BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 62, No. 2, 2014
DOI: 10.2478/bpasts-2014-0025

Dedicated to Professor J. Klamka
on the occasion of his 70th birthday

Scheduling arbitrary number of malleable tasks on multiprocessor systems

M.S. BARKETAU[1], M.Y. KOVALYOV[1], J. WĘGLARZ[2], and M. MACHOWIAK[2*]

[1] United Institute of Informatics Problems, National Academy of Sciences of Belarus, Surganova 6, 220012 Minsk, Belarus
[2] Institute of Computing Science, Poznan University of Technology, 3a Piotrowo St., 90-965 Poznań, Poland

Abstract. The problem of scheduling n tasks in a multiprocessor system with m processors to minimize the makespan is studied. Tasks are malleable, which means that a task can be executed by several processors at a time, its processing speed depends on the number of allocated processors, and a set of processors allocated to the same task can change over time. The processing speed of a task is a strictly increasing function of the number of processors allocated to this task. The earlier studies considered the case $n \leq m$. This paper presents results for arbitrary n and m including characterizations of a feasible domain and an optimal solution, polynomial time algorithms for strictly increasing convex and concave processing speed functions, and a combinatorial exponential algorithm for arbitrary strictly increasing functions.

Key words: multiprocessor scheduling, malleable tasks, makespan.

1. Problem formulation and literature review

A computing task is called *malleable* if it can be processed on several processors at the same time, its processing speed depends on the number of allocated processors, and the set of processors allocated to the same task can change over time. The following problem is studied.

There are n malleable tasks to be scheduled for processing on m identical parallel processors. Each task j is associated with its amount of work, p_j. Let $f_j(r)$ denote the processing speed of task j if it is allocated r processors. For the sake of the simplicity we denote with $f_j(r)$ functions instead of its value at r.

All $f_j(r)$ are assumed to be strictly increasing continuous integrable functions with $f_j(0) = 0$. A schedule specifies an allocation of processors to the tasks over time. We limit ourselves to schedules in which there is a finite number of time intervals where the same number of processors is allocated to the same task for all tasks in the same interval. The problem is to find a schedule with the minimum makespan. A schedule can be completely characterized by the number of time intervals, L, the interval lengths, $\Delta_1, \ldots, \Delta_L$, where the corresponding intervals are $[0, \Delta_1]$, $[\Delta_1, \Delta_1 + \Delta_2], \ldots$, and the number of processors $r_j^{(l)}$, $r_j^{(l)} \in \{0, 1, \ldots, m\}$, allocated to each task j in each interval l, $j = 1, \ldots, n$, $l = 1, \ldots, L$. The makespan of the corresponding schedule is $C_{\max} = \sum_{l=1}^{L} \Delta_l$. A schedule is called feasible if $\sum_{i=1}^{n} r_i^{(l)} \leq m$, $l = 1, \ldots, L$, and $\int_0^{C_{\max}} f_j(r_j(t))dt = \sum_{l=1}^{L} \Delta_l f_j(r_j^{(l)}) = p_j$, $j = 1, \ldots, n$, where $r_j(t)$ denote the number of processors allocated to task j at time moment t. We denote this problem as P, and the relaxed problem in which the numbers of allocated processors are not required to be integers as P_{cntn}, where $cntn$ is the abbreviation for *continuous*. For the latter problem, processors can be viewed as a continuously divisible renewable resource, whose amount is upper bounded by m at each time moment.

Motivation for problem P comes from the management of large scale multiprocessor systems intended for large scale parallel computations. This motivation is comprehensively described in Błażewicz et al. [1], Bernard et al. [2] and Dongarra et al. [3]. Problem P_{cntn} can be used as an approximate model for the problem of scheduling *Multiple Instruction stream, Multiple Data stream (MIMD)* processors.

The following results are available for the *moldable* task scheduling problem which differs from problem P in that the set of processors allocated to a task cannot change. Du and Leung [4] proved this problem to be NP-hard. However, some special cases are polynomially solvable, see Bianco et al. [5] and Błażewicz et al. [6]. Turek et al. [7] showed that any λ-approximation algorithm for the two dimensional bin-packing problem can be polynomially transformed into a λ-approximation algorithm for the moldable task scheduling problem. Based on this result, Ludwig [8] developed a 2-approximation algorithm. Rapine et al. [9] developed a two phase approximation algorithm with worst-case performance guarantee $\sqrt{3}$ and later proposed a 3/2-approximation algorithm, see [10]. Prasanna and Musicus [11] considered this problem with precedence constraints associated with the task set. For the case of the same processing speed functions r^α, $0 < \alpha < 1$, a closed form solution for a series-parallel precedence graph was derived. In [12] an approximation algorithm with very good average behaviour has been proposed. Surveys of the complexity and algorithms for moldable task scheduling problems are given by Drozdowski [13] and Błażewicz et al. [14].

In Sec. 2 we prove properties of the feasible domain and an optimal solution of the problem P_{cntn} with arbitrary piece-

*e-mail: maciej.machowiak@cs.put.poznan.pl

wise linear strictly increasing continuous functions $f_j(r)$. The same properties were proved by Węglarz [15, 16] for the case $n \geq m$. They were used in all succeeding publications on malleable task scheduling, and the authors often implicitly assumed that these properties hold for the general case of no relation between n and m. However, to the best of our knowledge, no formal proof of these properties existed so far for the general case.

Section 3 justifies application of the well known *gang strategy* to the case of convex functions $f_j(r)$. According to this strategy, each task is allocated all m processors until it is finished. No formal proof of optimality of this strategy was given for the case $n < m$ before.

