.g. processors in a multiprocessor system, blocks of shared memory). The amount of any one of the resources allotted to a task can be varied, with the running time of the task defined as a function of the quantity of each of the resource classes allotted to the task. We refer to such tasks as *malleable*. This paper examines the problem of generating a schedule for a set of n independent malleable tasks on a computer system having s different resource classes. Our goal is to find a non-preemptive schedule that minimizes the objective cost function. For the purposes of this paper, we consider two well known objective functions:

- The *makespan* or, equivalently, completion time of the last task.
- The *mean flow time* or, equivalently, average response time of the tasks.

For single resource tasks systems the problems defined by these objective functions have been well studied. For multiresource task systems these problems have received significantly less attention. In this paper we present a transformation that builds on the heuristics devised for single resource systems to generate a family of equivalent heuristics for multiresource systems.

Formally, a task system $\mathcal{T}$ consists of n tasks to be scheduled on a computer system containing several classes of resources $\{\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_s\}$. Associated with each task i is a function $t_i(\alpha_i^1, \alpha_i^2, \ldots, \alpha_i^s) > 0$ defining the *task execution time* as a function of the amount of each resource $\alpha_i^r \in \{1, ..., |\mathcal{R}_r|\}$ allotted to the task. We refer to $\mathcal{T}$ as a *multiresource malleable task system* (for short a *multiresource system*). In the special case where $s = 1$ we refer to a *unified malleable task system* (for short *unified system*) and denote it by $\mathcal{U}$.
The tasks are assumed to be non-preemptive, so that the amount α_i^r of each resource allotted to task i remains assigned to i for the duration of its execution. The system will start task i at some *starting time*, say τ_i. It will then complete task i at some later *completion time* $\tau_i + t_i(\alpha_i^1, \alpha_i^2, \ldots, \alpha_i^s)$. A *schedule* will consist, for each task, of a resource allotment $\{\alpha_i^1, \alpha_i^2, \ldots, \alpha_i^s\}$ and a starting time τ_i. A schedule is required to be *legal* in the sense that, for any time t, the amount of any resource used does not exceed the total amount of that resource

$$\forall r \sum_{\{i|\tau_i \le t < \tau_i + t_i(\alpha_i^1, \alpha_i^2, \ldots \alpha_i^s)\}} \alpha_i^r \le |\mathcal{R}_r|.$$

We are interested in minimizing the cost of the schedule. Our first objective function is the makespan given by

$$\mathcal{M}_\mathcal{T} = \max_{1 \le i \le n} \{\tau_i + t_i(\alpha_i^1, \ldots, \alpha_i^s)\}. \quad (1)$$

An important related measure to the makespan is the *throughput* of the system typically characterized by the on-line scheduling problem. By applying a slight variation of the techniques presented in [8] to the results presented in this paper we also derive algorithms for the on-line scheduling problem. The second objective function we study in this paper is the *flow time* [2] given by

$$\mathcal{F}_\mathcal{T} = \sum_{i=1}^{n} \{\tau_i + t_i(\alpha_i^1, \ldots, \alpha_i^s)\}. \quad (2)$$

Minimizing the flow time is equivalent to minimizing the average response time of the tasks in the system.

Both the minimum makespan and the minimum flow time problems are known to be $\mathcal{NP}$-complete [3, 5], even for the single resource case; consequently, the problems we study in this paper are also $\mathcal{NP}$-complete. Therefore we focus on heuristics having provable suboptimality bounds and polynomial running times. In particular, we give a transformation that maps any multiresource system $\mathcal{T}$ into a unified system $\mathcal{U}$. We show, for a large class of heuristics, that applying a heuristic H with a suboptimality bound of c_H for the unified system $\mathcal{U}$ yields a suboptimality bound of $c_H \cdot s$ for $\mathcal{T}$. As a direct result of our transformation we get

- An algorithm for the malleable makespan problem having a suboptimality bound of $2s$ and a running time of $O(n \sum_{r=1}^{s} |\mathcal{R}_r|)$, where n is the number of tasks, and $|\mathcal{R}_r|$ is the number of units of the rth resource. This bound holds also for the case where the resources allocated to a task need to be allocated from contiguous blocks.
- An algorithm for the on-line scheduling problem having a suboptimality bound of $4s$.
- An algorithm for the malleable average response time problem having a suboptimality bound of $8s$ and a running time of $O(n^3 + n^2 \sum_{r=1}^{s} |\mathcal{R}_r|)$. For the special case where the task execution times have *sublinear* speedups this bound reduces to $2s$.

