Geometrical Heuristics for Multiprocessor Flowshop Scheduling with Uniform Machines at Each Stage

S.V. Sevastianov*)

Institute of Mathematics,
Siberian Branch of the Russian Academy of Sciences,
prosp. Koptjuga 4, 630090, Novosibirsk-90, Russia
email: seva@math.nsc.ru

*)Supported by the Russian Foundation for Fundamental Research (Grant 99-01-00581).

Abstract

We consider the multi-stage multiprocessor flowshop scheduling problem with uniform machines at each stage and the minimum makespan objective. Using a vector summation technique, three polynomial-time heuristics are developed with absolute worst-case performance guarantees. As a direct corollary, in the special case of the ordinary flowshop problem we come to the best approximation algorithms (both in absolute performance guarantee and in running time) known for this problem.

Keywords: Multiprocessor Flowshop Scheduling, Uniform Parallel Machines, Approximation, Worst-case analysis.

1 Introduction

1.1 Problem definition

There are n jobs $\{J_1, \ldots, J_n\} = \mathcal{J}$ and r machine shops $\mathcal{M}_1, \ldots, \mathcal{M}_r$. The qth shop consists of m_q uniform machines: $\mathcal{M}_q = \{M_{q,1}, \ldots, M_{q,m_q}\}$, machine $M_{q,\nu}$ being characterized by a *speed* $s_{q,\nu}$ ($s_{q,\nu} \geq 1$). Each job J_j ($j = 1, \ldots, n$) represents a chain of operations $(o_1^j, \ldots, o_r^j)$ that have to be processed in this order. (We will also say that each job has r *stages* of processing.) Operation o_q^j can be processed by any (but only one) machine in the shop $\mathcal{M}_q$ and is characterized by a *volume of work* v_q^j to be performed. If an operation o_q^j is assigned to machine $M_{q,\nu} \in \mathcal{M}_q$, then it requires $v_q^j/s_{q,\nu}$ time units to be completed.*) Each machine can process at most one job at a time. An operation o_{q+1}^j may start only after the previous operation o_q^j of job J_j is completed. We assume that all jobs are available at time zero and that interruptions in the processing of any operation are not allowed. The objective is to find a schedule S (i.e., to assign a machine $M(o_q^j) \in \mathcal{M}_q$ and a nonnegative starting time $s(o_q^j)$ to every operation o_q^j) which minimizes the makespan (the maximum completion time over all operations) $C_{\max}(S)$.

Thus, the problem under consideration represents a "hybrid" of two scheduling problems: $F||C_{\max}$ and $Q||C_{\max}$, and following Schuurman and Woeginger [10] who used the notation $F(P)||C_{\max}$ or $Fr(P)||C_{\max}$ for the hybrid of $P||C_{\max}$ with either $F||C_{\max}$ or $Fr||C_{\max}$ (with a fixed number r of shops), we will denote our problem by $F(Q)||C_{\max}$ or $Fr(Q)||C_{\max}$ — for the case of an arbitrary or a fixed number r of shops, respectively.

1.2 Relevant results

First of all, as is shown by S.M. Johnson [6], the two-shop problem $F2||C_{\max}$ without parallel machines is polynomially solvable, whereas $Fr||C_{\max}$ is strongly NP-hard for any fixed $r \geq 3$ [2]. The simplest problem with parallel machines $F2(P)||C_{\max}$ with $m_1 = 1,\ m_2 = 2$ (or by symmetry, the one with $m_1 = 2,\ m_2 = 1$) is shown to be strongly NP-hard [5]. Therefore, it is natural to consider algorithms which construct approximate solutions in polynomial time.

Rather than survey the entire area of approximation algorithms for scheduling, we will only discuss those results most relevant to our work. In particular we will omit results without theoretical performance guarantees and results with ratio performance guarantees. For a broader survey we direct the reader to [10]. We thereby entirely focus on approximation algorithms with absolute performance guarantees. For each such algorithm it will be proved that, given an instance of the problem, the difference between its optimal makespan and the length of the schedule constructed by the algorithm is never greater than a certain amount that will be called an *absolute guarantee* (or a.g., for short). If such a guarantee turns out to be independent of the number of jobs n (as is in our case), this implies that the algorithm is asymptotically optimal on a sequence of instances

*) To keep our calculation correct, we will assume that volumes of work are also measured in time units. whereas machine speeds have a null dimension.

with $n \to \infty$ under easily satisfiable conditions on problem data. Thus, we get a rare opportunity to construct near-optimal schedules for strongly NP-hard problems, while remaining within not only theoretically (like in PTASes invented by Leslie Hall [3] for $Fr||C_{\max}$, by Hochbaum and Shmoys [4] for $P||C_{\max}$, and by Schuurman and Woeginger [10] for $F2(P)||C_{\max}$) but also practically polynomial-time algorithms. Moreover, our algorithms perform well even when the number of stages (shops) r and the total number of machines m are part of the input and may increase in the sequence of instances, though somewhat slower than the number of jobs n. (For instance, it follows from Theorem 8 that if the maximum ratio of machine speeds and the maximum volume of work of an operation are bounded above by constants, while $r = o(n^{1/2})$ and $m = o(n)$, then the solution obtained becomes asymptotically optimal as $n \to \infty$.)

To formulate relevant results, we need some notation. Let $S_q = \sum_{\nu=1}^{m_q} s_{q,\nu}$ be the total speed over all machines in the qth shop; $S_{\min} = \min_q S_q$; $S_{\max} = \max_q S_q$; $s_{q,\min} = \min_{M_i \in \mathcal{M}_q} s_{q,i}$ $l_q = \sum_{J_j \in \mathcal{J}} v_q^j / S_q$ be the average machine load in the qth shop; $l_{\max} = \max_q l_q$; $m_{\min} = \min_q m_q, m_{\max} = \max_q m_q$; $v_{\max} = \max_{j,q} v_q^j$ be the maximum volume of work of an operation; $p_q^j = v_q^j / S_q$ be the *shop processing time* of an operation o_q^j; $p_{\max} = \max_{j,q} p_q^j$ be the maximum shop processing time of an operation, and let $C^*_{\max}$ stand for the optimum value of the objective function.

Since $l_{\max}$ is an evident lower bound on the optimum makespan $C^*_{\max}$, to derive an upper bound on $C_{\max}(S_H) - C^*_{\max}$, it suffices to estimate the difference $C_{\max}(S_H) - l_{\max}$ from above. So, while speaking about an a.g. for some heuristic, we will mean some upper bound on the difference $C_{\max}(S_H) - l_{\max}$.

Parameters of the best known approximation algorithms with absolute performance guarantees for the $F||C_{\max}$ problem with n jobs and r shops are presented in Table 1. Columns 1 to 4 contain: 1) the number of stages (shops) r; 2) an a.g., i.e., an upper bound on the difference $C_{\max}(S) - C^*_{\max}$ that can be guaranteed for the solution S obtained by algorithm $\mathcal{A}_r$; 3) an upper bound $T(\mathcal{A}_r)$ on the running time of the algorithm; 4) a reference to a source paper. In Table 2 we summarize the best known results for the more general problem $F(P)||C_{\max}$.

r	a.g.$^{*)}$	$T(\mathcal{A}_r)$	source
2	1	n	trivial
3	3	n	[14]
4	6	$nr + n \log n$	[14]
$r \geq 5$	$r^2 - 3r + 3 + 1/(r-2)$	$n^2 r^2$	[13]

Table 1: Parameters of the best known algorithms $\mathcal{A}_r$ for $F||C_{\max}$;
$^{*)}$ $v_{\max}$ is scaled to one

It should be mentioned here about the geometrical method used for deriving the results accumulated in Tables 1 and 2 (since we are going to develop the method in the current paper to obtain similar results for a more general $F(Q)||C_{\max}$ problem). The method

r	known a.g.*)	$T(\mathcal{A}_r)$	source	new a.g.*) †)
2	$2-1/m_{\max}$	nm	[7]	$2-1/m_{\min}$
3	7	nm	[7]	4
$r \geq 4$	r^2-1	n^2r^2	[12]	$r^2-3r+3+1/(r-2)$

Table 2: Parameters of known and new algorithms $\mathcal{A}_r$ for $F(P)||C_{\max}$;
*) $v_{\max}$ is scaled to one;
†) the bounds on the running time are the same

consists in a reduction of the initial scheduling problem to a so called *compact vector summation* problem (in two versions: *strict* and *nonstrict* summation of vectors). Such a reduction works as follows.

For every job J_j $(j = 1, \ldots, n)$ of a given instance of the r-stage flowshop problem, an $(r-1)$-dimensional vector d_j is defined so that the total sum of vectors is equal to zero. In the compact vector summation problem, we wish to find an order of summing the vectors d_j so that all partial sums (or "almost all" — in the nonstrict vector summation) were in an optimal (according to a certain objective function) family of half-spaces in $\mathbb{R}^{r-1}$. To obtain an approximate solution with an a.g. for the vector summation problem, one of the algorithms developed by the author in his previous papers can be applied. The permutation found specifies the job order (due to the one-to-one correspondence between the vectors and the jobs). Next, the operations in each shop are distributed to machines in this order (or in the reverse order — for the last shop), after which the operations distributed to the same machine are sequenced according to the job order. Finally, the earliest schedule is constructed subject to the order of processing the operations of each job and the defined above order of operations on each machine. It is then shown that the a.g. obtained for the vector summation problem can be transformed to an a.g. for the flowshop problem.