Section 4 studies the case of concave functions $f_j(r)$. It describes an algorithm for problems P_{cntn} and P. Its time complexity is $O(n(\log C^+ + \log 1/\varepsilon))$ (without taking into account complexity of computing the revers function), where C^+ is an upper bound on the optimal makespan value and $\varepsilon > 0$ is an upper bound on the absolute deviation from the optimal value. Recently, Sanders and Speck [17] have proposed the $O(n + \min\{n, m\} \log m)$ time algorithm for the same problem. Both algorithms are based on the same bisection search approach and were developed independently. However, the algorithm of Sanders and Speck enumerates the number of processors rather than the makespan values. Section 4 also provides an analytical solution to the problem with functions $f_j(r) = r^\alpha$, $j = 1, \ldots, n$, where $0 < \alpha < 1$, which were introduced by Prasanna and Musicus [11].

Section 5 presents a combinatorial exponential algorithm for problem P in the case of general functions $f_j(r)$. Jansen and Porkolab [18] developed an optimal algorithm for this problem, which is polynomial in n and m. This algorithm generalizes their approach for a problem, in which the number of processors assigned to each task is given. Note that this algorithm is pseudo-polynomial because its run time estimation includes m, and the problem input length is $O(n)$. It is based on a linear programming characterization of the problem with a strong separation oracle to be handled by the ellipsoid method. Due to the sophistical structure of the algorithm, we are unable to establish a run time estimation of it. The authors do not give this estimation either. We think that our exponential algorithm can be competitive with their algorithm for some classes of instances, as it is in the case of exponential and polynomial algorithms for the linear programming problem. However, we are unable to provide a computer experiment with the algorithm of Jansen and Porkolab, which was also beyond their power. Jansen [19] developed an asymptotic fully polynomial time approximation scheme for problem P.

Section 6 summarizes the results and suggests directions for future research.

2. Characterization of feasible domain and optimal schedule for problem P_{cntn}

We first prove the following theorem.

Theorem 1. Let V be a closed convex set in the n-dimensional space of real numbers, C be a real number and $u(t) = (u_1(t), u_2(t), \ldots, u_n(t))$ be a vector of integrable functions such that $u(t) \in V$ for each $t \in [0, C]$. Then the point

$$u^{(C)} := \left(\int_0^C u_1(t)dt/C, \int_0^C u_2(t)dt/C, \ldots, \int_0^C u_n(t)dt/C \right)$$

belongs to the set V.

Proof. By the definition of the definite (Riemann) integral,

$$\int_0^C u_j(t)dt = \lim_{\delta_{\max} \to 0} \sum_{i=1}^{k} u_j(t_i)\delta_i,$$

where $0 = x_0 \leq t_1 \leq x_1 \leq \cdots \leq x_{k-1} \leq t_k \leq x_k = C$ is a finite sequence, $\delta_i = x_i - x_{i-1}$, and $\delta_{\max} = \max_{1 \leq i \leq k}\{\delta_i\}$. Thus, $u^{(C)}$ is a limit of a sequence of points which are convex combinations with coefficients $\alpha_i = \delta_i/C$ of points $u(t_i)$, $i = 1, \ldots, k$, each of which belongs to V. Since set V is convex, a convex combination of its points belongs to V. Furthermore, since V is closed, the limit of the above mentioned sequence of its points, $u^{(C)}$, belongs to V.

It is convenient to introduce the following notations with respect to problem P_{cntn}:
$C^0_{\max}$ – the optimal solution value,
$D = \{r = (r_1, r_2, \ldots, r_n) | r_j \geq 0, \sum_{j=1}^{n} r_j \leq m\}$ – the set of feasible resource allocations,
$U = \{u = (u_1, u_2, \ldots, u_n) | u_j = f_j(r_j), j = 1, \ldots, n, \ r \in D\}$ – the set of feasible *transformed* resource allocations,
conv$U$ – the convex hull of the set U, i.e., the set of all convex combinations of the elements of U.

Let $p = (p_1, \ldots, p_n)$. We now prove a theorem which characterizes an optimal solution of problem P_{cntn}. This theorem is illustrated in Fig. 1.

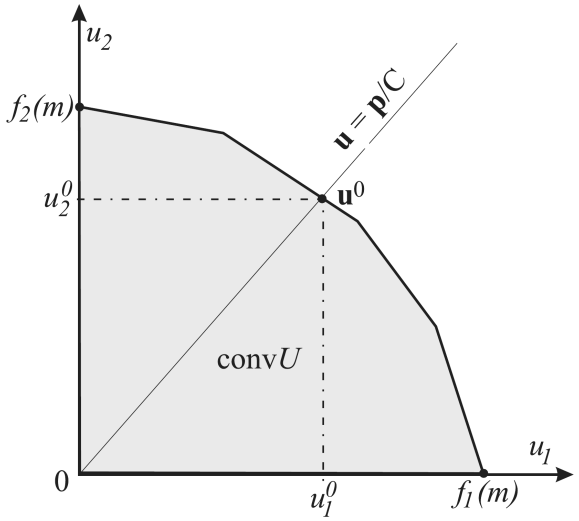

Fig. 1. An illustration of Theorem 2

Theorem 2. Let $u = p/C$ be a straight line in the space of transformed resource allocations given by the parametric equations $u_j = p_j/C$, $j = 1, \ldots, n$. Then

$$C^0_{\max} = \min\left\{C | C > 0, \ \frac{p}{C} \in \text{conv}U\right\}.$$

Proof. Since functions $f_j(r)$ are continuous and set D is a bounded polyhedron, set U is a bounded closed set. It is known that a convex hull of a bounded closed set is a bounded closed set. Thus, conv$U$ is a bounded closed set.