Earlier works refer only to the makespan problem in *non-malleable* multiresource systems (i.e. where each task has a *fixed* requirement for resources): Garey and Graham [3] considered this problem and showed that for the case where the allocation of resources may be non-contiguous the List scheduling algorithm achieves a suboptimality bound of $s + 1$. Srivastav and Stangier studied in [7] the special case where all tasks have *unit* running times and showed that for any $\varepsilon > 0$ the optimal makespan can be approximated within a factor of $1 + \varepsilon$, provided that the amount from each resource is $\Omega(\frac{1}{\varepsilon^2} \lg(Cs))$, with C the size of the minimal schedule.
In generalizing the above results, we derive $O(s)$-approximation bounds for both the makespan and the flow time problems in malleable multiresource systems. The bounds obtained for the makespan problem are shown to hold also for the case where release times are *unknown*, and where resource allocation units are required to be *contiguous*.
In Section 2 we present some preliminary results used to prove the applicability of our transformation to the makespan and flow time problems. Section 3 gives an outline of the transformation in terms of a general objective function. In Sections 4 and 5 we show how to apply our transformation to minimizing the makespan and the flow time. We conclude in Section 6 with discussion of areas for further study.

2 Preliminaries

Let $|\mathcal{R}_r|$ denote the number of units of resource $\mathcal{R}_r$ in the system, $\beta_i^r = \frac{\alpha_i}{|\mathcal{R}_r|}$ is defined to be the fraction of the resource capacity allotted to task i. Also, let $\vec{\beta}_i$ denote the allocation $\{\beta_i^1, \beta_i^2, \ldots \beta_i^s\}$ of resources to task i. In a multiresource system we refer to the running time of task i in terms of $\vec{\beta}_i$; i.e., $t_i(\vec{\beta}_i)$. W.o.l.g, we assume that t_i is a nonincreasing function; that is, increasing the amount of any resource without a corresponding decrease in the amount of another resource will not result in an increase in the task's execution time.

Lemma 1: *Let $\mathcal{S}$ be any schedule for a multiresource system $\mathcal{T}$, with a given resource allotment $\vec{\beta}$, then, the latest task completion $\mathcal{M}_{\mathcal{T}}(\vec{\beta})$ in $\mathcal{S}$ satisfies*

$$\mathcal{M}_{\mathcal{T}}(\vec{\beta}) \geq \max\{\mathcal{W}_{\mathcal{T}}(\vec{\beta}),\ \mathcal{H}_{\mathcal{T}}^{\max}(\vec{\beta})\}\ , \tag{3}$$

where

$$\mathcal{H}_{\mathcal{T}}^{\max}(\vec{\beta}) = \max_{1\leq i\leq n} t_i(\vec{\beta}_i) \tag{4}$$

is the maximum height bound, *and*

$$\mathcal{W}_{\mathcal{T}}(\vec{\beta}) = \frac{1}{s}\sum_{i=1}^{n} t_i(\vec{\beta}_i)[\sum_{r=1}^{s}\beta_i^r] \tag{5}$$

is the work bound *of $\mathcal{T}$ for the allotment $\vec{\beta}$.*

Proof: By definition, $\mathcal{H}_{\mathcal{T}}^{\max}(\vec{\beta})$ denotes the execution of the longest task in $\mathcal{T}$. Thus, $\mathcal{H}_{\mathcal{T}}^{\max}(\vec{\beta})$ is a lower bound on the cost of any schedule.

