BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 67, No. 1, 2019
DOI: 10.24425/bpas.2019.127335

Better polynomial algorithms for scheduling unit-length jobs with bipartite incompatibility graphs on uniform machines

T. PIKIES* and M. KUBALE

Department of Algorithms and System Modelling, Gdańsk University of Technology, ETI Faculty,
Gabriela Narutowicza 11/12, 80-233 Gdańsk, Poland

Abstract. The goal of this paper is to explore and to provide tools for the investigation of the problems of unit-length scheduling of incompatible jobs on uniform machines. We present two new algorithms that are a significant improvement over the known algorithms. The first one is Algorithm 2 which is 2-approximate for the problem $Qm|p_j = 1, G = bisubquartic|C_{\max}$. The second one is Algorithm 3 which is 4-approximate for the problem $Qm|p_j = 1, G = bisubquartic|\Sigma C_j$, where $m \in \{2, 3, 4\}$. The theory behind the proposed algorithms is based on the properties of 2-coloring with maximal coloring width, and on the properties of ideal machine, an abstract machine that we introduce in this paper.

Key words: approximation algorithm, graph coloring, incompatible job, polynomial algorithm, scheduling, uniform machine, unit-time job.

1. Introduction

Suppose we have to process n bins of chemical substances and we have m parallel uniform machines, i.e. machines that have identical functionality but may have different processing speeds. On no machine can we process two substances that may react with each other in order to guarantee machines safety or due to spacial reasons. The aim is such an assignment of the bins to the machines that the processing time of all bins is as short as possible or the mean flow time of a bin is as short as possible. We assume here that each of the bins has to be processed on exactly one machine without interruptions.

The problem can be expressed as the following scheduling problem. Let us assume that we have n identical jobs with unit execution times. We also have m parallel uniform machines. They are uniform in the sense that the execution of a job on a machine takes time inversely proportional to the speed of the machine to be completed. There may be incompatibilities between some jobs, that is, jobs belonging to incompatible pairs cannot be scheduled on the same machine. We consider two criteria of optimality: the length of schedule and the total completion time.

Let us recall definitions of graph theory used further in this paper. For a graph $G = (V, E)$ by $n(G) = |V|$ we denote the number of its vertices (nodes) and by $e(G) = |E|$ the number of its edges. We denote by $\Delta(G)$ the maximum degree of a vertex in G. By $V_0(G)$ we mean the set of all vertices of a graph G with degree 0. A graph G is said to be a *cubic* (*quartic*) graph if it is 3-regular (4-regular). If, however, G fulfills $\Delta(G) \leq 3$ $(\Delta(G) \leq 4)$ then it is called *subcubic* (*subquartic*). A bipartite cubic (quartic) graph is said to be *bicubic* (*biquartic*), and a bipartite subcubic (subquartic) graph is said to be *bisubcubic* (*bisubquartic*). The symbol $\alpha(G)$ stands for the size of a maximum independent set in G. We say that a graph is *k-colorable* if its vertices can be partitioned into k independent sets, called *color classes*, and by a *k-coloring* of G we mean such a partition of its vertices. The *width* of a coloring is the difference between the sizes of the largest and the smallest color class of the coloring. For other definitions the reader is referred to [10]. Conflicts between jobs can be modeled by an *incompatibility graph*, i.e. by a graph which has exactly one unique vertex assigned to each of the jobs and which has an edge between a pair of vertices if the corresponding jobs cannot be processed on the same machine. We say that jobs in a given set are *compatible* if the vertices corresponding to them form an independent set in the incompatibility graph.