Let a vector function $r(t) = (r_1(t), r_2(t), \ldots, r_n(t))$, $t \in [0, C^*]$, define an optimal resource allocation with value C^*. Then, for each t, point $f(r(t)) = (f_1(r_1(t)), f_2(r_2(t)), \ldots, f_n(r_n(t)))$ belongs to the set conv$U$. According to Theorem 1, point

$$\left(\int_0^{C^*} f_1(r_1(t))dt/C^*, \int_0^{C^*} f_2(r_2(t))dt/C^*, \ldots, \int_0^{C^*} f_n(r_n(t))dt/C^* \right) = \left(\frac{p_1}{C^*}, \frac{p_2}{C^*}, \ldots, \frac{p_n}{C^*} \right)$$

belongs to the set conv$U$. Therefore, $C^0_{\max} \le C^*$ as required.

Let $f_j^{-1}(v)$ denote the reverse function of $f_j(r)$: $f_j^{-1}(f_j(r)) = r$, and let $f^{-1}(v) = (f_1^{-1}(v_1), \ldots, f_n^{-1}(v_n))$. According to Caratheodory's theorem, point

$$u^0 := \left(\frac{p_1}{C^0_{\max}}, \frac{p_2}{C^0_{\max}}, \ldots, \frac{p_n}{C^0_{\max}} \right) \in \text{conv}U$$

can be represented as a convex combination of L, $L \le n+1$, points, $v^{(1)}, \ldots, v^{(L)}$, of U: $u^0 = \sum_{l=1}^{L} \lambda_l v^{(l)}$, where $\sum_{l=1}^{L} \lambda_l = 1$. Then $p_j = u^0_j C^0_{\max} = \sum_{l=1}^{L} \lambda_l C^0_{\max} v^{(l)}_j$, $j = 1, \ldots, n$. We deduce that a schedule, in which there are L intervals of lengths $\Delta_l = \lambda_l C^0_{\max}$ and resource allocation $r^{(l)}_j = f_j^{-1}(v^{(l)}_j)$, $j = 1, \ldots, n$, $l = 1, \ldots, L$, is optimal, see Fig. 2.

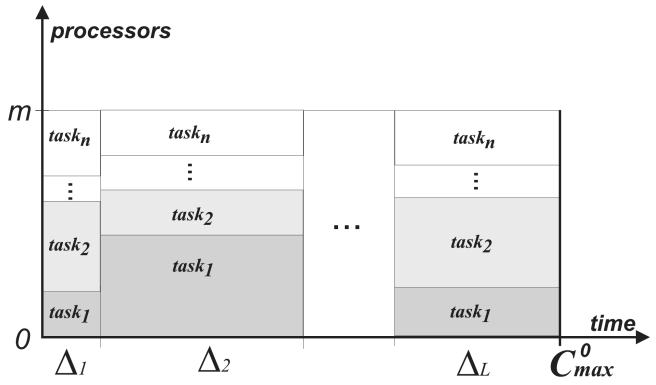

Fig. 2. An optimal schedule with L intervals of length Δ_l and makespan $C^0_{\max}$

Thus, problem P_{cntn} can be solved by enumerating and selecting appropriate transformed resource allocations $v^{(1)}, \ldots, v^{(L)}$.

The following theorem characterizes the case of strictly increasing piecewise linear continuous functions $f_j(r)$ such that $f_j(r) = a^{(k)}_j r + b^{(k)}_j$, $r \in [k, k+1]$, $k = 0, \ldots, m-1$, $j = 1, \ldots, m$.

Theorem 3. If $f_j(r)$ are strictly increasing piecewise linear continuous functions, then conv$U$ is a bounded polyhedron with vertices from the set U.

Proof. The set D of feasible resource allocations can be represented as a union of bounded *elementary* polyhedrons each of which is defined by $2n+1$ half-spaces $r_j \ge k_j$, $r_j \le k_j + 1$, $j = 1, \ldots, n$, and $r_1 + r_2 + \ldots + r_n \le m$, where $k_j \in \{0, 1, \ldots, m-1\}$. There are m^n elementary polyhedrons because k_i and k_j can take different values for different i and j. Linear continuous functions $a^{(k_j)}_j r + b^{(k_j)}_j$, $j = 1, \ldots, n$, $k_j \in \{0, 1, \ldots, m-1\}$, transform corresponding bounded elementary polyhedron into the bounded polyhedron defined by $2n$ half-spaces $u_j \ge a^{(k_j)}_j k_j + b^{(k_j)}_j$, $u_j \le a^{(k_j)}_j (k_j + 1) + b^{(k_j)}_j$ and half-space $\sum_{j=1}^{n} \frac{u_j - b^{(k_j)}_j}{a^{(k_j)}_j} \le m$. Thus, U is a union of a finite number of bounded polyhedrons and conv$U$ is a bounded polyhedron with all vertices belonging to U.

3. Convex functions $f_j(r)$

Similar to Błażewicz et al. [1], problem P_{cntn} with arbitrary n and m and convex processing speed functions admits an optimal solution in which there are n time intervals and in each interval one task is processed on all m processors. Below we give a justification.