For the $\mathcal{W}_{\mathcal{T}}(\vec{\beta})$ component, consider relaxing the resource constraints on all the resources but one. In particular, define

$$\mathcal{W}^r(\vec{\beta}) = \sum_{i=1}^{n} t_i(\vec{\beta}_i)\beta_i^r$$

to be the work bound given resource r. Observing that $\mathcal{W}^r(\vec{\beta})$ denotes a lower bound on the length of schedule $\mathcal{S}$, i.e $\mathcal{M}_{\mathcal{T}}(\vec{\beta}) \geq \mathcal{W}^r(\vec{\beta})$, we have

$$\mathcal{M}_{\mathcal{T}}(\vec{\beta}) \geq \frac{1}{s}\sum_{r=1}^{s}\mathcal{W}^r(\vec{\beta})\sum_{i=1}^{s}\sum_{i=1}^{n} t_i(\vec{\beta}_i)\beta_i^r = \frac{1}{s}\sum_{i=1}^{n} t_i(\vec{\beta}_i)[\sum_{r=1}^{s}\beta_i^r] = \mathcal{W}_{\mathcal{T}}(\vec{\beta}), \tag{6}$$

completing the proof of the lemma. ■

We now define the components that will be used to define the lower bounds for the flow time problem.

Definition 1: *For a given allocation vector $\vec{\beta}$, denote by $\sigma_{\vec{\beta}}$ the arrangement of the tasks in nondecreasing order by the areas, defined as $t_i(\vec{\beta}_i)\sum_{r=1}^{s}\beta_i^r$ for $1 \leq i \leq n$, then $t_i(\vec{\beta}_i, \sigma_{\vec{\beta}})$ is the execution time of the ith task in $\sigma_{\vec{\beta}}$. Let $\mathcal{T}$ be a task system consisting of n tasks. The* squashed area bound *of $\mathcal{T}$ for the allotment $\vec{\beta}$ is defined to be*

$$\mathcal{A}_{\mathcal{T}}(\vec{\beta}) = \frac{1}{s}\sum_{i=1}^{n} t_i(\vec{\beta}_i, \sigma_{\vec{\beta}})(n-i+1)[\sum_{r=1}^{s}\beta_i^r]. \tag{7}$$

Lemma 2: *Let $\mathcal{S}$ be any schedule for a multiresource system $\mathcal{T}$. Then for a given allocation $\vec{\beta}$, the flow time of the schedule $\mathcal{S}$ satisfies*

$$\mathcal{F}_{\mathcal{T}}(\vec{\beta}) \geq \mathcal{A}_{\mathcal{T}}(\vec{\beta}) + \frac{1}{2}(\mathcal{H}_{\mathcal{T}}^{\text{tot}}(\vec{\beta}) - \mathcal{W}_{\mathcal{T}}(\vec{\beta}))\ , \tag{8}$$

where

$$\mathcal{H}_{\mathcal{T}}^{\text{tot}}(\vec{\beta}) = \sum_{j=1}^{n} t_i(\vec{\beta}_i) \tag{9}$$

is the total height bound *of $\mathcal{T}$ for the allotment $\vec{\beta}$, and $\mathcal{W}_{\mathcal{T}}(\vec{\beta})$, $\mathcal{A}_{\mathcal{T}}(\vec{\beta})$ are defined in (5) and (7).*

The proof of the lemma is similar to the proof of Theorem 1 in [4] and is thus omitted. ■

3 The Transformation to a Unified Task System

We now describe an allocation algorithm for multiresource malleable system, based on a transformation to a unified system. Our transformation is formulated in terms of a general objective function.