Now we pass on to an overview of known results on the $F(Q)||C_{\max}$ problem. The earliest analysis of $F(Q)||C_{\max}$ of which we are aware is due to Kyparisis and Koulamas [8]. Using the vector summation technique and known ([11]) algorithms of strict vector summation, they developed a polynomial-time approximation algorithm with an a.g. of the form

$$C_{\max}(S_H) - C^*_{\max} \leq \left(r^2 - 3r + 3 + \frac{1}{r-2}\right) p_{\max} + \left(\sum_{q=1}^{r}(m_q - 1)/S_q + \sum_{q=2}^{r-1}(S_q - s_{q,i_q})/S_q\right) v'_{\max}, \quad (1)$$

where $v'_{\max}$ is the value of $v_{\max}$ for a *modified* matrix of volumes of work obtained from the original matrix as a result of the application of a Machine Load Adjustment Procedure. Although $v'_{\max}$ is not a characteristic of the original instance and depends on some procedure that increases the original volumes of work, however it can be bounded above

by $v_{\max}S_{\max}/S_{\min}$ (or by $p_{\max}S_{\max}$). Therefore, an a.g. independent of the number of jobs n can also be derived here.

1.3 Our results

The contribution of our paper is the following. We present three heuristics with a.g.s for the $F(Q)||C_{\max}$ problem. Since we normally use the same vector summation technique, our formulas look somewhat similar to (1) but with a little difference: they do not depend on strange parameters such as $v'_{\max}$.

For the first heuristic H_1, we derive an a.g. formulated in units of $p_{\max}$ (Theorem 7). Within the framework of this heuristic, different algorithms for solving an auxiliary vector summation problem can be applied, which results in a family of algorithms for $F(Q)||C_{\max}$ with different a.g.s and different bounds on running time (see Theorem 8). In particular, for the cases of r listed in Table 1, we obtain a series of algorithms with running time identical to those presented in the table and such that in the case of the ordinary flowshop problem, we obtain a.g.s exactly the same as presented in Table 1. Thus, our heuristic H_1 can be viewed as an extension of the best known results for $F||C_{\max}$.

For heuristics H_2 and H_3 we prove a.g.s formulated in terms of $v_{\max}$ units only (Theorem 10). As a direct corollary, we formulate an a.g. which is independent of any shop characteristic (such as the number of machines and their speeds) and is identical to the a.g. presented in Table 1 for the $F||C_{\max}$ problem in the case of arbitrary $r \geq 5$. Thus we obtain another extension of the algorithm with the best known a.g. for the $F||C_{\max}$ problem to the case of $F(Q)||C_{\max}$ problem. It is also clear that the same a.g. is valid for the case of identical machines ($F(P)||C_{\max}$) thereby improving the a.g. presented in Table 2 for the case of arbitrary r.

Since the $F3(Q)||C_{\max}$ problem reduces to a one-dimensional vector summation problem, it is possible to solve the latter with the same guarantee as in the general case, but now in linear time instead of $O(n^2)$. Substituting the value $r = 3$ to the general formula, we obtain an a.g. for the case of three stages: $C_{\max}(S_{H_2}) - l_{\max} \leq 4v_{\max}$, which clearly improves the known a.g. presented in Table 2 for the $F3(P)||C_{\max}$ problem.

Finally, we present a linear-time heuristic H_3 for the two-stage case ($F2(Q)||C_{\max}$) with an a.g.

$$C_{\max}(S_{H_3}) - l_{\max} \leq (2 - 1/S_{\min})v_{\max}.$$

Applying this to the special case of identical machines, we obtain a linear time algorithm with an a.g.

$$C_{\max}(S_{H_3}) - l_{\max} \leq (2 - 1/m_{\min})v_{\max}$$

which is similar to the result in Table 2, but with one important difference: we use $m_{\min}$ instead of $m_{\max}$. Thus, in the special cases when $m_1 = 1$ or $m_2 = 1$, we achieve the best possible a.g.:

$$C_{\max}(S_{H_3}) - l_{\max} \leq v_{\max}.$$

(It can be attained even for the ordinary flowshop problem.)

The remainder of the paper is organized as follows. In Section 2 we formulate auxiliary geometrical results used later in Sections 3 and 4. In Section 3 we present heuristic H_1 (with an a.g. in terms of $p_{\max}$), whereas Section 4 describes heuristics H_2 and H_3 with a.g.s in terms of $v_{\max}$.

2 Vector summation in $\mathbb{R}^m$

Let us define some geometrical notions and introduce necessary notation.

An m-dimensional real space is denoted by $\mathbb{R}^m$. Given a family of vectors $X = \{x_1, \ldots, x_n\} \subset \mathbb{R}^m$, its *convex hull* is denoted by conv $\{x_1, \ldots, x_n\}$. Similarly, for sets $B' \subset \mathbb{R}^m$ and $B'' \subset \mathbb{R}^m$, conv $\{B', B''\}$ will stand for the convex hull of $B' \cup B''$. The standard *basis vectors* in $\mathbb{R}^m$ are denoted by $e_1, \ldots, e_m$. Thus, for any vector $x = (x(1), \ldots, x(m))$ we have $x = \sum_{i=1}^m x(i)e_i$. For any two vectors $x = (x(1), \ldots, x(m))$ and $y = (y(1), \ldots, y(m))$, we can define their *inner product* (x, y) by $(x, y) \doteq \sum_{i=1}^m x(i)y(i)$.

Given a vector $c \in \mathbb{R}^m$ and a number $\beta \in \mathbb{R}$, $P(c, \beta)$ will stand for the closed half-space $\{x \in \mathbb{R}^m \,|\, (c, x) \le \beta\}$.

A finite family of vectors $\{x_1, \ldots, x_n\} \subset \mathbb{R}^m$ is called a 0-*family* if $\sum x_i = 0$.

A word combination "permutation $\pi = (\pi_1, \ldots, \pi_n)$" will always mean "a permutation $\pi = (\pi_1, \ldots, \pi_n)$ of indices $\{1, 2, \ldots, n\}$".

We say that permutation $\pi = (\pi_1, \ldots, \pi_n)$ *provides strict summation of vectors* $\{x_1, \ldots, x_n\} = X$ *within a set* $Y \subset \mathbb{R}^m$ (and denote this fact by $(X, \pi) \in S_m Y$), if all partial sums $x_\pi^k \doteq \sum_{i=1}^k x_{\pi_i}$, $k = 1, \ldots, n$, are in the set Y. The latter means that while summing the vectors $\{x_1, \ldots, x_n\}$ in order π, the whole summing trajectory lies within the set Y.

We say that permutation $\pi = (\pi_1, \ldots, \pi_n)$ *provides nonstrict summation of vectors* $\{x_1, \ldots, x_n\} = X$ *within a set* $Y \subset \mathbb{R}^m$ (and denote this fact by $(X, \pi) \in NS_m Y$), if for any $k = 1, \ldots, n$, the relation $x_\pi^{k-1} \notin Y$ implies $x_\pi^k \in Y$. In other words, while summing the vectors nonstrictly in Y, we allow the summing trajectory to go out of Y sometimes, but every time as the summing trajectory goes out of Y, it must return to Y at the next step. (In particular, if Y does not contain the origin, and hence, $x_\pi^0 \notin Y$, then x_π^1 must be in Y.)

We also say that π provides strict (nonstrict) summation of vectors X within a family of sets $\mathcal{Y} = \{Y_1, \ldots, Y_l\}$ (and denote this fact by $(X, \pi) \in S_m\mathcal{Y}$, or $(X, \pi) \in NS_m\mathcal{Y}$, respectively), if it provides strict (nonstrict) summation of X within each set $Y_i \in \mathcal{Y}$.

Let a norm $s : \mathbb{R}^m \to \mathbb{R}^+$ be defined in $\mathbb{R}^m$. A finite family of vectors $\{x_1, \ldots, x_n\} \subset \mathbb{R}^m$ is called an s-*family* if it is a 0-family, and $\|x_j\|_s \le 1$, $\forall j = 1, \ldots, n$.

Now we define a special norm in $\mathbb{R}^m$ that plays an important role in applications of the vector summation technique to scheduling problems. This norm is normally denoted by $\hat{s}$ and defined in $\mathbb{R}^m$ by its unit ball $B_{m,\hat{s}}$ specified by the formula

$$B_{m,\hat{s}} \doteq \{x \in \mathbb{R}^m \mid |x(i_1)| \le 1,\ |x(i_1) - x(i_2)| \le 1,\ \forall\ i_1, i_2 = 1, \ldots, m\}.$$

(It is clear that having the unit ball $B \subset \mathbb{R}^m$ defined, we can definitely compute the norm of any vector $x \in \mathbb{R}^m$ by the formula $\|x\| = \min\{\lambda > 0 \,|\, x/\lambda \in B\}$ — for $x \ne 0$, and

$\|x\| = 0$ — for $x = 0$.) We next formulate two minimization problems.