Now let us introduce some definitions concerning scheduling theory. Let us denote the set of jobs by $\mathcal{J} = \{j_1, \ldots, j_n\}$. Let us also denote the set of the machines by $\mathcal{M} = \{M_1, \ldots, M_m\}$. By $s(M)$ we denote the speed of machine M. Without loss of generality we assume that $s(M_1) \geq \ldots \geq s(M_m)$, i.e. that the machines are ordered according to their speeds. By $s_{total} = \sum_{M \in \mathcal{M}} s(M)$ we denote the sum of the speeds of all machines. A *schedule* is for each job an allocation of a time interval to a machine [2]. A *subschedule* is a schedule for a restricted set of jobs $\mathcal{J}' \subseteq \mathcal{J}$ on a restricted set of machines $\mathcal{M}' \subseteq \mathcal{M}$. By $n_S(M)$ we denote the number of jobs scheduled on machine M in a schedule S. By $C_S(M)$ we denote the latest completion time of the jobs scheduled on machine M in a schedule S. We have $C_S(M) = n_S(M)/s(M)$. Thus the *schedule length* is equivalent to $\max C_S(\mathcal{M})$. Symbol $\Sigma_S(N)$ stands for the total completion time of jobs from a set $N \subseteq \mathcal{J}$ in a schedule S, therefore the *total completion time* is denoted by $\Sigma_S(\mathcal{J})$. By a *greedy assignment* of compatible jobs to machines we understand an assignment in which the jobs are scheduled one by one and each job is assigned to the machine that guarantees the shortest completion time at the moment of scheduling this job. In the

*e-mail: tytpikie@student.pg.edu.pl, kubale@eti.pg.edu.pl

Manuscript submitted 2018-01-04, revised 2018-04-16, initially accepted for publication 2018-05-08, published in February 2019.

following algorithms all the assignments of jobs to machines should be performed greedily, e.g. using an Algorithm given in [3].

For a set of the machines $K \subseteq \mathscr{M}$ by an *ideal machine* M_{id} we mean a machine with the speed equal to $\sum_{M \in K} s(M)$, that can ignore incompatibilities between jobs. By $C_{id}(N)$ we denote the latest completion time of the jobs from set $N \subseteq \mathscr{J}$ scheduled on the ideal machine M_{id}, hence $C_{id}(N) = |N| / \sum_{M \in K} s(M)$. By $\Sigma_{id}(N)$ we denote the total completion time of jobs from a set $N \subseteq \mathscr{J}$ scheduled on the ideal machine M_{id}, hence $\Sigma_{id}(N) = 0.5|N|(|N|+1)/\sum_{M \in K} s(M)$.

There are several papers addressing the problem of scheduling with incompatible job constraints on identical machines. However, to the best of our knowledge, there are few results involving uniform machines. The problem is hard even for identical machines. The authors of [1] have proved that even if there are only three identical machines and the incompatibilities between jobs can be modeled by a bipartite graph then the problem $P3|p_j = 1, G = bipartite|C_{\max}$ remains NP-hard.

There is a review of known algorithms for unit-length scheduling with incompatible jobs presented in [4]. The authors of [4] have also considered a model of more powerful machines that are called *parallel batching* machines (for details of this model see e.g. [2]). They presented optimal polynomial algorithms for the problems $Q3|batch, p_j = 1, G = bicubic|\Sigma C_j$ and $Q4|batch, p_j = 1, G = biquartic|\Sigma C_j$. Based on [7], they also presented optimal polynomial algorithms for the problem $Q3|batch, p_j = 1, G = bipartite|\Sigma C_j$ when $s(M_1) = s(M_2) \geq s(M_3)$ and for the problem $Q3|batch, p_j = 1, G = bisubquartic|\Sigma C_j$ when $s(M_1) = 2s(M_2) = 3s(M_3)$. An optimal Algorithm for the problem $Q3|p_j = 1, G = connected\ bicubic|C_{\max}$ with time complexity $O(n^2)$ was presented in [6]. Also in that paper there has been given a 19/7-approximation Algorithm for the NP-hard problem $Q3|p_j = 1, G = connected\ 3\text{-}colorable\ cubic|C_{\max}$ when the machines have speeds $s(M_1) > s(M_2) = s(M_3)$. This Algorithm has time complexity $O(n^3)$. There is a $O(n^2)$ Algorithm for optimal scheduling in the $Q4|p_j = 1, G = bisubcubic|C_{\max}$ problem if the machines have speeds $s(M_1) \geq 12s(M_2) = 12s(M_3) = 12s(M_4)$ [5]. The authors have also discussed an extension of this algorithm, that is a $O(mn^{1.5})$ Algorithm for the problem $Qm|p_j = 1, G = bipartite, \Delta(G) \leq m|C_{\max}$ and machines with speeds $s(M_1) \geq m(m-1)s(M_m), s(M_2) = \ldots = s(M_m)$.