Let $\mathcal{T}$ denote a multiresource system with s resources and the set of tasks $T_1, \ldots, T_n$. Then $\mathcal{T}$ is defined by $t_i(\vec{\beta_i}), \forall \vec{\beta_i} \in [0,1]^s \;\; 1 \leq i \leq n$.
For the rth resource with $|\mathcal{R}_r|$ allocation units, let $\vec{x^r} = (x_1^r, \ldots, x_{|R_r|}^r)$ be the vector of possible allocations of R_r, given as the possible values of β_i^r for some task $1 \leq i \leq n$. Thus,

$$x_k^r = \frac{k}{|\mathcal{R}_r|} \;\; , \;\; 0 \leq x_k^r \leq 1 \qquad \forall \;\; 1 \leq k \leq |\mathcal{R}_r| \;\; .$$

For a given allocation vector $\vec{\beta} = (\vec{\beta_1}, \ldots, \vec{\beta_n})$ we define the *maximal allocation vector* for $\vec{\beta}$ as $\vec{\Gamma} = (\vec{\Gamma_1}, \ldots, \vec{\Gamma_n})$, such that $\vec{\Gamma_i} = (\Gamma_i^1, \ldots, \Gamma_i^s)$, and

$$\Gamma_i^r = \{ \max_{1 \leq k \leq |\mathcal{R}_r|} x_k^r \mid x_k^r \leq \Gamma_i \} \;\; , \tag{10}$$

with

$$\Gamma_i = \max_{1 \leq r \leq s} \beta_i^r \quad \forall 1 \leq r \leq s \;\; . \tag{11}$$

Hence, when transferring from $\vec{\beta} = (\vec{\beta_1}, \ldots, \vec{\beta_n})$ to $\vec{\Gamma} = (\vec{\Gamma_1}, \ldots, \vec{\Gamma_n})$ in $\mathcal{T}$, the ith task is allocated the amount Γ_i^r from the resource R_r. This is the fraction of R_r, which is closest to Γ_i. Denote by f a mapping from $\mathcal{T}$ to the unified system $\mathcal{U}$, $f : [0,1]^s \longrightarrow [0,1]$, such that $f(\vec{\beta_i}) = \Gamma_i$, where Γ_i is defined in (11), and the running time of the ith task in $\mathcal{U}$ is given by $t_i'(\Gamma_i) = t_i(\vec{\Gamma_i})$.
The following algorithm uses the above transformation to a unified system $\mathcal{U}$, for finding efficient allocation of resources to the tasks in $\mathcal{T}$.

Algorithm $\mathbf{A}_{\mathcal{MR}}$ (*Multiple Resource* allocation in a malleable task system):

1. Let $\vec{x^r} = (x_1^r, \ldots, x_{|R_r|}^r)$ be the vector of possible allocations of R_r, $0 \leq x_k^r \leq 1 \;\; \forall 1 \leq k \leq |\mathcal{R}_r|$.
2. Merge the vectors $\vec{x^1}, \ldots, \vec{x^s}$ into the ordered allocation vector $\vec{x}$.
3. Transfer to a unified system $\mathcal{U}$, in which the possible allocations of the single resource R are given by $\vec{x}$. For any allocation Γ_i of $\mathcal{R}$ to T_i, let $t_i'(\Gamma_i) = t_i(\vec{\Gamma_i})$ be the execution time of T_i in $\mathcal{U}$, where $\vec{\Gamma_i} = (\Gamma_i^1, \ldots, \Gamma_i^s)$ and Γ_i^r is defined in (10).
4. For a minimization criterion given by the function $f_{\mathcal{U}} : [0,1]^n \longrightarrow \mathbf{R}$, find in $\mathcal{U}$ the allocation vector $\vec{\Gamma'} = (\Gamma_1', \ldots, \Gamma_n')$ such that
$$f_{\mathcal{U}}(\vec{\Gamma'}) = \min_{\vec{\Gamma}} f_{\mathcal{U}}(\vec{\Gamma}) \;\; , \tag{12}$$
using a *Resource to Task Allotment Algorithm* $\mathcal{A}$ for a unified task system.
5. For the chosen allocation $\vec{\Gamma_i}'$ for T_i in $\mathcal{U}$, use the corresponding allocation vector $\vec{\Gamma_i}'$ in $\mathcal{T}$.

