Problem $F_2||C_{\text{max}}$ with Forbidden Jobs in the First or Last Position is Easy

Mikhail Y. Kovalyov$^a$, Frank Werner$^b$, 1

$^a$ Faculty of Economics, Belarus State University, and United Institute of Informatics Problems, National Academy of Sciences of Belarus, Skorini 4, 220050 Minsk, Belarus, E-mail: koval@newman.bas-net.by

$^b$ Faculty of Mathematics, Otto-von-Guericke-University Magdeburg, PSF 4120, 39016 Magdeburg, Germany, E-mail: frank.werner@mathematik.uni-magdeburg.de

Abstract

Saadani et al. [3] studied the classical flow shop scheduling problem $F_2||C_{\text{max}}$ with an additional constraint that some jobs cannot be placed in the first or last position. There exists an optimal job sequence for this problem, in which at most one job in the first or last position is deferred from its position in Johnson’s [1] permutation. The problem was solved in $O(n^2)$ time by enumerating all candidate job sequences. We suggest a simple $O(n)$ algorithm for this problem provided that Johnson’s permutation is given. Since Johnson’s permutation can be obtained in $O(n \log n)$ time, the problem in [3] can be solved in $O(n \log n)$ time as well.

Key words: scheduling, flow shop, makespan.

Saadani et al. [3] studied the following extension of problem $F_2||C_{\text{max}}$. There are $n$ jobs to be processed in a two-machine flow shop. The processing of each job $j$ consists of two operations: on machine 1 within $a_j$ time units and on machine 2 within $b_j$ time units. For the same job, the operation on machine 2 can start only after the operation on machine 1 is completed. In addition, there is a set $X$ of jobs forbidden for processing in the first position or a set of jobs $Y$ forbidden for processing in the last position. The objective is to find a schedule minimizing the makespan, $C_{\text{max}}$.

It is easy to see that like in the classical problem $F_2||C_{\text{max}}$ a search for an optimal schedule

1Corresponding author
can be limited to permutation schedules, in which the jobs are processed in the same sequence on both machines.

A solution to this problem was used by Baptiste and Hguny [2] in a branch and bound algorithm for problem \( F|no-idle|C_{\text{max}} \), where jobs are to be processed in an \( m \)-machine flow shop with no machine idle time between the operations on the same machine.

The problem with jobs forbidden in the last position is a mirror image of the problem with jobs forbidden in the first position. Therefore, we shall consider only one of these problems, namely, with jobs forbidden in the first position.

Saadani et al. [3] correctly stated that there exists an optimal job sequence, in which all jobs appear in Johnson’s [1] order, except the first job. They suggested an evident solution procedure, which is to evaluate all \( n - |X| \) candidate job sequences. They stated that evaluation of a sequence takes \( O(n) \) time and therefore their solution procedure requires \( O(n^2) \) time, which is not satisfactory for its repeated usage in the branch and bound algorithm.

Let \((i_1, \ldots, i_n)\) be a Johnson’s sequence, i.e. a job sequence in accordance with Johnson’s rule. It can be constructed in \( O(n \log n) \) time, see Johnson [1]. A candidate for an optimal job sequence is \((i_k, i_1, \ldots, i_{k-1}, i_{k+1}, \ldots, i_n)\), where \( i_k \notin X \). Let \( C_{\text{max}}^{(k)} \) denote the makespan of this sequence.

We shall demonstrate that all the values \( C_{\text{max}}^{(k)} \), \( k = 2, \ldots, n \), can be calculated in \( O(n) \) time, subject to the assumption that Johnson’s sequence is given. Then the problem with jobs forbidden in the first or last position can be solved in \( O(n \log n) \) time.

Calculate

\[
A^u_1 = \sum_{j=1}^{u} a_{ij}, \quad B^n_u = \sum_{j=u}^{n} b_{ij} \quad \text{and} \quad AB_u = A^u_1 + B^n_u, \quad u = 1, \ldots, n.
\]

Further, calculate

\[
C_{\text{max}}^{(k)} = \max \{ a_{ik} + B^n_1, \max_{1 \leq u \leq k-1} \{ a_{ik} + AB_u - b_{ik} \}, \max_{k+1 \leq u \leq n} \{ AB_u \} \}
\]

\[
= \max \{ a_{ik} + B^n_1, a_{ik} - b_{ik} + MAX_{1,k-1}, MAX_{k+1,n} \}, \quad k = 2, \ldots, n,
\]

where

\[
MAX_{1,k-1} = \max_{1 \leq u \leq k-1} \{ AB_u \}, \quad k = 2, \ldots, n,
\]

\[
MAX_{k+1,n} = \max_{k+1 \leq u \leq n} \{ AB_u \}, \quad k = 1, \ldots, n-1, \quad MAX_{n+1,n} = 0.
\]

Since values \( A^u_1, B^n_u \) and \( AB_u, u = 1, \ldots, n \), are partial sums of the given sequences of numbers, all these values together can iteratively be computed in \( O(n) \) time. Similarly, since
values $MAX_{1,k-1}$, $k = 2, \ldots, n$, and $MAX_{k+1,n}$, $k = 1, \ldots, n - 1$, are partial maximums of the given sequences of numbers, they all together can be computed in $O(n)$ time. Finally, each makespan value $C_{\text{max}}^{(k)}$, $k \in \{2, \ldots, n\}$, is the maximum of three given values. Therefore, all the values $C_{\text{max}}^{(k)}$, $k = 2, \ldots, n$, can be computed in $O(n)$ time, as we claimed.

We deduce that the time complexity of solving problem $F2||C_{\text{max}}$ with a set of jobs forbidden in the first or last position is determined by the time of obtaining Johnson’s sequence, which is $O(n \log n)$.

**Acknowledgments**

This research was supported in part by INTAS under grant number 03-51-5501.

**References**