In this paper we analyze the problem of the scheduling of incompatible unit-length jobs with respect to two criteria of optimality. The first criterion is the schedule length minimization. The second one is the total completion time minimization. The remainder of the paper is organized as follows. In the next section we describe a procedure for 2-coloring of G with maximum coloring width. In Section 3 we give a 2-approximation Algorithm for the problem $Qm|p_j = 1, G = bisubquartic|C_{\max}$. This Algorithm is better than Algorithm 4 in [5] in the sense that it has a smaller approximation ratio, it runs in $O(n)$ time rather than $O(n^{1.5})$ time and requires no assumptions as to the machine speeds. In Section 4 we give a 4-approximation Algorithm for the problem $Qm|p_j = 1, G = bisubquartic|\Sigma C_j$, where $m \in \{2, 3, 4\}$.

2. 2-coloring with maximal coloring width

We present a simple Algorithm that constructs a 2-coloring of a bipartite graph with maximal coloring width.

Algorithm 1 Non-Equitable Coloring

Input: A bipartite graph G.
Output: A two-coloring with maximal coloring width

1. $Col_1 = Col_2 = \emptyset$
2. Let $G_1, \ldots, G_c$ be connected components of G.
3. **for** $i = 1$ **to** c **do**
4. Split $V(G_i)$ into color classes $Col_1(G_i)$ and $Col_2(G_i)$, where $|Col_1(G_i)| \geq |Col_2(G_i)|$.
5. $Col_1 = Col_1 \cup Col_1(G_i)$.
6. $Col_2 = Col_2 \cup Col_2(G_i)$.
7. **end for**
8. **return** (Col_1, Col_2)

Lemma 1. Let G be a bisubquartic graph. For the set Col_2 returned by Algorithm 1 we have $|Col_2| \leq 2(n(G) - \alpha(G))$.

Proof. The set of vertices of G can be split into $V_0(G)$ and $V(G) \setminus V_0(G)$. Since no isolated vertex belongs to Col_2, we will consider the set $V(G) \setminus V_0(G)$ only.

Let H be a graph with at least one edge. The following inequality is well known in graph theory (see e.g. [11]).

$$\alpha(H) \leq \left\lfloor n(H) - \frac{e(H)}{\Delta(H)} \right\rfloor.$$

Let us notice that for a connected component G_i we have

$$\alpha(G_i) \leq \left\lfloor \frac{3}{4} n(G_i) + \frac{1}{4} \right\rfloor.$$

because G_i is connected and subquartic. Now we show how the number of vertices in $Col_2(G_i)$ is related to $n(G_i) - \alpha(G_i)$. For this reason, let $n(G_i) = 4l + r$, where $l \in N$, $r \in \{0, 1, 2, 3\}$. Consider four cases, as follows.

- If $r = 0$, we have $\alpha(G_i) \leq 3l$. From this we obtain $n(G_i) - \alpha(G_i) \geq l$. Thus $|Col_2(G_i)| \leq 2l \leq 2(n(G_i) - \alpha(G_i))$.
- If $r = 1$, we have $\alpha(G_i) \leq 3l + 1$. From this we have $n(G_i) - \alpha(G_i) \geq l$, and also we obtain $|Col_2(G_i)| \leq 2l \leq 2(n(G_i) - \alpha(G_i))$.
- If $r = 2$, we have $\alpha(G_i) \leq 3l + 1$. From this we have $n(G_i) - \alpha(G_i) \geq l + 1$, so in this case $|Col_2(G_i)| \leq 2l + 1 \leq 2(n(G_i) - \alpha(G_i))$.
- Finally, let us assume that $r = 3$. Then we have $\alpha(G_i) \leq 3l + 2$. Like in the previous case we have $n(G_i) - \alpha(G_i) \geq l + 1$, and similarly we conclude that $|Col_2(G_i)| \leq 2l + 1 \leq 2(n(G_i) - \alpha(G_i))$.

So for each connected component G_i, $Col_2(G_i)$ contains no more than $2(n(G_i) - \alpha(G_i))$ vertices. From these observations it follows that $|Col_2| \leq 2(n(G) - \alpha(G))$. □

3. 2-approximation Algorithm for the problem $Qm|p_j = 1, G = bisubquartic|C_{\max}$

Observation 1. For a set $K \subseteq \mathscr{M}$ and a set $N \subseteq \mathscr{J}$ and the ideal machine M_{id} for K, $C_{id}(N)$ is a lower bound on the length of any subschedule for N on K.