4 Scheduling to Minimize the Makespan

In applying the algorithm $A_{\mathcal{MR}}$ to the makespan problem we compute for each allocation vector $\vec{\Gamma}$ in the unified system $\mathcal{U}$ the function

$$f_{\mathcal{U}}(\vec{\Gamma}) = \max\left(\mathcal{W}_{\mathcal{U}}(\vec{\Gamma}), \mathcal{H}_{\mathcal{U}}^{\max}(\vec{\Gamma})\right) \ . \tag{13}$$

Theorem 1: *Let H be a scheduling heuristic such that for the allocation vector $\vec{\Gamma'}$ chosen by $A_{\mathcal{MR}}$, the suboptimality bound c_H on the schedule produced by H on $\mathcal{U}$ can be formulated as a function of the form*

$$\frac{\mathcal{M}_{\mathcal{U}}(\vec{\Gamma'})}{\max(\mathcal{W}_{\mathcal{U}}(\vec{\Gamma^*}), \mathcal{H}_{\mathcal{U}}^{\max}(\vec{\Gamma^*}))} \leq c_H \ , \tag{14}$$

where $\vec{\Gamma^*}$ is an optimal allocation vector in $\mathcal{U}$. Then, applying H to $\mathcal{U}$ will yield a schedule that is feasible in $\mathcal{T}$ with a suboptimality bound on the makespan of $c_H \cdot s$.

Proof: For the allocation vector $\vec{\Gamma'}$ chosen in $\mathcal{U}$ by the algorithm $A_{\mathcal{MR}}$ and the corresponding vector $\vec{\Gamma'}$ in $\mathcal{T}$, we need to show that if

$$\mathcal{M}_{\mathcal{U}}(\vec{\Gamma'}) \leq c_H \max(\mathcal{W}_{\mathcal{U}}(\vec{\Gamma'}), \mathcal{H}_{\mathcal{U}}^{\max}(\vec{\Gamma'})) \tag{15}$$

then

$$\mathcal{M}_{\mathcal{T}}(\vec{\Gamma'}) \leq c_H s \max(\mathcal{W}_{\mathcal{T}}(\vec{\beta^*}), \mathcal{H}_{\mathcal{T}}^{\max}(\vec{\beta^*})) \ , \tag{16}$$

where $\vec{\beta^*}$ is an optimal allocation vector in $\mathcal{T}$.
Let $\vec{\Gamma^*}$ denote an allocation vector in $\mathcal{U}$ corresponding to an optimal allocation vector $\vec{\beta^*}$ in $\mathcal{T}$. By the mapping from $\mathcal{T}$ to $\mathcal{U}$ $t'(\Gamma_i) = t_i(\vec{\Gamma_i})$, hence

$$\begin{aligned} \mathcal{M}_{\mathcal{T}}(\vec{\Gamma'}) \leq \mathcal{M}_{\mathcal{U}}(\vec{\Gamma'}) \quad &\leq c_H \max(\mathcal{W}_{\mathcal{U}}(\vec{\Gamma'}), \mathcal{H}_{\mathcal{U}}^{\max}(\vec{\Gamma'})) \\ &\leq c_H \max(\mathcal{W}_{\mathcal{U}}(\vec{\Gamma^*}), \mathcal{H}_{\mathcal{U}}^{\max}(\vec{\Gamma^*})) \ . \end{aligned}$$