NVS1(m)-problem. Given an $\hat{s}$-family of vectors $X = \{x_1, \ldots, x_n\} \in \mathbb{R}^m$, we wish to find a permutation $\pi = (\pi_1, \ldots, \pi_n)$ and a family of real numbers $b = \{\beta_1, \ldots, \beta_{m+1}\}$ such that

- π provides nonstrict summation of vectors X within the family of half-spaces

$$\mathcal{P}_1(m, b) = \{P(e_1 - e_2, \beta_1), \ldots, P(e_{m-1} - e_m, \beta_{m-1}), P(e_m, \beta_m), P(-e_1, \beta_{m+1})\};$$

- b gives the minimum to the objective function $\theta_1(b) \doteq \sum_{i=1}^{m+1} \beta_i$.

NVS2(m)-problem. Given an $\hat{s}$-family of vectors $X = \{x_1, \ldots, x_n\} \in \mathbb{R}^m$, we wish to find a permutation $\pi = (\pi_1, \ldots, \pi_n)$ and a family of real numbers $b = \{\beta_1, \ldots, \beta_m\}$ such that

- π provides nonstrict summation of vectors X within the family of half-spaces

$$\mathcal{P}_2(m, b) = \{P(e_1 - e_2, \beta_1), \ldots, P(e_{m-1} - e_m, \beta_{m-1}), P(e_m, \beta_m)\};$$

- b gives the minimum to the objective function $\theta_2(b) \doteq \sum_{i=1}^{m} \beta_i$.

We also introduce two analogous problems VS1(m) and VS2(m) about strict vector summation. They differ from the corresponding NVS1(m)- and NVS2(m)-problem in two points: first, we deal with 0-families of vectors instead of $\hat{s}$-families, and second, the requirement of nonstrict summation should be replaced by that of strict summation.

In the following lemma, a reduction from the NVS2(m)-problem to the NVS1($m-1$)-problem is stated. We now define a hyperplane $H = \{x \in \mathbb{R}^m \mid x(1) = 0\}$. For a given vector $x \in \mathbb{R}^m$, let x' be the *orthogonal projection* of x onto H, i.e., $x' = x - x(1)e_1$. Given a family of vectors $X = \{x_1, \ldots, x_n\} \subset \mathbb{R}^m$, we define $X_H = \{x'_1, \ldots, x'_n\}$.

Lemma 1 (see Lemma 2 [14, p.247]) *Fix $m \geq 2$. Let $\mathcal{A}'$ be an algorithm for the* NVS1($m-1$)*-problem which runs in time $T(\mathcal{A}')$. There exists an algorithm $\mathcal{A}$ for the* NVS2(m)*-problem which runs in time $O(T(\mathcal{A}'))$ and for which*

$$\theta_2(b(X)) \leq \theta_1(b'(X_H))$$

holds for any input X of the NVS2(m)*-problem, where X_H is defined for X as said above.*

Using the same argument, we can prove a similar lemma for the VS2(m)- and VS1($m-1$)-problems.

Lemma 2 *Fix $m \geq 2$. Let $\mathcal{A}'$ be an algorithm for the* VS1($m-1$)*-problem which runs in time $T(\mathcal{A}')$. There exists an algorithm $\mathcal{A}$ for the* VS2(m)*-problem which runs in time $O(T(\mathcal{A}'))$ and for which*

$$\theta_2(b(X)) \leq \theta_1(b'(X_H))$$

holds for any input X of the VS2(m)*-problem, where X_H is defined for X as said above.*

We will also need the following

Theorem 3 (see [11]) *For any 0-family of vectors* $\{x_1, \ldots, x_n\} = X \subset \mathbb{R}^m$, $m \geq 2$, *and any vector* $a \in \mathbb{R}^m$, *a permutation* $\pi = (\pi_1, \ldots, \pi_n)$ *can be found in* $O(n^2m^2)$ *time such that* $(X, \pi) \in S_m\{(m-1)B(X) + B_a\}$, *where* $B(X) = \text{conv}\,\{x_1, \ldots, x_n\}$, $B_a = \text{conv}\,\{0, a - \frac{1}{m}B(X)\}$. □

Considering s-families of vectors in Theorem 3 and choosing vector $a = 0$, we obtain the following

Corollary 4 *Given an s-family of vectors* $\{x_1, \ldots, x_n\} = X \subset \mathbb{R}^m$, *a permutation* $\pi = (\pi_1, \ldots, \pi_n)$ *can be found in* $O(n^2m^2)$ *time such that* $(X, \pi) \in S_m\left\{(m - 1 + \frac{1}{m})B_{m,s}\right\}$. □

In the case of $m = 1$, we have a property of 0-families similar to that formulated in Theorem 3, except that instead of speaking about vector summation we now may speak about summation of numbers within an interval, which can be done much faster.

Theorem 5 *For any 0-family of numbers* $\{x_1, \ldots, x_n\} = X \subset [-a, b]$, $a \geq 0, b \geq 0$, *and any number* $\alpha \in [-a, b]$, *a permutation* $\pi = (\pi_1, \ldots, \pi_n)$ *can be found in* $O(n)$ *time such that* π *provides strict summation of numbers* X *within the interval* $[-a - \alpha, b - \alpha]$. □

(We leave the proof to the reader.) Finally, we will use

Lemma 6 (see Corollary 1 [14, p.259]) *The* NVS1(2)*-problem can be solved in* $O(n \log n)$ *time with an a.g.* $\sum \beta_i \leq 3$. □

3 Heuristic H_1

Heuristic H_1 consists of five steps described below.

Step 0 (adjusting the machine loads).
Increase the volumes of work v_q^j of some operations o_q^j without increasing the values of $l_{\max}$ and $p_{\max}$, until the equation

$$l_q = l_{\max} \tag{2}$$

is achieved for every $q = 1, \ldots, r$. This is possible, because each l_q is a sum of exactly n summands p_q^j, whereas $p_{\max} = \max_{j,q} p_q^j$. It is also easily seen that this step can be performed in $O(nr)$ time.

For convenience of notation, we will assume that v_q^j, p_q^j, and l_q denote the adjusted values. Furthermore, in this section it will be assumed that $p_{\max}$ is scaled to one, which implies $p_q^j \leq 1$, $\forall\ j, q$. For each job J_j, $j = 1, \ldots, n$, we define the vector of shop processing times of its operations

$$p_j = (p_1^j, \ldots, p_r^j)$$

and the vector of differences of processing times

$$d_j = (d_j(1), \ldots, d_j(r-1)) = (p_1^j - p_r^j, \ldots, p_{r-1}^j - p_r^j).$$

Due to (2), we have $\sum_{j=1}^{n} d_j = 0$. Since we also have $|d_j(i')| \le p_{\max} = 1$ and $|d_j(i') - d_j(i'')| = |p^j_{i'} - p^j_{i''}| \le 1$, $\forall i', i'' \in \{1, \ldots, r-1\}$, the family of vectors $\{d_1, \ldots, d_n\}$ represents an $\hat{s}$-family in $\mathbb{R}^{r-1}$.

Step 1 (computing a permutation of the jobs).

Fix a permutation π of the jobs and for notational convenience assume that the jobs are already numbered according to this permutation. (In other words, we assume that $\pi = (1, 2, \ldots, n)$. Later we will specify several different methods to choose the permutation π. In every case, an appropriate algorithm for solving one of the above formulated vector summation problems will be applied.)

Step 2 (distribution of the qth stage operations, $q = 1, \ldots, r-1$, to the qth shop machines).

For every $q = 1, \ldots, r-1$, we distribute the operations $o^1_q, \ldots, o^n_q$ (in this order) to m_q machines of the qth shop according to the following LS-rule (List Scheduling): the next in turn operation should be distributed to the machine that provides minimum to the completion time of the operation. On each machine $M_{q,i} \in \mathcal{M}_q$ process the operations according to the order of jobs chosen at Step 1.

Step 3 (distribution of the rth stage operations to the rth shop machines).

For the rth shop, we distribute its operations to machines according to an RLS-rule [†)] (Reverse List Scheduling, see [1, 9]): distribute the operations to machines according to the LS-rule, but in the reverse order: $o^n_r, \ldots, o^1_r$; the operations distributed to each machine $M_{r,i} \in \mathcal{M}_r$ should be scheduled according to the order of jobs chosen at Step 1.

Step 4 (scheduling).

Construct the *earliest* feasible schedule subject to the operation orders determined for each machine and each job. The schedule obtained (i. e., the distribution of the operations to machines and the starting times of the operations) is taken for a schedule of the initial problem instance (with shorter operations), and it clearly remains feasible.

This completes the description of heuristic H_1.