Lemma 2. Let $K \subseteq \mathscr{M}$ and let $N \subseteq \mathscr{J}$ be a set of compatible unit-time jobs. Let M_{id} be the ideal machine for K and S_{opt} be a subschedule for N on K with the minimal length. Then

$$\frac{\max C_{S_{opt}}(K)}{C_{id}(N)} \leq 1 + \frac{|K|-1}{|N|}.$$

Proof. Without loss of generality we may assume that S_{opt} was created greedily, i.e. by the application of the Algorithm given in [3]. The completion time on M_{id} is equal to

$$C_{id}(N) = \max C_{S_{opt}}(K) - t, \tag{1}$$

for some $t \geq 0$. Let $M_i \in K$ be any machine such that $C_{S_{opt}}(M_i) = \max C_{S_{opt}}(K)$. Let $n(M)$ for $M \in K$ means the maximal number of jobs that M can execute in time equal to $C_{id}(N)$. Then

$$n(M_j) \leq n_{S_{opt}}(M_j) + 1 - s(M_j)t$$

for any machine $M_{j\neq i} \in K$. The above bound results from the following observation. Each $M_{j\neq i} \in K$ can complete at most $n_{S_{opt}}(M_j) + 1$ jobs in time equal to $\max C_{S_{opt}}(K)$. Otherwise, it would provide a faster completion of the last job on M_i than M_i does. But this is not possible due to the Algorithm for schedule construction. M_i can complete $n(M_i) = n_{S_{opt}}(M_i) - s(M_i)t$ jobs in time equal to $C_{id}(N)$. So we have

$$\sum_{M\in K} n(M) = |N|,$$
$$\sum_{M\in K\setminus\{M_i\}} n(M) + n_{S_{opt}}(M_i) - s(M_i)t = |N|,$$
$$\sum_{M\in K\setminus\{M_i\}} (n_{S_{opt}}(M) + 1 - s(M)t) + n_{S_{opt}}(M_i) - s(M_i)t \geq |N|.$$

Applying the equality

$$\sum_{M\in K} n_{S_{opt}}(M) = |N|,$$

we obtain

$$\sum_{M\in K\setminus\{M_i\}} (1 - s(M)t) - s(M_i)t \geq 0,$$
$$|K| - 1 \geq t \sum_{M\in K} s(M),$$

so

$$t \leq \frac{|K|-1}{\sum_{M\in K} s(M)}.$$

Substituting the above inequality to (1) we get

$$\max C_{S_{opt}}(K) \leq C_{id}(N) + \frac{|K|-1}{\sum_{M\in K} s(M)}.$$

It follows immediately that

$$\frac{\max C_{S_{opt}}(K)}{C_{id}(N)} \leq \frac{C_{id}(N) + \frac{|K|-1}{\sum_{M\in K} s(M)}}{C_{id}(N)} = 1 + \frac{|K|-1}{|N|}. \qquad \square$$

Observation 2. Let G be an incompatibility graph and let S be a subschedule for the unit-time jobs corresponding to $V(G) - V_0(G)$ on $\mathscr{M}$. Let S be $k \geq 1$ times longer than a subschedule for $V(G) - V_0(G)$ on $\mathscr{M}$ with the minimal length and let a schedule S' be an extension of S created by the greedy assignment of the jobs corresponding to $V_0(G)$. Then the length of S' is at most k times greater than the minimal schedule length for G.

Therefore we assume $V_0(G) = \emptyset$ for the problem under consideration.

Corollary 1. Let S be a subschedule created by a greedy assignment of $N \subseteq \mathscr{J}$ to $K \subseteq \mathscr{M}$. If a number k is such that

$$k\frac{\sum_{M\in K} s(M)}{s_{total}} > \frac{|N|}{|\mathscr{J}|} \ \textit{ and } \ |N| \geq \frac{|K|-1}{k\frac{\sum_{M\in K} s(M)}{s_{total}} - \frac{|N|}{|\mathscr{J}|}}, \tag{2}$$

or

$$|K| = 1 \ \textit{ and } \ k\frac{\sum_{M\in K} s(M)}{s_{total}} = \frac{|N|}{|\mathscr{J}|}, \tag{3}$$

then

$$\max C_S(K) \leq kC_{id}(\mathscr{J}),$$

where M_{id} is the ideal machine for $\mathscr{M}$.