Now, if $\mathcal{H}_{\mathcal{U}}^{\max}(\vec{\Gamma^*}) > \mathcal{W}_{\mathcal{U}}(\vec{\Gamma^*})$, then inequality (16) holds, since $\mathcal{H}_{\mathcal{U}}^{\max}(\vec{\Gamma^*}) \leq \mathcal{H}_{\mathcal{U}}^{\max}(\vec{\beta^*})$.
For the case where $\mathcal{H}_{\mathcal{U}}^{\max}(\vec{\Gamma^*}) < \mathcal{W}_{\mathcal{U}}(\vec{\Gamma^*})$ we have

$$t'_i(\Gamma_i^*)\Gamma_i^* \leq t_i(\vec{\beta^*}) \sum_{r=1}^{s} \beta_i^r \ ,$$

therefore $\mathcal{W}_{\mathcal{U}}(\vec{\Gamma^*}) \leq s\mathcal{W}_{\mathcal{T}}(\vec{\beta^*})$. ■
The performance bounds of many heuristics for solving the makespan problem are formulated in terms of the suboptimality bound defined in Theorem 1 (see, for example, [1],[12]).

Corollary 1: Using the algorithm $A_{\mathcal{MR}}$ with $f_{\mathcal{U}}$ as given in (13), there exists a schedule for the tasks achieving a $2s$-approximation to the optimal makespan. The overall complexity is $O(n \sum_r |\mathcal{R}_r|)$.

Proof: Let m be the least common dividend of $(|\mathcal{R}_1|, \ldots, |\mathcal{R}_s|)$. We apply $\mathcal{A}$ on a system $\mathcal{U}$ with a single resource having $|R| = m$ units, such that the possible allocations of R are determined by the fractions available in the vector $\vec{x}$. Hence, there are at most $l = \sum_r |\mathcal{R}_r|$ possible allocations of R. This defines a thinning $\{i_1, \ldots, i_l\}$ on the set $\{1, \ldots, m\}$. Applying the allocation algorithm in [12]

to the set $\{i_1, \ldots, i_l\}$ for choosing an optimal allocation with respect to (12) and the single resource $\mathcal{R}$ in $\mathcal{U}$ requires $O(n|\mathcal{R}|)$ steps. Scheduling the tasks with the allocation chosen by the algorithm $A_{\mathcal{MR}}$ by Riverse-Fit [6] yields a $2s$-approximation to the optimal makespan. ■
We note that the Riverse-Fit schedule guarantees contiguous allocation of resources to the tasks, and hence, the bound holds with the additional requirement of contiguity.

Using Theorem 2.1 in [8], we obtain

Corollary 2: The implementation of the algorithm $A_{\mathcal{MR}}$ for the case where the release dates of the tasks are unknown yields a $4s$-approximation to the optimal makespan. The overall complexity is $O(n \sum_r |\mathcal{R}_r|)$.

5 Minimizing the Average Response Time

We apply the algorithm $A_{\mathcal{MR}}$ to the average response time problem by using

$$f_{\mathcal{U}}(\vec{\Gamma}) = \mathcal{A}_{\mathcal{U}}(\vec{\Gamma}) + \frac{1}{2}(\mathcal{H}_{\mathcal{U}}^{\mathsf{tot}}(\vec{\Gamma}) - \mathcal{W}_{\mathcal{U}}(\vec{\Gamma})) \ , \tag{17}$$

where $\mathcal{A}_{\mathcal{U}}(\cdot), \mathcal{H}_{\mathcal{U}}^{\mathsf{tot}}(\cdot)$ and $\mathcal{W}_{\mathcal{U}}(\cdot)$ correspond to $\mathcal{A}_{\mathcal{T}}(\cdot), \mathcal{H}_{\mathcal{T}}^{\mathsf{tot}}(\cdot)$ and $\mathcal{W}_{\mathcal{T}}(\cdot)$ respectively, for the special case where $s = 1$.

Theorem 2: *Let H be a scheduling heuristic such that for the allocation vector $\vec{\Gamma'}$ chosen by $A_{\mathcal{MR}}$, the suboptimality bound c_H on the schedule produced by H for $\mathcal{U}$ can be formulated as a function of the form*

$$\frac{\mathcal{F}_{\mathcal{U}}(\vec{\Gamma'})}{\mathcal{A}_{\mathcal{U}}(\vec{\Gamma^*}) + \frac{1}{2}(\mathcal{H}_{\mathcal{U}}^{\mathsf{tot}}(\vec{\Gamma^*}) - \mathcal{W}_{\mathcal{U}}(\vec{\Gamma^*}))} \leq c_H \ , \tag{18}$$

where $\vec{\Gamma^*}$ is an optimal allocation vector in $\mathcal{U}$. Then, applying H to $\mathcal{U}$ will yield a schedule that is feasible in $\mathcal{T}$ with a suboptimality bound on the flow time of $c_H \cdot s$.