Theorem 7 *Suppose that for some $r \ge 2$ we have an algorithm $\mathcal{A}^*_r$ that solves the* NVS2$(r-1)$*-problem with an a.g. $\sum \beta_i \le \theta^*(r)$ in $T(\mathcal{A}^*_r)$ time. Then the $F(Q)||C_{\max}$ problem with n jobs, r shops, and m machines can be solved in $O(nm + T(\mathcal{A}^*_r))$ time with an a.g.*

$$C_{\max}(S) - l_{\max} \le \left(\theta^*(r) + m - (r-1) + \sum_{q=2}^{r-1} S_q / s_{q,\min} \right) p_{\max}.$$

Proof. To construct the desired schedule S, we use heuristic H_1 with the algorithm $\mathcal{A}^*_r$ applied at Step 1. More exactly, since the vectors $\{d_j\}$ defined at Step 0 represent an $\hat{s}$-family in $\mathbb{R}^{r-1}$, we can apply the algorithm $\mathcal{A}^*_r$ for solving the NVS2$(r-1)$-problem

[†)] Such a use of the RLS-rule for the operations of the rth shop instead of the common LS-rule looks somewhat tricky. However, it has a natural explanation (see page 11). It could be added that a schedule with the same a.g. can be obtained if we apply the right order of jobs for distributing the operations in the first k shops, and apply the reverse order — in the last $(r-k)$ shops for any k, $1 \le k < r$.

with the input $X = \{d_1, \ldots, d_n\} \subset \mathbb{R}^{r-1}$. We next treat the permutation π found by $\mathcal{A}_r^*$ as a permutation of jobs. Let us now derive the desired a.g. for the solution constructed by H_1.

To each machine $M_{q,i}$ we assign a single index $i' \doteq \sum_{\nu=1}^{q-1} m_\nu + i$, and define vectors $\{t_j = (t_j(1), \ldots, t_j(m)) \mid j = 1, \ldots, n\}$, where $t_j(i)$ is the amount of time job J_j is processed on machine $M_i,\ i = 1, \ldots, m$, i.e., $t_j(i) = v_q^j / s_i$, if operation o_q^j is distributed to machine $M_i \in \mathcal{M}_q$, and $t_j(i) = 0$ otherwise. We also introduce notation for sums of k vectors: $T_k = \sum_{j=1}^k t_j,\ P_k = \sum_{j=1}^k p_j,\ D_k = \sum_{j=1}^k d_j,\ k = 1, \ldots, n$.

We first derive four useful relations (namely, (5), (7), (9), and (11)) between components of vectors T_k, T_{k-1}, P_k, and P_{k-1}. Due to the LS-rule, for every $k = 1, \ldots, n$, every $q = 1, \ldots, r$, and any two machines $M_{i'}, M_{i''} \in \mathcal{M}_q,\ i' \neq i''$, we have

$$T_k(i') \leq T_k(i'') + v_q^{k'} / s_{i''}, \tag{3}$$

where $o_q^{k'}$ is the operation on machine $M_{i'}$ scheduled last among the operations $\{o_q^1, \ldots, o_q^k\}$. (Indeed, due to the LS-rule, we have the relations

$$T_{k'}(i') = T_{k'-1}(i') + v_q^{k'} / s_{i'} \leq T_{k'-1}(i'') + v_q^{k'} / s_{i''} = T_{k'}(i'') + v_q^{k'} / s_{i''}$$

at step k' when the operation $o_q^{k'}$ is assigned to its machine. Later on, at steps $j = k'+1, \ldots, k$, the value of $T_j(i')$ does not change, whereas $T_j(i'')$ may only increase, keeping the inequality $T_j(i') \leq T_j(i'') + v_q^{k'} / s_{i''}$ valid.) Multiplying both sides of (3) by $s_{i''}$ and summing over all machines $M_{i''} \in \mathcal{M}_q$ (with an equation $T_k(i') = T_k(i'')$ for $i'' = i'$ instead of (3)), we come to the relations

$$T_k(i') S_q \leq \sum_{M_{i''} \in \mathcal{M}_q} T_k(i'') s_{i''} + (m_q - 1) v_q^{k'} = \sum_{j=1}^k v_q^j + (m_q - 1) v_q^{k'}, \tag{4}$$

from which we can derive for every $M_{i'} \in \mathcal{M}_q;\ q = 1, \ldots, r;\ k = 1, \ldots, n$:

$$T_k(i') \leq P_k(q) + (m_q - 1) p_{\max}. \tag{5}$$

Similarly, multiplying both sides of (3) by $s_{i'}$ and summing over all machines $M_{i'} \in \mathcal{M}_q$, we come to the relations

$$\sum_{M_{i'} \in \mathcal{M}_q} T_k(i') s_{i'} = \sum_{j=1}^k v_q^j \leq T_k(i'') S_q + \frac{1}{s_{i''}} \sum_{i' \neq i''} v_q^{k'} \cdot s_{i'} \tag{6}$$

from which we can derive for every $M_{i''} \in \mathcal{M}_q;\ q = 1, \ldots, r;\ k = 1, \ldots, n$:

$$-T_k(i'') \leq -\sum_{j=1}^k \frac{v_q^j}{S_q} + \frac{v_q^{k'}}{S_q} \cdot \frac{S_q - s_{i''}}{s_{i''}} \leq -P_k(q) + \left(\frac{S_q}{s_{i''}} - 1 \right) p_{\max}. \tag{7}$$

Furthermore, we have

$$P_k(q) S_q = \sum_{j=1}^{k-1} v_q^j + v_q^k = \sum_{M_{i'} \in \mathcal{M}_q} T_{k-1}(i') s_{i'} + v_q^k$$

$$\stackrel{\text{by (3)}}{\leq} \sum_{i' \neq i''} (T_{k-1}(i'') + v_q^{k'}/s_{i''}) s_{i'} + T_{k-1}(i'') s_{i''} + v_q^k$$

$$= T_{k-1}(i'') S_q + \frac{1}{s_{i''}} \sum_{i' \neq i''} v_q^{k'} s_{i'} + v_q^k. \tag{8}$$

Dividing the left- and the right-hand sides of (8) by S_q, we come to the relations $(\forall M_{i''} \in \mathcal{M}_q;\ q = 1, \ldots, r;\ k = 1, \ldots, n)$:

$$-T_{k-1}(i'') \leq -P_k(q) + \frac{p_{\max}}{s_{i''}} \sum_{i' \neq i''} s_{i'} + p_{\max} = -P_k(q) + \frac{S_q}{s_{i''}} p_{\max}. \tag{9}$$

And symmetrically,

$$P_{k-1}(q) S_q \quad = \quad \sum_{j=1}^{k} v_q^j - v_q^k = \sum_{M_{i''} \in \mathcal{M}_q} T_k(i'') s_{i''} - v_q^k$$

$$\stackrel{\text{by (3)}}{\geq} \quad \sum_{i''} T_k(i') s_{i''} - (m_q - 1) v_q^{k'} - v_q^k = T_k(i') S_q - (m_q - 1) v_q^{k'} - v_q^k. \tag{10}$$

Again, dividing the left and the right hand sides of (10) by S_q, we obtain $(\forall M_{i'} \in \mathcal{M}_q;\ q = 1, \ldots, r;\ k = 1, \ldots, n)$:

$$T_k(i') \leq P_{k-1}(q) + (m_q - 1) \frac{v_q^{k'}}{S_q} + \frac{v_q^k}{S_q} \leq P_{k-1}(q) + m_q p_{\max}. \tag{11}$$

Now we introduce notation for the complementary sums of k vectors $\{t_j\}$ and $\{p_j\}$:

$$\overline{T}_k \doteq \sum_{j=n-k+1}^{n} t_j = T_n - T_{n-k},\ \bar{P}_k \doteq \sum_{j=n-k+1}^{n} p_j = P_n - P_{n-k}. \tag{12}$$

It can be easily seen that, while distributing the operations of the rth shop to machines according to the RLS-rule, the quantities $\overline{T}_k(i)$ and $\bar{P}_k(q)$ meet relations similar to (5),(7),(9),(11). In particular, we obtain the following analogues for (5) and (11):

$$\overline{T}_k(i') \leq \bar{P}_k(r) + (m_r - 1) p_{\max}, \tag{13}$$

$$\overline{T}_k(i') \leq \bar{P}_{k-1}(r) + m_r p_{\max}. \tag{14}$$

Now, in order to derive the desired bound on the length of the final schedule, we construct a network $G(\pi)$ specifying the processing order on the set of all operations. Each operation o_q^j is represented in G by a node with weight v_q^j/s_i, where s_i denotes the speed of the machine $M_i \in \mathcal{M}_q$ to which the operation o_q^j is distributed. The operations distributed to machine M_i constitute the ith horizontal row of nodes, and the nodes of the ith row are connected by arcs in the order of their processing on machine M_i. (The total number of "horizontal" arcs in the network G is clearly $rn - m$.) To specify the

order of processing the operations of each job J_j, we introduce "vertical" arcs (o_q^j, o_{q+1}^j), each going from a horizontal row of set $\mathcal{M}_q$ to a horizontal row of set $\mathcal{M}_{q+1}$. There are $(r-1)n$ vertical arcs in G in total, which together with horizontal arcs makes up $O(rn)$ arcs in G. All arcs have zero weights.