We first prove that the intersection of the line p/C with conv$U$ lies on the hyperplane defined by the points $v^{(l)} = (f_1(0), \ldots, f_{l-1}(0), f_l(m), f_{l+1}(0), \ldots, f_n(0))$, $l = 1, \ldots, n$. The equality defining this hyperplane is

$$\frac{x_1}{f_1(m)} + \frac{x_2}{f_2(m)} + \ldots + \frac{x_n}{f_n(m)} = 1. \tag{1}$$

Consider an arbitrary feasible resource allocation $(r_1, r_2, \ldots, r_n)$ and the corresponding transformed resource allocation $(f_1(r_1), f_2(r_2), \ldots, f_n(r_n))$. By the convexity of $f_j(r)$ and the equality $f_j(0) = 0$, for $\lambda_j = r_j/m$, we have $(1 - \lambda_j) f_j(0) + \lambda_j f_j(m) \ge f_j((1 - \lambda_j)0 + \lambda_j m)$, which implies $f_j(r_j) \le \frac{r_j}{m} f_j(m)$. Furthermore, by substituting $x_j = f_j(r_j)$, $j = 1, \ldots, n$, in the left hand side of (1), we obtain

$$\sum_{j=1}^{n} \frac{f_j(r_j)}{f_j(m)} \le \sum_{j=1}^{n} \frac{\frac{r_j}{m} f_j(m)}{f_j(m)} \le \sum_{j=1}^{n} \frac{r_j}{m} \le 1.$$

Thus, all feasible transformed allocations lie on one side of this hyperplane. Therefore, the point of intersection of the line p/C with conv$U$ lies on this hyperplane.

We deduce that, in the case of convex processing speed functions, an optimal schedule for both problems P_{cntn} and P consists of n intervals with lengths $\Delta_j = p_j / f_j(m)$, and all m processors allocated to task j in the interval j, $j = 1, \ldots, n$. The corresponding algorithm, which is known as the gang strategy, can be implemented to run in $O(n)$ time if each value $f_j(m)$ is computable in a constant time.

4. Concave functions $f_j(r)$

Firstly, we consider problem P_{cntn} and prove that the set U of feasible transformed resource allocations is convex.

Consider two feasible transformed resource allocations, $f(r^{(1)})$ and $f(r^{(2)})$. Due to the concavity, we have $u := \lambda f(r^{(1)}) + (1-\lambda) f(r^{(2)}) \le f(\lambda r^{(1)} + (1-\lambda) r^{(2)})$ for any $0 < \lambda < 1$. Note that $r' := \lambda r^{(1)} + (1-\lambda) r^{(2)}$ is a feasible resource allocation. Since functions $f_j(r)$ are strictly increasing and continuous, it follows that $f^{-1}(u) \le r'$ and u is a feasible transformed resource assignment. Therefore, U is convex what implies that the point of intersection of the line p/C and conv$U$ belongs to U.

We deduce that there exists an optimal schedule for problem P_{cntn} with the single time interval $[0, C^0_{\max}]$ and the resource allocation $f^{-1}(p/C^0_{\max})$.

Note that a schedule with a single time interval $[0, C]$ and the resource allocation $f^{-1}(p/C)$ is feasible for any $C \ge C^0_{\max}$. We will use this property in the following bisection search algorithm for problem P_{cntn} with the concave processing speed functions. The algorithm finds a feasible schedule with the makespan $C^{(1)}_{\max} \le C^0_{\max} + \varepsilon$, where ε is any given rational number.

At the beginning we build a feasible schedule with n intervals of lengths $\Delta_j = p_j / f_j(m)$ and all m processors allocated to task j in interval j, $j = 1, \ldots, n$. We perform a bisection search in the range $[0, C^+]$, $C^+ = \lceil \sum_{j=1}^{n} \Delta_j \rceil$, of the optimal makespan value $C^0_{\max}$. For a trial value $C \in [A, B]$, where A and B are the current lower and upper bounds for $C^0_{\max}$, if the schedule with one interval $[0, C]$ and the corresponding resource allocation $f^{-1}(p/C)$ is feasible, then we re-set $B := C$, otherwise we re-set $A := C$. The procedure terminates when $B \le A + \varepsilon$.

The time complexity of this algorithm is $O(n(\log C^+ + \log 1/\varepsilon))$ multiplied by the time complexity of computing the reverse function $f_j^{-1}(v)$. If functions $f_j(r)$ are piecewise linear with at most m breakpoints, as they are in Błażewicz et al. [1], then every reverse function can be computed in $O(\log m)$ time.

Observe that the optimality of algorithms given in Błażewicz et al. [1, 20] for the case $n \le m$ is justified by the properties of optimal solutions established there. The relation $n \le m$ is used in the proofs of these properties but it is not present in the properties themselves. We have proved that the same properties hold for arbitrary n and m. Therefore, the algorithms from [1, 20] can be applied for the case of arbitrary n and m. In particular, the case of piecewise linear concave strictly increasing processing speed functions in problem P_{cntn} can be handled by the $O(n \max\{m, n \log^2 m\})$ time algorithm in [1].

For specific concave processing speed functions problem P_{cntn} can admit a faster solution algorithm. Same as Prasanna and Musicus [11], let us consider functions $f_j(r) = r^\alpha$, $j = 1, \ldots, n$, where $0 < \alpha < 1$. In this case, an optimal solution can be described analytically, as it is shown in the following theorem.