Proof: For the allocation vector $\vec{\Gamma'}$ chosen in $\mathcal{U}$ and the corresponding vector $\vec{\Gamma'}$ in $\mathcal{T}$, we need to show that if

$$\mathcal{F}_{\mathcal{U}}(\vec{\Gamma'}) \leq c_H[\mathcal{A}_{\mathcal{U}}(\vec{\Gamma'}) + \frac{1}{2}(\mathcal{H}_{\mathcal{U}}^{\mathsf{tot}}(\vec{\Gamma'}) - \mathcal{W}_{\mathcal{U}}(\vec{\Gamma'}))] \ , \tag{19}$$

then

$$\mathcal{F}_{\mathcal{T}}(\vec{\Gamma'}) \leq c_H \ s \ [\mathcal{A}_{\mathcal{T}}(\vec{\beta^*}) + \frac{1}{2}(\mathcal{H}_{\mathcal{T}}^{\mathsf{tot}}(\vec{\beta^*}) - \mathcal{W}_{\mathcal{T}}(\vec{\beta^*}))] \ , \tag{20}$$

where $\vec{\beta^*}$ is an optimal allocation in $\mathcal{T}$.
Let $\vec{\Gamma^*}$ be the allocation vector in $\mathcal{U}$ corresponding to an optimal allocation vector $\vec{\beta^*}$ in $\mathcal{T}$, then

$$\mathcal{F}_{\mathcal{T}}(\vec{\Gamma'}) \leq c_H[\mathcal{A}_{\mathcal{U}}(\vec{\Gamma^*}) + \frac{1}{2}(\mathcal{H}_{\mathcal{U}}^{\mathsf{tot}}(\vec{\Gamma^*}) - \mathcal{W}_{\mathcal{U}}(\vec{\Gamma^*}))] \ .$$

Using the assumption that an increased allocation of resources does not increase the running time of a task, we have

$$\mathcal{H}_{\mathcal{U}}^{\mathsf{tot}}(\vec{\Gamma^*}) \leq \mathcal{H}_{\mathcal{T}}^{\mathsf{tot}}(\vec{\beta^*}) \ .$$

Also,

$$\mathcal{A}_{\mathcal{U}}(\vec{\Gamma^*}) - \frac{1}{2}\mathcal{W}_{\mathcal{U}}(\vec{\Gamma^*}) \leq \sum_{i=1}^{n}(n-i+\frac{1}{2})t_i'(\Gamma_i^*, \sigma_{\vec{\beta^*}})\Gamma_i^* \leq s\,[\mathcal{A}_{\mathcal{T}}(\vec{\beta^*}) - \frac{1}{2}\mathcal{W}_{\mathcal{T}}(\vec{\beta^*})]\ , \tag{21}$$

which completes the proof. ■

Heuristics for solving the flow time problem can often be formulated in terms of the suboptimality bound defined in Theorem 2 (e.g. [4, 5, 10]). Theorem 2 states that for this class of heuristics our claimed suboptimality bound will hold. In particular, a $8s$-approximation can be obtained by applying the allocation algorithm in [12] for minimizing $f_{\mathcal{U}}$ as in (17), with the schedule determined by the SMART algorithm presented in [9]. Hence, we have

Corollary 3: Using the algorithm $A_{\mathcal{MR}}$, there exists a schedule for the tasks achieving a $8s$-approximation to the minimal average response time with the complexity $O(n^3 + n^2 \sum_{r=1}^{s} |\mathcal{R}_r|)$.