As we know from project scheduling, the length of the "earliest schedule" S constructed at Step 4 of heuristic H_1 is equal to the length of the *critical path* in G (i.e., the path of maximum length). Each such path consists of $r-1$ vertical arcs corresponding to some jobs $k_1, k_2, \ldots, k_{r-1}$, $(1 \le k_1 \le k_2 \le \cdots \le k_{r-1} \le n)$, and of r horizontal chains corresponding to machines $M_{i_1}, M_{i_2}, \ldots, M_{i_r}$, where $M_{i_q} \in \mathcal{M}_q$. Therefore, the length of the critical path can be written as

$$C_{\max}(S) = \max_{\{k_q\},\{i_q\}} \left(\sum_{j=1}^{k_1} t_j(i_1) + \sum_{j=k_1}^{k_2} t_j(i_2) + \cdots + \sum_{j=k_{r-1}}^{n} t_j(i_r) \right). \tag{15}$$

We will further assume that $\{k_q\}$ and $\{i_q\}$ are just the values at which the maximum in the right hand side of equality (15) is attained, i.e.,

$$\begin{aligned} C_{\max}(S) &= \sum_{j=1}^{k_1} t_j(i_1) + \sum_{j=k_1}^{k_2} t_j(i_2) + \cdots + \sum_{j=k_{r-1}}^{n} t_j(i_r) \\ &= \sum_{q=1}^{r-1} \bigl(T_{k_q}(i_q) - T_{k_q-1}(i_{q+1}) \bigr) + T_n(i_r). \end{aligned} \tag{16}$$

Let f_q denote $T_{k_q}(i_q) - T_{k_q-1}(i_{q+1})$. We now estimate each quantity f_q, $q = 1, \ldots, r-2$, from above, and separately — the quantity

$$f_{r-1} + T_n(i_r) = T_{k_{r-1}}(i_{r-1}) + (T_n(i_r) - T_{k_{r-1}-1}(i_r)) = T_{k_{r-1}}(i_{r-1}) + \overline{T}_{n-k_{r-1}+1}(i_r).^{\ddagger)}$$

For $q = 1, \ldots, r-2$, we derive from (5) and (9) that

$$\begin{aligned} f_q &\le P_{k_q}(q) - P_{k_q}(q+1) + \bigl(m_q - 1 + S_{q+1}/s_{i_{q+1}} \bigr) p_{\max} \\ &= (D_{k_q}, e_q - e_{q+1}) + \bigl(m_q - 1 + S_{q+1}/s_{i_{q+1}} \bigr) p_{\max}. \end{aligned} \tag{17}$$

Another bound on f_q can be derived from (11) and (7):

$$\begin{aligned} f_q &\le P_{k_q-1}(q) - P_{k_q-1}(q+1) + \bigl(m_q + S_{q+1}/s_{i_{q+1}} - 1 \bigr) p_{\max} \\ &= (D_{k_q-1}, e_q - e_{q+1}) + \bigl(m_q - 1 + S_{q+1}/s_{i_{q+1}} \bigr) p_{\max}. \end{aligned} \tag{18}$$

$^{\ddagger)}$ Such a transformation of three partial sums $T_k(i)$ into two sums serves to decreasing the error that appears in the subsequent calculation, while replacing the amounts $T_k(i_q)$ by $P_k(q)$. On the other hand, it can be observed that, to obtain a good bound on a reverse partial sum $\overline{T}_k(i_r)$, it would be better if the operations in the rth shop were distributed to machines in the reverse order. That is why we do it at Step 3 (see page 8).

Upper bounds on $f_{r-1}+T_n(i_r)$ can also be obtained in two different ways. From (5) and (14) we derive

$$\begin{aligned} & f_{r-1}+T_n(i_r)=T_{k_{r-1}}(i_{r-1})+\overline{T}_{n-k_{r-1}+1}(i_r)\le P_{k_{r-1}}(r-1)+\bar{P}_{n-k_{r-1}}(r) \\ + \quad & (m_{r-1}-1+m_r)p_{\max}=P_n(r)+(P_{k_{r-1}}(r-1)-P_{k_{r-1}}(r))+(m_{r-1}+m_r-1)p_{\max} \\ = \quad & P_n(r)+D_{k_{r-1}}(r-1)+(m_{r-1}+m_r-1)p_{\max}, \end{aligned} \tag{19}$$

while application of (11) and (13) yields

$$\begin{aligned} f_{r-1} \quad + \quad & T_n(i_r)\le P_{k_{r-1}-1}(r-1)+\bar{P}_{n-k_{r-1}+1}(r)+(m_r-1+m_{r-1})p_{\max} \\ = \quad & P_n(r)+(P_{k_{r-1}-1}(r-1)-P_{k_{r-1}-1}(r))+(m_{r-1}+m_r-1)p_{\max} \\ = \quad & P_n(r)+D_{k_{r-1}-1}(r-1)+(m_{r-1}+m_r-1)p_{\max}. \end{aligned} \tag{20}$$

Choosing for every $f_q,\ q=1,\dots,r-2$, the best of its bounds (17), (18) and for $f_{r-1}+T_n(i_r)$ — the best of bounds (19), (20) and defining $e_r=0$, we obtain from (16) that

$$\begin{aligned} C_{\max}(S) \quad \le \quad & P_n(r)+\sum_{q=1}^{r-1}\min\{(D_{k_q-1},e_q-e_{q+1}),(D_{k_q},e_q-e_{q+1})\} \\ + \quad & \left(\sum_{q=1}^{r-2}\left(m_q-1+S_{q+1}/s_{i_{q+1}}\right)+(m_{r-1}+m_r-1)\right)p_{\max}. \end{aligned} \tag{21}$$

Since the permutation found by $\mathcal{A}_r^*$ provides nonstrict summation of vectors $\{d_j\}$ within a family of half-spaces $\mathcal{P}=\{P(e_1-e_2,\beta_1),\dots,P(e_{r-2}-e_{r-1},\beta_{r-2}),P(e_{r-1},\beta_{r-1})\}$ (with an a.g. $\sum\beta_i\le\theta^*(r)$), every $\min\{(D_{k_q-1},e_q-e_{q+1}),(D_{k_q},e_q-e_{q+1})\}$ in formula (21) can be bounded above by β_q (because at least one of the two successive partial sums D_{k_q-1} and D_{k_q} of vectors $\{d_j\}$ must be in the half-space $P(e_q-e_{q+1},\beta_q)$). This with (21) implies the bound

$$C_{\max}(S)-l_{\max}=C_{\max}(S)-P_n(r)\le\left(\theta^*(r)+\sum_{q=1}^{r}m_q-(r-1)+\sum_{q=2}^{r-1}\frac{S_q}{s_{q,\min}}\right)p_{\max}.$$

To complete the proof of Theorem 7, it remains to justify the bound on the running time. Indeed, Steps 0 and 4 of heuristic H_1 can be implemented in $O(nr)$ time, whereas Steps 2 and 3 require $O(nm)$ time. Finally, as assumed in Theorem 7, Step 1 can be implemented in $O(T(\mathcal{A}_r^*))$ time. Since it is natural to assume that $r\le m$, the total running time can bounded by $O(nm+T(\mathcal{A}_r^*))$. □

Note that we can use the theorem above with any algorithm solving the NVS2$(r-1)$ problem. To solve the NVS2$(r-1)$-problem for $r\ge 3$, we can use its reduction to the NVS1$(r-2)$-problem of summation of the $(r-2)$-dimensional projections $\{d'_j\}$ of vectors $\{d_j\}$ to the hyperplane $H=\{x\in\mathbb{R}^{r-1}\mid x(1)=0\}$ (see Lemma 1).

For solving the NVS1$(r-2)$-problem in general case ($r\ge 5$), we can use Corollary 4 which provides strict summation of the $\hat{s}$-family of vectors $\{d'_j\}$ within the ball of

radius $\left(r-3+\frac{1}{r-2}\right)$, and hence, yields a solution to the NVS1$(r-2)$-problem with each $\beta_i \le r-3+\frac{1}{r-2}$, $i=1,\ldots,r-1$, i.e., with an a.g. $\sum \beta_i \le (r-1)\left(r-3+\frac{1}{r-2}\right)$ in $O(n^2r^2)$ time, and due to Lemma 1, we have the same a.g. for the NVS2$(r-1)$-problem. In the case of $r=4$, due to Lemma 6, we solve the NVS1(2)-problem (and hence, the NVS2(3)-problem) with $\theta^*=3$ in $O(n\log n)$ time.

In the case of $r=3$, the one-dimensional NVS1(1)-problem and the two-dimensional NVS2(2)-problem are solvable in $O(n)$ time with $\theta^*=1$ (see Theorem 5).

Finally, in the case of $r=2$, there is no need for the reduction from the NVS2$(r-1)$-problem to the NVS1$(r-2)$-problem, because the first one is already one-dimensional. So, it can be solved in $O(n)$ time with an a.g. $\theta^*=0$. This provides a solution to the $F2(Q)||C_{\max}$ problem in $O(nm)$ time with an a.g.

$$C_{\max}(S)-l_{\max}\le (m-1)p_{\max}$$

independent of machine speeds.