Theorem 4. Let $f_j(r) = r^\alpha$, $0 < \alpha < 1$, for each j, $1 \le j \le n$. Then there exists an optimal schedule with one processing interval $[0, C^0_{\max}]$ for each task and corresponding resource allocation $r_1, \ldots, r_n$ such that

$$C^0_{\max} = \sqrt{\frac{\sum_{j=1}^{n} (p_j)^{1/\alpha}}{\alpha v}},$$

$$r_j = v^{\frac{1}{2\alpha}} \left(p_j \frac{\sqrt{\alpha}}{\sqrt{\sum_{j=1}^{n} (p_j)^{1/\alpha}}} \right)^{1/\alpha}, \qquad 1 \le j \le n,$$

where

$$v = \frac{m^{2\alpha}}{\left(\sum_{j=1}^{n} \left(\frac{p_j \sqrt{\alpha}}{\sqrt{\sum_{j=1}^{n} (p_j)^{1/\alpha}}} \right)^{1/\alpha} \right)^{2\alpha}}. \tag{2}$$

Proof. Since functions $f_j(r) = r^\alpha$ are concave, U=conv$U$ and, according to Theorem 2, there exists an optimal schedule with one processing interval $[0, C^0_{\max}]$ for each task. The value $C^0_{\max}$ and the corresponding optimal resource allocation can be derived from the solution to the following convex programming problem.

$$\min C, \text{ subject to}$$

$$\sum_{j=1}^{n} r_j - m \le 0,$$

$$\frac{p_j}{C} - (r_j)^\alpha \le 0,$$

$$(r_1, r_2, \ldots, r_n, C) \in Q$$

$$:= \{(r_1, r_2, \ldots, r_n, C) | r_j > 0, 1 \le j \le n, C > 0\}.$$

Denote

$$x := (r_1, r_2, \ldots, C)$$

and

$$\lambda := (\lambda_1, \lambda_2, \ldots, \lambda_n, \lambda_{n+1}), \qquad \lambda \ge 0.$$

Lagrange function for this problem is

$$F(x, \lambda) = C + \lambda_{n+1} \left(\sum_{j=1}^{n} r_j - m \right) + \sum_{j=1}^{n} \lambda_j \left(\frac{p_j}{C} - (r_j)^\alpha \right).$$

Since there exists a solution to the above convex programming problem, there exists a point (x^0, λ^0) such that $F(x^0, \lambda) \le F(x^0, \lambda^0) \le F(x, \lambda^0)$, $x \in Q$, $\lambda \in R^{n+1}$, $\lambda \ge 0$. Let us make an additional assumption that $\lambda^0 > 0$. We will demonstrate the existence of the point x^0 and Lagrange multipliers λ^0 which satisfy these properties.

Firstly, point x^0 must be stationary, that is, equalities $\frac{\partial F}{\partial C} = 0$ and $\frac{\partial F}{\partial r_j} = 0$, $j = 1, \ldots, n$, must be satisfied.

Equality $\frac{\partial F}{\partial C} = 0$ is equivalent to

$$C = \sqrt{\sum_{j=1}^{n} \lambda_j p_j} \tag{3}$$

and equality $\frac{\partial F}{\partial r_j} = 0$, $1 \le j \le n$, is equivalent to

$$r_j = \left(\frac{\lambda_j}{\lambda_{n+1}\alpha}\right)^{\frac{1}{1-\alpha}}. \tag{4}$$

Furthermore, $x^0 = (r_1, r_2, \ldots, C)$ with r_j and C defined in (3) and (4) is a global minimum for $F(x, \lambda)$, $x \in Q$, because this minimum exists.

Equality $\frac{\partial F}{\partial \lambda_{n+1}} = 0$ is equivalent to

$$\sum_{j=1}^{n} r_j - m = 0 \tag{5}$$

and equality $\frac{\partial F}{\partial \lambda_j} = 0$ is equivalent to

$$r_j = \left(\frac{p_j}{C}\right)^{1/\alpha}. \tag{6}$$

Furthermore, if (5) and (6) hold, then any $\lambda^0 > 0$ is a global maximum for $F(x, \lambda)$.

In order to satisfy (3), (4) and (6), λ must be chosen such that

$$\frac{p_j}{\sqrt{\sum_{j=1}^{n} \lambda_j p_j}} = \left(\frac{\lambda_j \alpha}{\lambda_{n+1}}\right)^{\alpha/(1-\alpha)}.$$

This formula holds if we define λ_j, $1 \le j \le n+1$, as

$$\lambda_j = \frac{1}{\alpha v}(p_j)^{\frac{1}{\alpha}-1}, \qquad 1 \le j \le n, \tag{7}$$

$$\lambda_{n+1} = \left(\frac{1}{\alpha}\sum_{j=1}^{n}(p_j)^{\frac{1}{\alpha}}\right)^{\frac{1-\alpha}{2\alpha}} v^{-\frac{1-\alpha}{2\alpha}-1}, \tag{8}$$

where v is a positive number.

Re-write (3) and (4) using (7) and (8):

$$C^0_{\max} = \sqrt{\frac{1}{\alpha v}\sum_{j=1}^{n}(p_j)^{\frac{1}{\alpha}}}, \tag{9}$$

$$r_j = v^{\frac{1}{2\alpha}}\left(\frac{p_j\sqrt{\alpha}}{\sqrt{\sum_{j=1}^{n}(p_j)^{\frac{1}{\alpha}}}}\right)^{\frac{1}{\alpha}}, \qquad 1 \le j \le n. \tag{10}$$

Now define v by (2) so that formula (5) holds.