6 Discussion

In this paper we presented a technique that allows algorithms for malleable task scheduling in a single resource task system to be extended to a multiresource task system having s resources. For a large set of previously studied algorithms our approach adds, at most, a multiplicative factor of s to the original bounds. Our results apply to the two main measures commonly used in evaluating the performance of scheduling heuristics: The makespan and the mean flow time of the schedule. Several areas remain open for future research:

- Garey and Graham [3] give a suboptimality bound of $s+1$ for the makespan problem as applied to non-malleable tasks in a multiresource system. Using our technique we derive a bound of $2s$ for a malleable task system. While we conjecture that a direct transformation of the Garey and Graham result to a malleable task system would yield a bound of $s+1$ we currently have no proof of that bound.

- Our algorithm for the on-line makespan problem builds on the result by Shmoys, Wein and Williamson [8]. This extra step incurs an additional factor of 2. We expect a more direct transformation to yield a bound that is better than $4s$.

- Finally, Srivastav and Stangier showed in [7] that a constant approximation exists for a restricted version of the multiresource non-malleable makespan problem, where all tasks have unit running times, and a lower bound is set on the amount of each of the resources. The question whether a constant bound exists for the malleable case remains open. Alternatively, it would be of interest to show the existence of a function g, such that any heuristic yields an $\Omega(g(s))$-approximation to the optimal makespan. Similarly for the multiresource flow time problem which was not studied previously.

References

[1] Belkhale, K. and P. Banerjee. Approximate Scheduling Algorithms for the Partitionable Independent Task Scheduling Problem. In *Proceedings of the 1990 International Conference of Parallel Processing*, volume I, pages 72–75, August 1990.

[2] Coffman, E., Editor. *Computer and Job-Shop Scheduling Theory.* Wiley, New York, 1976.

[3] Garey, M. and R. Graham. Bounds for Multiprocessor Scheduling with Resource Constraints. *SIAM Journal on Computing*, 4(2):187–200, June 1975.

[4] Ludwig, W. and P. Tiwari. The Power of Choice in Scheduling Parallel Tasks. Technical Report 1190, University of Wisconsin, Madison, Computer Science Department, November 1993.

[5] Shachnai, H. and J. Glasgow. Minimizing the Flow Time for Parallelizable Task Systems. Technical Report TR790, Technion IIT, November 1993.

[6] Schiermayer I. Reverse-Fit: A 2-Optimal Algorithm for Packing Rectangles, Proceedings of the Second European Symposium on Algorithms (ESA'94), LNCS 855, Springer-Verlag, pages 290–299, September 1994.

[7] Srivastav, A. and P. Stangier. Tight Approximation for Resource Constrained Scheduling Problems Proceedings of the Second European Symposium on Algorithms (ESA'94), LNCS 855, Springer-Verlag, pages 307–318, September 1994.

[8] Shmoys, D., J. Wein, and D. Williamson. Scheduling Parallel Machines On-Line. In *Proceedings of the* 32^{nd} *Annual Symposium on Foundations of Computer Science*, pages 131–140, October 1991.

[9] Turek, J., U. Schwiegelshohn, J. Wolf, and P. Yu. Smart SMART Bounds for Weighted Response Time Scheduling. Manuscript, 1994.

[10] Turek, J., U. Schwiegelshohn, J. Wolf, and P. Yu. Scheduling Parallel Tasks to Minimize Average Response Times. In *Proceedings of the SIAM Symposium on Discrete Algorithms*, Arlington, VA, pages 112–121, January 1994.

[11] Turek, J., J. Wolf, K. Pattipati, and P. Yu. Scheduling Parallelizable Tasks: Putting it all on the Shelf. In *Proceedings of the ACM Sigmetrics Conference, Newport, RI*, pages 225–236, June 1992.

[12] Turek, J., J. Wolf, and P. Yu. Approximate Algorithms for Scheduling Parallelizable Tasks. In *Proceedings of the 4th Annual Symposium on Parallel Algorithms and Architectures, San Diego, CA*, pages 323–332, June 1992.