The results obtained can be collected in the following

Theorem 8 *For the $F(Q)||C_{\max}$ problem with n jobs, r shops, and m machines, a series of approximation algorithms $\mathcal{A}_r$ can be developed in the framework of heuristic H_1 for different values of r. The parameters of these algorithms (namely, the a.g. and the running time) are specified in Table 3.* □

r	known a.g.*)	$T(\mathcal{A}_r)$	sourse	new a.g.*)	$T(\mathcal{A}_r)$
2				$m-1$	nm
3	$\left(4+\frac{1}{S_{\min}}\right)S_{\max}-2$	$nm+n\log n$	[8]	$m-1+\frac{S_2}{s_{2,\min}}$	nm
4				$m+\frac{S_2}{s_{2,\min}}+\frac{S_3}{s_{3,\min}}$	$nm+n\log n$
$r\ge 5$	$\gamma(r)+(2r-2)S_{\max}$	n^2r^2	[8]	$\gamma(r)+m+\sum_{q=2}^{r-1}\frac{S_q}{s_{q,\min}}$	n^2r^2

Table 3: Parameters of known and new algorithms $\mathcal{A}_r$ for $F(Q)||C_{\max}$;
*) $p_{\max}$ is scaled to one; $\gamma(r)=r^2-5r+5+1/(r-2)$

In the case of the ordinary flowshop problem, we have $S_q=1$ for every $q=1,\ldots,r$. Therefore, the quantity $p_{\max}$ is equal to $v_{\max}$, and all new results of Table 3 turn into the corresponding results of Table 1. Thus, Theorem 8 extends all the best known results for the flowshop problem to the case of the $F(Q)||C_{\max}$ problem.

We can apply our algorithm to the special case $F(P)||C_{max}$ by setting $s_{q,i}=1$, $S_q=m_q$, $\forall q,i$. Since steps 2 and 3 can be now implemented in $O(nr\log m_{\max})$ time (as opposed to $O(nm)$ in the general case), we have

Corollary 9 *The $F(P)||C_{\max}$ problem with n jobs and r shops can be solved by heuristic H_1 with an a.g. of the form*

$$C_{\max}(S)-l_{\max}\le \alpha p_{\max},$$

where the values of α and the bound on the running time $T(\mathcal{A}_r)$ for different r are specified in Table 4. □

r	new a.g.*)	$T(\mathcal{A}_r)$
2	$m-1$	$n \log m_{\max}$
3	$m+m_2-1$	$n \log m_{\max}$
4	$m+m_2+m_3$	$nm+n \log n$
$r \geq 5$	$\gamma(r)+m+\sum_{q=2}^{r-1} m_q$	$n^2 r^2$

Table 4: Parameters of heuristic H_1 for $F(P)||C_{\max}$;
*) $p_{\max}$ is scaled to one; $\gamma(r) = r^2 - 5r + 5 + 1/(r-2)$

In order to compare our a.g.s derived in terms of $p_{\max}$ units with a.g. (1) obtained by Kyparisis and Koulamas [8], we have to replace $v'_{\max}$ in (1) by the amount $S_{\max} p_{\max}$, which results in the bound

$$C_{\max}(S) - C^*_{\max} \leq \gamma(r) + (2r-2) S_{\max} \tag{22}$$

presented in Table 3. First of all, we make sure that for any values of the input parameters our a.g. in Table 3 is not worse than (22). Indeed, subtracting the amount $\gamma(r)$ (depending only on the number of shops) from both a.g.s, we can compare the remainders:

$$m + \sum_{q=2}^{r-1} \frac{S_q}{s_{q,\min}} \leq \sum_{q=1}^{r} m_q + \sum_{q=2}^{r-1} S_q \leq \sum_{q=1}^{r} S_q + \sum_{q=2}^{r-1} S_q \leq (2r-2) S_{\max}.$$

A similar relation can be derived for the case $r = 3$:

$$m - 1 + \frac{S_2}{s_{2,\min}} \leq \sum_{q=1}^{3} S_q + S_2 - 1 \leq 4 S_{\max} - 1 \leq \left(4 + \frac{1}{S_{\min}}\right) S_{\max} - 2.$$

In reality, a.g. (22) may be much worse in the case when $S_{\max}$ is much greater than the average m_q and the number of shops r.

To complete the comparison of the two methods, it remains to observe that our a.g. for the case of two shops does not depend on machines speeds at all, whereas applying the method from [8], one cannot avoid the using of the $S_{\max}$ amount.

A similar conclusion concerning the comparison of the known and the new results could be drawn for $F(P)||C_{\max}$. (To adapt the results from [8] to this particular case, it suffices to replace $S_{\max}$ by $m_{\max}$.) However, we cannot see much sense in doing this.

4 Heuristics H_2 and H_3

In this section, two heuristics will be described with a.g.s formulated in terms of $v_{\max}$ units. For convenience of calculation, we denote $v_{\max} = \boldsymbol{v}$, which implies that all volumes

of work v_q^j are in the interval $[0, \boldsymbol{v}]$. In fact, heuristics H_2 and H_3 (that will be applied in the cases $r \geq 3$ and $r = 2$, respectively) represent slight modifications of heuristic H_1 and only differ from the latter in Step 0. In heuristic H_2, we equate the values of l_q over all shops $q = 1, \ldots, r$, without changing the value of $v_{\max}$ (unlike heuristic H_1 in which $p_{\max}$ stayed invariable, while $v_{\max}$ might increase considerably), whereas in heuristic H_3 we do not perform Step 0 at all.

Heuristic H_2

Step 0 (adjusting the machine loads).

We first find the most loaded shop $\mathcal{M}_{q^*}$ with $l_{q^*} = l_{\max}$, and then for every $q \neq q^*$ such that $l_q < l_{\max}$ apply the following four procedures.

A-procedure. Increase the volumes of work v_q^j ($j \in \{1, \ldots, n\}$) up to at most $\boldsymbol{v}$ until we get either $l_q = l_{\max}$ or $v_q^j = \boldsymbol{v}$, $\forall\ j = 1, \ldots, n$.

If the latter holds, i.e., volumes of work of all operations of stage q received the maximum possible value but the average machine load in shop $\mathcal{M}_q$ is still less than $l_{\max}$ (which implies that $S_q > S_{q^*}$), then we continue with the following

B-procedure. Number the machines in shop $\mathcal{M}_q$ in nonincreasing order of their speeds: $s_{q,1} \geq s_{q,2} \geq \cdots \geq s_{q,m_q}$. If

$$\frac{n}{s_{q,1}}\ \boldsymbol{v} \geq l_{\max}, \tag{23}$$

remove the slowest machines $M_{q,m_q}, M_{q,m_q-1}, \ldots, M_{q,k+1}$ from shop $\mathcal{M}_q$, until for some $k > 1$ we get

$$\frac{n}{S_q^k}\ \boldsymbol{v} < l_{\max} \leq \frac{n}{S_q^{k-1}}\ \boldsymbol{v}, \tag{24}$$

where $S_q^k \doteq \sum_{i=1}^{k} s_{q,i}$. (Thus, no operations will be distributed to the removed machines at Steps 2 and 3 of our heuristic.) After this, continue with

C-procedure. Introduce additional (dummy) operations of the qth shop and assign them the maximum possible volumes of work until we get (2).

If (23) does not hold (which implies $s_{q,1} > S_{q^*}$), we apply

D-procedure. Remove all machines $\{M_{q,2}, \ldots, M_{q,m_q}\}$ from shop $\mathcal{M}_q$ and decrease the speed $s_{q,1}$ of the remaining machine $M_{q,1}$, until we get $l_q = l_{\max}$.

After all quantities l_q are equalized, we combine dummy operations of different shops into a minimal number of dummy jobs. (To that end, in each shop we choose an arbitrary dummy operation, if any, and the resulting family of operations take for a dummy job. Repeat this procedure until the set of dummy operations in every shop becomes empty.) This completes the description of Step 0. Steps 1–4 of heuristic H_2 are identical to those described in heuristic H_1. ∎

Observe that due to the removing of the slowest machines in the B-procedure, we have from (24) that

$$\frac{n}{S_q^k}\ \boldsymbol{v} < l_{\max} \leq \frac{2n}{S_q^k}\ \boldsymbol{v}.$$

This implies that to get (2) in the C-procedure, we need an introduction of at most n dummy operations of the qth shop. Therefore, we get at most n dummy jobs after the combining of the dummy operations of different shops, which does not affect any bound on the running time of an algorithm that we apply at subsequent steps of our heuristic.

Now let us begin with analysis of heuristic H_2. It follows from (23) that changing the value of S_q in the B- or D-procedure does not violate the relation $S_q \geq S_{q^*}$, and hence, does not change the value of $S_{\min}$.