Formulas (3)–(5) and (6) are true. Therefore, for the point (x^0, λ^0) the following relations are satisfied: $F(x^0, \lambda) \le F(x^0, \lambda^0) \le F(x, \lambda^0)$ for $x \in Q$, $\lambda \in R^{n+1}$, $\lambda \ge 0$. Thus, an optimal solution of problem P_{cntn} has been determined.

Note that the above formulas suggest only an approximate solution to the problem because the rational power of a number cannot be calculated precisely. Assuming that the rational power is calculated precisely in a constant time, the problem with functions $f_j(r) = r^\alpha$, $j = 1, \ldots, n$, $0 < \alpha < 1$, is solvable in $O(n)$ time.

Finally, solutions of the continuous problems studied in this section can be transformed into solutions of the corresponding discrete problem P with the same makespan values in $O(n)$ time by the approach in Błażewicz et al. [20].

5. Arbitrary functions $f_j(r)$

Here we assume that $f_j(r)$ are piecewise linear strictly increasing continuous functions with $f_j(0) = 0$. In this case we propose a combinatorial exponential algorithm to solve problems P_{cntn} and P.

It is shown in Theorem 3 that D is a union of elementary polyhedrons each being an intersection of a unit n-dimensional cube and the half-space $r_1 + \ldots + r_n \le m$. Given $k_j \in Z$, $1 \le k_j \le m$, $j = 1, \ldots, n$, we call $(k_1, \ldots, k_n)$-*cube* the cube whose points $(r_1, \ldots, r_n)$ satisfy $k_j - 1 \le r_j \le k_j$, $j = 1, \ldots, n$.

Theorem 5. If $(k_1, \ldots, k_n)$-cube intersects with hyperplane $r_1 + \ldots + r_n = m$ in at least two points, then $m + 1 \le k_1 + \ldots + k_n \le m + n - 1$.

Proof. Let $k_1 + \ldots + k_n > m + n - 1$. Then $k_1 + \ldots + k_n \ge m + n$ and any point $(r_1, \ldots, r_n)$ of the $(k_1, \ldots, k_n)$-cube satisfies $r_1 + \ldots + r_n \ge m$. Furthermore, there is at most one such point where this relation is a strict equality. Similarly, if $k_1 + \ldots + k_n < m + 1$, then $k_1 + \ldots + k_n \le m$. Then any point $(r_1, \ldots, r_n)$ of the $(k_1, \ldots, k_n)$-cube satisfies $r_1 + \ldots + r_n \le m$, and there is at most one such point where this relation is a strict equality.

Theorem 6. If $(k_1, \ldots, k_n)$-cube and hyperplane $r_1 + \ldots + r_n = m$, $m \in Z$, intersect in at least two points, then the intersection of this cube and the halfspace $r_1 + \ldots + r_n \le m$ is a polyhedron. Furthermore, all vertices of this polyhedron belong to the set of vertices of this cube.

Proof. Each vertex of the resulting polyhedron is defined by the intersection of n hyperplanes among which there are facets of the cube and the hyperplane $r_1 + \ldots + r_n = m$. If a vertex is defined only by the facets of the cube, then it is clearly a vertex of the cube. If it is defined by n hyperplanes including the hyperplane $r_1 + \ldots + r_n = m$, then, since m is an integer, all its coordinates are integer. Therefore, it is a vertex of the cube as well.

The number of $(k_1, \ldots, k_n)$-cubes intersecting with the hyperplane $r_1 + \ldots + r_n = m$ in at least two points is $\sum_{r=\max\{m+1,n\}}^{m+n-1} C_{r-1}^{n-1}$, where C_n^k is the number of combinations of k elements out of n. Since $C_{n-1}^{k-1} \le C_n^k$ for $k \le n$, we have $\sum_{r=\max\{m+1,n\}}^{m+n-1} C_{r-1}^{n-1} \le \sum_{r=0}^{n+m-2} C_{n+m-2}^{r} = 2^{n+m-2}$.

Then the total number of vertices of the considered cubes is $O(2^{2n+m-2})$.

The algorithm we propose consists of two stages. In Stage 1 it finds all vertices of the $(k_1,\ldots,k_n)$-cubes that intersect with the hyperplane, and their transformations by functions $f_j(r)$. Vertices of the polyhedron conv$U$ belong to the set U. By Theorems 5 and 6 and the fact that functions $f_j(r)$ are continuous and piecewise linear, we deduce that vertices of the polyhedron conv$U$ are among the above mentioned $O(2^{2n+m-2})$ transformed vertices.

In Stage 2 the algorithm enumerates all hyperplanes each of which is defined by n *base points* among those found in Stage 1 and such that all the remaining points are on the same side of this hyperplane in the space R^n as the point $(0,\ldots,0)$. It is clear that all the hyperplanes that define facets of the polyhedron conv$U$ are among these hyperplanes. Calculate the intersection point of the line p/C with each such hyperplane and find the hyperplane with the maximum value of C, which is equal to $C^0_{\max}$. Denote this intersection point as u^0. Let $v^{(1)},\ldots,v^{(n)}$ be the base points for the hyperplane corresponding to $C^0_{\max}$. We solve a system of linear equalities $\sum_{l=1}^{n}\lambda_l v_i^{(l)} = u_i^0$, $i=1,\ldots,n$, and $\sum_{l=1}^{n}\lambda_l = 1$ for $\lambda_l \geq 0$, $l=1,\ldots,n$. Let the solution contain $L^+ \leq n$ positive values $\lambda_1,\ldots,\lambda_{L^+}$. Then an optimal schedule contains L^+ intervals. Interval l, $1 \leq l \leq L^+$, has length $\lambda_l C^0_{\max}$ and resource allocation vector $r^{(l)}$ corresponding to the point $v^{(l)}$. Recall that the resource allocations $r^{(l)}$, $l=1,\ldots,L^+$, have integer components according to Theorem 6. Therefore, the corresponding solution is optimal for both problems P_{cntn} and P.