For convenience of notation, we will assume that $n, m_q, m, s_{q,i}, S_q, l_q$, and v_q^j denote the new values of the corresponding quantities, for which (2) holds for every $q = 1, \ldots, r$. (The original jobs and the dummy ones will not be distinguished in subsequent steps.) In the same way we define the vectors $\{p_1, \ldots, p_n\} \subset \mathbb{R}^r$, $\{d_1, \ldots, d_n\} \subset \mathbb{R}^{r-1}$, $\{t_1, \ldots, t_n\} \subset \mathbb{R}^m$, and T_k, P_k for $k = 1, \ldots, n$. This time, $\{d_j\}$ does not represent an $\hat{s}$-family but is still a 0-family in $\mathbb{R}^{r-1}$.

Now we can derive relations similar to (5), (7), (11), but this time formulated in terms of the $v_{\max}$ units. To begin with, we can obtain from (4), (6), and (10) respectively:

$$T_k(i') \leq P_k(q) + \frac{m_q - 1}{S_q}\, \boldsymbol{v}, \tag{25}$$

$$-T_k(i'') \leq -P_k(q) + \left(\frac{1}{s_{i''}} - \frac{1}{S_q}\right) \boldsymbol{v}, \tag{26}$$

$$T_k(i') \leq P_{k-1}(q) + \frac{m_q}{S_q}\, \boldsymbol{v}, \tag{27}$$

which are valid for any $M_{i'}, M_{i''} \in \mathcal{M}_q;\ q = 1, \ldots, r;\ k = 1, \ldots, n$. Relations similar to (25) and (27) can be derived for the complementary quantities:

$$\overline{T}_k(i') \leq \bar{P}_k(r) + \frac{m_r - 1}{S_r}\, \boldsymbol{v}, \tag{28}$$

$$\overline{T}_k(i') \leq \bar{P}_{k-1}(r) + \frac{m_r}{S_r}\, \boldsymbol{v}. \tag{29}$$

From (27) and (26), we can derive

$$\begin{aligned} f_q &= T_{k_q}(i_q) - T_{k_q-1}(i_{q+1}) \leq P_{k_q-1}(q) + \frac{m_q}{S_q}\, \boldsymbol{v} - P_{k_q-1}(q+1) + \left(\frac{1}{s_{i_{q+1}}} - \frac{1}{S_{q+1}}\right) \boldsymbol{v} \\ &= (D_{k_q-1}, e_q - e_{q+1}) + \left(\frac{m_q}{S_q} + \frac{1}{s_{i_{q+1}}} - \frac{1}{S_{q+1}}\right) \boldsymbol{v}, \quad q = 1, \ldots, r-2. \end{aligned} \tag{30}$$

Using (25), (29), and (12), we derive

$$\begin{aligned} f_{r-1} + T_n(i_r) &= T_{k_{r-1}}(i_{r-1}) + \overline{T}_{n-k_{r-1}+1}(i_r) \\ &\leq P_{k_{r-1}}(r-1) + \bar{P}_{n-k_{r-1}}(r) + \left(\frac{m_{r-1} - 1}{S_{r-1}} + \frac{m_r}{S_r}\right) \boldsymbol{v} \\ &= D_{k_{r-1}}(r-1) + P_n(r) + \left(\frac{m_{r-1}}{S_{r-1}} + \frac{m_r}{S_r} - \frac{1}{S_{r-1}}\right) \boldsymbol{v}. \end{aligned} \tag{31}$$

Finally, from (27), (28), and (12), we derive

$$\begin{aligned} f_{r-1} + T_n(i_r) &= T_{k_{r-1}}(i_{r-1}) + \overline{T}_{n-k_{r-1}+1}(i_r) \\ &\leq P_{k_{r-1}-1}(r-1) + \frac{m_{r-1}}{S_{r-1}}\,\boldsymbol{v} + \bar{P}_{n-k_{r-1}+1}(r) + \frac{m_r - 1}{S_r}\,\boldsymbol{v} \\ &= D_{k_{r-1}-1}(r-1) + P_n(r) + \left(\frac{m_{r-1}}{S_{r-1}} + \frac{m_r}{S_r} - \frac{1}{S_r}\right)\boldsymbol{v}. \end{aligned} \tag{32}$$

Now using (16), (30), and (32) and defining $e_r = 0$, we obtain

$$\begin{aligned} C_{\max}(S) &\leq P_n(r) + \sum_{q=1}^{r-1}(D_{k_q-1}, e_q - e_{q+1}) + \left(\sum_{q=1}^{r}\frac{m_q}{S_q} + \sum_{q=2}^{r-1}\frac{1}{s_{i_q}} - \sum_{q=2}^{r}\frac{1}{S_q}\right)\boldsymbol{v} \\ &\leq l_{\max} + \theta^* + \left(r - 2 + \sum_{q=1}^{r}\frac{m_q}{S_q} - \sum_{q=2}^{r}\frac{1}{S_q}\right)\boldsymbol{v}, \end{aligned} \tag{33}$$

where θ^* is an a.g. that can be obtained for the VS2$(r-1)$-problem with the family of vectors $\{d_j\}$ as its input. (We also remind the reader that the inequality $s_i \geq 1$ is assumed to be held for each machine M_i, $i = 1, \ldots, m$, which implies $S_q \geq m_q$ for every $q = 1, \ldots, r$.) In the case of $r \geq 3$, due to Lemma 2, we can reduce the VS2$(r-1)$-problem to the VS1$(r-2)$-problem of strict summation of the $(r-2)$-dimensional projections $\{d'_j\}$ of vectors $\{d_j\}$ to the hyperplane $H = \{x \in \mathbb{R}^{r-1} \mid x(1) = 0\}$. For each vector d'_j and each its coordinate $i = 2, \ldots, r-1$ we have relations

$$-\boldsymbol{v}/S_r \leq -p_j(r) \leq d'_j(i) = p_j(i) - p_j(r) \leq p_j(i) \leq \boldsymbol{v}/S_i,$$

which imply that vector d'_j belongs to the half-spaces $P(-e_i, \boldsymbol{v}/S_r)$ and $P(e_i, \boldsymbol{v}/S_i)$ for every $i = 2, \ldots, r-1$. From the definition of $d'_j(i)$ we can also derive relations

$$-\boldsymbol{v}/S_{i+1} \leq d'_j(i) - d'_j(i+1) = p_j(i) - p_j(i+1) \leq \boldsymbol{v}/S_i,$$

which imply that vector d'_j belongs to the half-spaces $P(e_{i+1} - e_i, \boldsymbol{v}/S_{i+1})$ and $P(e_i - e_{i+1}, \boldsymbol{v}/S_i)$ for every $i = 2, \ldots, r-2$. In particular, all vectors $\{d'_1, \ldots, d'_n\}$ are in the intersection $\hat{P}$ of the half-spaces

$$\{P(e_2 - e_3, \boldsymbol{v}/S_2), P(e_3 - e_2, \boldsymbol{v}/S_3), \ldots, P(e_{r-2} - e_{r-1}, \boldsymbol{v}/S_{r-2}), P(e_{r-1} - e_{r-2}, \boldsymbol{v}/S_{r-1}),$$

$$P(-e_2, \boldsymbol{v}/S_r), P(e_2, \boldsymbol{v}/S_2), P(e_{r-1}, \boldsymbol{v}/S_{r-1}), P(-e_{r-1}, \boldsymbol{v}/S_r)\}.$$

Applying Theorem 3 with vector $a = 0$, we can find a permutation $\pi = (\pi_1, \ldots, \pi_n)$ that provides summation of vectors $\{d'_j\}$ in the set $(r-3)\hat{P} + P_a$, where $P_a = -\frac{1}{r-2}\hat{P}$ is the intersection of half-spaces

$$\{P(e_2 - e_3, \boldsymbol{v}/S_3), P(e_3 - e_2, \boldsymbol{v}/S_2), \ldots, P(e_{r-2} - e_{r-1}, \boldsymbol{v}/S_{r-1}), P(e_{r-1} - e_{r-2}, \boldsymbol{v}/S_{r-2}),$$

$$P(-e_2, \boldsymbol{v}/S_2), P(e_2, \boldsymbol{v}/S_r), P(e_{r-1}, \boldsymbol{v}/S_r), P(-e_{r-1}, \boldsymbol{v}/S_{r-1})\}$$

multiplied by $\frac{1}{r-2}$. Hence, all partial sums $\sum_{j=1}^{k} d'_{\pi_j}$ are contained in the set

$$\begin{aligned}(r-3)\hat{P}+P_a &\subseteq P\left(e_2-e_3,\frac{r-3}{S_2}\,\boldsymbol{v}+\frac{1}{(r-2)S_3}\,\boldsymbol{v}\right)\cap\\ \cdots &\cap P\left(e_{r-2}-e_{r-1},\frac{r-3}{S_{r-2}}\,\boldsymbol{v}+\frac{1}{(r-2)S_{r-1}}\,\boldsymbol{v}\right)\\ &\cap P\left(e_{r-1},\frac{r-3}{S_{r-1}}\,\boldsymbol{v}+\frac{1}{(r-2)S_r}\,\boldsymbol{v}\right)\cap P\left(-e_2,\frac{r-3}{S_r}\,\boldsymbol{v}+\frac{1}{(r-2)S_2}\,\boldsymbol{v}\right),\end{aligned}$$

and therefore, permutation π and the family b of numbers

$$\beta_2=\left(\frac{r-3}{S_2}+\frac{1}{(r-2)S_3}\right)\boldsymbol{v},\ldots,\ \beta_{r-2}=\left(\frac{r-3}{S_{r-2}}+\frac{1}{(r-2)S_{r-1}}\right)\boldsymbol{v},$$

$$\beta_{r-1}=\left(\frac{r-3}{S_{r-1}}+\frac{1}{(r-2)S_r}\right)\boldsymbol{v},\ \beta_r=\left(\frac{r-3}{S_r}+\frac{1}{(r-2)S_2}\right)\boldsymbol{v}$$

provide a solution for the VS1$(r-2)$-problem with the input family of vectors $\{d'_1,\ldots,d'_n\}$ and the value of the objective function:

$$\theta_1(b)=\sum_{q=2}^{r}\beta_q=\left(r-3+\frac{1}{r-2}\right)\sum_{q=2}^{r}\frac{\boldsymbol{v}}{S_q}\doteq\theta^*.$$