A formal description of the algorithm, which we denote as Enum, is given below.

Algorithm Enum

Stage 1 Set $U_1=\emptyset$, $C^0_{\max}=0$ and $u^0=(0,\ldots,0)$. Consider $(k_1,k_2,\ldots,k_n)$-cubes such that $m+1 \leq k_1+\ldots+k_n \leq m+n-1$. For each such cube consider its vertices $r=(r_1,\ldots,r_n)$, $r_i \in \{k_i-1,k_i\}$, $i=1,\ldots,n$. For each such vertex, in $O(n)$ time find its transformed resource allocation $u=(f_1(r_1),\ldots,f_n(r_n))$ and add it to the set U_1. Let $U_1=\{u^{(1)},\ldots,u^{(K)}\}$.

Stage 2 Consider all n-element subsets of the set U_1. For each such subset $\{u^{(i_1)},\ldots,u^{(i_n)}\}$, find a hyperplane H that goes through the points of this subset in the space R^n. Let $u^{(i_j)}=(u_1^{(i_j)},\ldots,u_n^{(i_j)})$, $j=1,\ldots,n$, and let the hyperplane H be defined by an equation $a_1u_1+\ldots+a_nu_n+b=0$. Its coefficients are a solution of the following system of linear equations:

$$a_1u_1^{(i_1)}+\ldots+a_nu_n^{(i_1)}+b=0,$$

$$a_1u_1^{(i_2)}+\ldots+a_nu_n^{(i_2)}+b=0,$$

$$\ldots$$

$$a_1u_1^{(i_n)}+\ldots+a_nu_n^{(i_n)}+b=0,$$

which can be solved in $O(n^3)$ time.

For each hyperplane H, check if all points of the set U_1 are on the same side of this hyperplane as the point $(0,0,\ldots,0)$: if $a_1u_1^{(l)}+\ldots+a_nu_n^{(l)}$ has the same sign as b, then the point $u^{(l)} \in U_1$ is on the same side of H as the zero point. If all points of U_1 are on the same side of the hyperplane H as the zero point, then call this hyperplane *feasible*.

For each feasible hyperplane H calculate its intersection point u^H with line p/C: $u^H=(p_1/C,\ldots,p_n/C)$ where $a_1p_1/C+\ldots+a_np_n/C+b=0$. We have $C=-a_1p_1/b\ldots-a_np_n/b$. If $C>C^0_{\max}$, then re-set $C^0_{\max}:=C$ and $u^0:=u^H$.

Let $\{v^{(1)},\ldots,v^{(n)}\}$ be a subset of the set U_1 corresponding to the point u^0 with the maximum value $C^0_{\max}$. In $O(n^3)$ time solve a system of $n+1$ linear equalities $\sum_{l=1}^{n}\lambda_l v_i^{(l)}=u_i^0$, $i=1,\ldots,n$, and $\sum_{l=1}^{n}\lambda_l=1$ for $\lambda_l \geq 0$, $l=1,\ldots,n$. Let the solution contain $L^+ \leq n$ positive values $\lambda_1,\ldots,\lambda_{L^+}$. Then an optimal schedule contains L^+ intervals. Interval l, $1 \leq l \leq L^+$, has length $\lambda_l C^0_{\max}$ and resource allocation vector $r^{(l)}$ corresponding to the transformed resource allocation $v^{(l)}$.

Algorithm Enum can be implemented to run in $O(n^3 2^{(2n+m-2)n})$ time.

Example: There are three processors and two tasks with $p_1=10$, $p_2=25$, $f_1(r)=r$ for all r, and $f_2(r)=2r$ on the interval [0,1], $f_2(r)=2$ in the interval [1,2], and $f_2(r)=r$ in the interval [2,3]. Firstly, we obtain transformed points $v_1=(0,3)$, $v_2=(0,2)$, $v_3=(1,2)$, $v_4=(1,2)$, $v_5=(2,2)$, $v_6=(2,0)$, $v_7=(3,0)$. After this we find point $u_0=(1,2.5)$. This point lies on the line (v_1,v_5) and line p/C when $C=10$ (see Fig. 3). Then we establish $\frac{1}{2}v_1+\frac{1}{2}v_5=u_0$. Thus, an optimal schedule consists of two intervals, each of five time units length, with resource allocations $r^{(1)}=(0,3)$ and $r^{(2)}=(2,1)$.