Due to Lemma 2, the same a.g. $\theta_2(b)\le\theta^*$ is valid for the corresponding solution of the VS2$(r-1)$-problem with the input family of vectors $\{d_1,\ldots,d_n\}$, which together with (33) provides a possibility of constructing a schedule S with an a.g.

$$C_{\max}(S)-l_{\max}\le\left(r-4+\frac{1}{r-2}\right)\sum_{q=2}^{r}\frac{\boldsymbol{v}}{S_q}+\left(r-2+\sum_{q=1}^{r}\frac{m_q}{S_q}\right)\boldsymbol{v}. \tag{34}$$

Since the 1-dimensional problem VS1(1) (we refer to in the case of $r=3$) can be solved with the same a.g. in linear time, we have a linear-time procedure for solving the $F3(Q)||C_{\max}$ problem with a.g. (34).

In the case of $r=2$, we apply heuristic H_3 which is exactly heuristic H_1 without Step 0. Since we do not try to equate the values of l_q over all q, the family of numbers $\{d_1,\ldots,d_n\}$ is not a 0-family. However, a problem of summation of the numbers $d_j=p_j(1)-p_j(2)$ $(j\in\{1,\ldots,n\})$ can be solved at Step 1 of heuristic H_3 in linear time so as to guarantee $D_k\le(P_n(1)-P_n(2))^+$, $\forall k=1,\ldots,n$. Choosing the best bound on $f_1+T_n(i_2)$ from among (31) and (32), we obtain

$$\begin{aligned}C_{\max}(S) &= f_1+T_n(i_2)\le(P_n(1)-P_n(2))^+ +P_n(2)+\left(\frac{m_1}{S_1}+\frac{m_2}{S_2}-\max\left\{\frac{1}{S_1},\frac{1}{S_2}\right\}\right)\boldsymbol{v}\\ &= l_{\max}+\left(\frac{m_1}{S_1}+\frac{m_2}{S_2}-\frac{1}{S_{\min}}\right)\boldsymbol{v},\end{aligned}$$

from which we can conclude that heuristic H_3 finds a solution to the $F2(Q)||C_{\max}$ problem with an a.g.

$$C_{\max}(S) - l_{\max} \leq \left(\frac{m_1}{S_1} + \frac{m_2}{S_2} - \frac{1}{S_{\min}} \right) \boldsymbol{v} \tag{35}$$

in $O(nm)$ time.

Since we omit Step 0, the quantities $\{m_i\}$ and $\{S_i\}$ used in formula (35) have their original values.

Summarizing our results, we can formulate

Theorem 10 *The $F(Q)||C_{\max}$ problem with n jobs and $r \geq 4$ shops can be solved in $O(n^2r^2)$ time with an a.g.*

$$C_{\max}(S) - l_{\max} \leq \left(\left(r - 4 + \frac{1}{r-2} \right) \sum_{q=2}^{r} \frac{1}{S_q} + r - 2 + \sum_{q=1}^{r} \frac{m_q}{S_q} \right) v_{\max},$$

where m_q and S_q are the total number and the total speed of machines in the qth shop.

The $F3(Q)||C_{\max}$ problem can be solved in $O(nm)$ time with an a.g.

$$C_{\max}(S) - l_{\max} \leq \left(1 + \sum_{q=1}^{3} \frac{m_q}{S_q} \right) v_{\max}.$$

The $F2(Q)||C_{\max}$ problem can be solved in $O(nm)$ time with an a.g.

$$C_{\max}(S) - l_{\max} \leq \left(\frac{m_1}{S_1} + \frac{m_2}{S_2} - \frac{1}{S_{\min}} \right) v_{\max}. \qquad \square$$

Recalling that the inequalities $s_i \geq 1$ are assumed to be valid for every machine M_i, we derive from Theorem 10 the following

Corollary 11 *The $F(Q)||C_{\max}$ problem with n jobs and $r \geq 4$ shops can be solved in $O(n^2r^2)$ time with an a.g.*

$$C_{\max}(S) - l_{\max} \leq (r-1) \left(r - 2 + \frac{1}{r-2} \right) v_{\max}.$$

The $F3(Q)||C_{\max}$ problem can be solved in $O(nm)$ time with an a.g.

$$C_{\max}(S) - l_{\max} \leq 4 v_{\max}.$$

The $F2(Q)||C_{\max}$ problem can be solved in $O(nm)$ time with an a.g.

$$C_{\max}(S) - l_{\max} \leq \left(2 - \frac{1}{S_{\min}} \right) v_{\max}. \qquad \square$$

Comparing the a.g.s presented in Corollary 11 with those collected in Table 1, we can see that while the corresponding a.g.s are identical for the case of $r \geq 5$, they differ for the cases of $r = 3$ and $r = 4$. Namely, heuristic H_2, being applied to the special case of $Fr||C_{\max}$ problem with $r = 3$ or $r = 4$, will provide somewhat worse a.g.s in comparison with the approximation algorithms presented in Table 1. It should be noted that those two algorithms use the advantage of application of the nonstrict vector summation to $(r-2)$-dimensional vectors, whereas we have not managed to perform a similar dimension reduction in the case of arbitrary machine speeds. Maybe, there are prospects here for improving the a.g.s presented in Theorem 10 in the case of 3 and 4 shops.

On the other hand, it is clear that the most considerable prospects are hidden in improving the technique of strict vector summation, which would enable one to improve the results of Theorem 3, and hence, the a.g.s presented in Theorems 8 and 10 for the case of arbitrary number of shops.

Acknowledgement. I'd like to express my much gratitude to the referees whose constructive suggestions contributed much to improving the paper.

References

[1] Chen, B. Scheduling Multiprocessor Flow Shops. In: D.-Z. Du and J. Sun (eds.). New Advances in Optimization and Approximation. Kluwer, 1994, 1–8.

[2] Garey, M.R., D.S. Johnson, and R. Sethi. The Complexity of Flowshop and Jobshop Scheduling, *Math. Oper. Res.* **1** (1976) 117–129.

[3] Hall, L.A. Approximability of flow shop scheduling, in: *Proceedings of 36th IEEE Symposium on Foundations of Computer Science*, 1995, 82–91.

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

[5] Hoogeveen, J.A., J.K. Lenstra, and B. Veltman. Preemptive Scheduling in a Two-Stage Multiprocessor Flow Shop is NP-Hard, *Eur.J.Oper.Res.* **89** (1996) 172–175.

[6] Johnson, S.M. Optimal Two and Three-Stage Production Schedules with Set-up Times Included, *Nav. Res. Log. Quart.* **1** (1954) 61–68.

[7] Kyparisis, G.J. and C. Koulamas. Asymptotically Optimal Linear Time Algorithms for Two-Stage and Three-Stage Flexible Flowshops, *INFORMS J. Computing*, submitted.

[8] Kyparisis, G.J. and C. Koulamas. Flexible Flowshop Scheduling with Uniform Parallel Machines, *Math. Oper. Res.*, submitted.

[9] Lee, C.-Y. and G.L. Vairaktarakis. Minimizing Makespan in Hybrid Flowshops, *Oper. Res. Letters* **16** (1994) 149–158.

[10] Schuurman, P. and G.J. Woeginger. A Polynomial Time Approximation Scheme for the Two-Stage Multiprocessor Flow Shop Problem, *Report Woe-01, March 1997, Combinatorial Approximation Algorithms*, TU Graz, Austria, 1997.

[11] Sevastianov, S.V. On a Compact Vector Summation. (In Russian), *Diskretnaya Matematika* **3** (1991), no.3, 66–72.

[12] Sevastianov, S.V. Construction of an approximate schedule for a flow-type system. (In Russian), *Upravlyaemye Sistemy* **31** (1993) 66–71.

[13] Sevastianov, S.V. Vector summation in Banach space and polynomial algorithms for flow shops and open shops, *Math.Oper.Res.* **20** (1995) 90–103.

[14] Sevastianov, S.V. Nonstrict vector summation in the plane and its application to scheduling problems, in: *A.D. Korshunov (ed.), Operations Research and Discrete Analysis*, Kluwer, Dordrecht, 1997, 241–272.