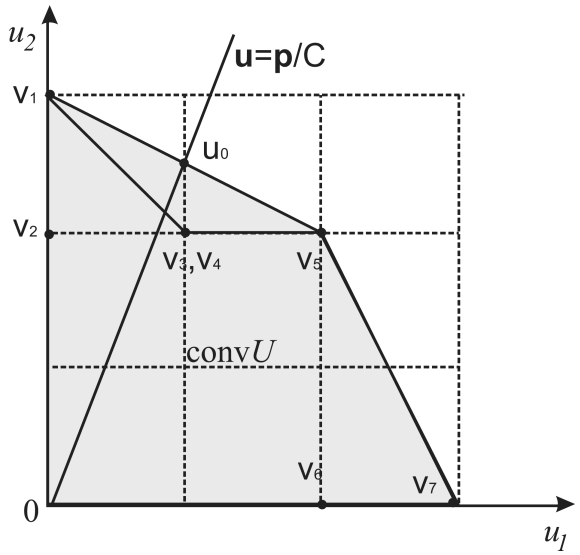

Fig. 3. The polyhedron conv U and intersection point of line p/C with set conv$U$

6. Conclusions

The earlier studies of malleable task scheduling problems were based on the properties proved for the case $m \geq n$. In this paper we present a formal proof of these properties for the general case. Furthermore, we suggest a bisection search $O(n(\log C^+ + \log 1/\varepsilon))$ time algorithm for the problem with arbitrary concave processing speed functions and an analytical $O(n)$ time solution approach for the problem with the earlier studied functions $f_j(r) = r^\alpha$, $j = 1, \ldots, n$, where $0 < \alpha < 1$. For the case of general functions $f_j(r)$, we presented a combinatorial exponential algorithm with run time $O(n^3 2^{(2n+m-2)n})$.

For future research it is interesting to find formulas and efficient exact and approximate solution algorithms for other practically motivated types of processing speed functions.

Also more general models of tasks, as these studied e.g. in [21] could be considered.

Acknowledgements. This paper was partially supported by the grant No. N N519 643340.

REFERENCES

[1] J. Błażewicz, M.Y. Kovalyov, M. Machowiak, D. Trystram, and J. Węglarz, "Scheduling malleable tasks on parallel processors to minimize the makespan", *Ann. Oper. Res.* 129 (1–4), 65–80 (2004).

[2] P.E. Bernard, T. Gautier, and D. Trystram, "Large scale simulation of parallel molecular dynamics", *Proc. 13. IPPS / 10. SPDP* 1, 638–644 (1999).

[3] J. Dongarra, L. Duff, D. Danny, C. Sorensen, and H. van der Vorst, *Numerical Linear Algebra for High Performance Computers (Software, Environments, Tools)*, Society for Industrial & Applied Mathematics, London, 1999.

[4] J. Du and J.Y.-T. Leung "Complexity of scheduling parallel tasks systems", *SIAM J. Discrete Math.* 2 (4), 473–487 (1989).

[5] L. Bianco, J. Błażewicz, P. Dell'Olmo, and M. Drozdowski, "Scheduling multiprocessor tasks on a dynamic configuration of dedicated processors", *Ann. Oper. Res.* 58, 493–517 (1995).

[6] J. Błażewicz, P. Dell'Olmo, M. Drozdowski, and M.G. Speranza, "Scheduling multiprocessor tasks on 3 dedicated processors", *Inform. Process. Lett.* 49, 269–270 (1994).

[7] J. Turek, J. Wolf, and P. Yu, "Approximate algorithm for scheduling parallelizable tasks", *Proc. ACM SPAA* 1, 323–332 (1992).

[8] W.T. Ludwig, "Algorithms for Scheduling malleable and nonmalleable parallel tasks", *PhD Thesis*, University of Wisconsin-Madison, Department of Computer Science, Madison, 1995.

[9] G. Mounie, C. Rapine, and D. Trystram, "Efficient approximation algorithms for scheduling malleable tasks", *Proc. ACM SPAA* 1, 23–32 (1999).

[10] G. Mounie, C. Rapine, and D. Trystram, "A 3/2-dual approximation algorithm for scheduling independent monotonic malleable tasks", *SIAM J. Comput.* 37 (2), 401–412 (2007).

[11] G.N.S. Prasanna and B.R. Musicus, "The optimal control approach to generalized multiprocessor scheduling", *Algorithmica* 15 (1), 17–49 (1995).

[12] J. Błażewicz, T.C.E. Cheng, M. Machowiak, and C. Oguz, "Berth and quay crane allocation: a moldable task scheduling model", *J. Oper. Res. Soc.* 62 (7), 1189–1197 (2011).

[13] M. Drozdowski, "Scheduling multiprocessor tasks – an overview", *Eur. J. Oper. Res.* 94 (2), 215–230 (1996).

[14] J. Błażewicz, K. Ecker, B. Plateau, and D. Trystram, *Handbook on Parallel and Distributed Processing*, Springer, Berlin, 2000.

[15] J. Węglarz, "Time-optimal control of resource allocation in a complex of operations framework", *IEEE T. Syst. Man Cyb.* 6 (11), 783–788 (1976).

[16] J. Węglarz, "Multiprocessor scheduling with memory allocation – a deterministic approach", *IEEE T. Comput.* 29 (8), 703–709 (1980).

[17] P. Sanders and J. Speck, "Efficient parallel scheduling of malleable tasks", *IEEE Conf. IPDPS* 1, 1156–1166 (2011).

[18] K. Jansen and L. Porkolab, "Computing optimal preemptive schedules for parallel tasks: linear programming approaches", *Math. Program.* 95 (3), 617–630 (2003).

[19] K. Jansen, "Scheduling malleable parallel tasks: an asymptotic fully polynomial time approximation scheme", *Algorithmica* 39 (1), 59–81 (2004).

[20] J. Błażewicz, M.Y. Kovalyov, M. Machowiak, D. Trystram, and J. Węglarz, "Preemptable malleable task scheduling problem", *IEEE T. Comput.* 55 (4), 485–490 (2006).

[21] J. Klamka, "Controllability of dynamical systems. A survey", *Bull. Pol. Ac.: Tech.* 61 (2), 221–229 (2013).