Proof. Let us assume that (2) occurs. Let $M_{id'}$ be the ideal machine for K. By Lemma 2, we have

$$\frac{\max C_S(K)}{C_{id'}(N)} \leq 1 + \frac{|K|-1}{|N|}.$$

Applying the equality

$$\frac{C_{id'}(N)}{C_{id}(\mathscr{J})} = \frac{|N|}{|\mathscr{J}|}\frac{s_{total}}{\sum_{M\in K} s(M)}.$$

and substituting the bound on $|N|$ into the above inequality, we immediately obtain the thesis of Corollary 1.

Now let us assume that (3) occurs, then we may simply compare the $\max C_S(K)$ and $C_{id}(\mathscr{J})$ and the thesis follows immediately. $\square$

In the rest of the paper we use this corollary with $k = 2$.

Algorithm 2 A 2-approximation algorithm for the problem $Qm|p_j = 1, G = bisubquartic|C_{\max}$

Input: A bisubquartic graph G.
Output: A schedule

1. $(Col_1, Col_2) = Non\text{-}Equitable\ Coloring(G)$.
2. **case**
3. **:** $s(M_1) \in [\frac{2}{5}s_{total}, s_{total})$ **:**
4. $M_1 \leftarrow Col_1,\ M_2, \ldots, M_m \leftarrow Col_2$.
5. **:** $s(M_1) \in [\frac{1}{4}s_{total}, \frac{2}{5}s_{total})$ **:**
6. $M_1 \leftarrow Col_2,\ M_2, \ldots, M_m \leftarrow Col_1$.
7. **:** $s(M_1) \in [\frac{1}{m}s_{total}, \frac{1}{4}s_{total})$ **:**
8. Let k be the smallest integer such that $K = \{M_1, \ldots, M_k\}$ and $\sum_{M \in K} s(M) \geq \frac{9}{20}s_{total}$.
9. **if** $n < 10(m-2)$ **then**
10. Find an optimal schedule by a brute-force algorithm.
11. **else**
12. $M_1, \ldots M_k \leftarrow Col_1,\ M_{k+1}, \ldots, M_m \leftarrow Col_2$.
13. **end if**
14. **end case**

Theorem 1. Algorithm 2 is 2-approximate for the problem $Qm|p_j = 1, G = bisubquartic|C_{\max}$.

Proof. Let S be a schedule produced by the algorithm. Let S_{opt} be a schedule with the minimal length. If $s(M_1) \in [\frac{2}{5}s_{total}, s_{total})$ then we may estimate $C_S(M_1)$ using Corollary 1. By Lemma 1, $\max C_S(\{M_2, \ldots, M_m\}) \leq 2\max C_{S_{opt}}(\mathscr{M})$ since at most twice the minimal number of jobs is scheduled on $M_2, \ldots, M_m$.

If $s(M_1) \in [\frac{1}{4}s_{total}, \frac{2}{5}s_{total})$ then we may estimate $C_S(M_1)$ using Corollary 1, because $\frac{1}{2}n \geq |Col_2|$. If the number of jobs scheduled on $M_2, \ldots, M_m$ in S_{opt} is greater than $\frac{2}{5}n$, then $\max C_S(\{M_2, \ldots, M_m\}) \leq 2\max C_{S_{opt}}(\mathscr{M})$, because we schedule on $M_2, \ldots, M_m$ at most twice the number of jobs assigned to them in S_{opt}, since $\frac{4}{5}n \geq \alpha(G) \geq |Col_1|$ [5]. If the number of jobs scheduled on $M_2, \ldots, M_m$ in S_{opt} is less than or equal to $\frac{2}{5}n$, then $\max C_{S_{opt}}(\mathscr{M}) \geq C_{S_{opt}}(M_1) \geq \frac{3}{2}\frac{n}{s_{total}}$. Let us bound the ratio of $\max C_S(\{M_2, \ldots, M_m\})$ to $\max C_{S_{opt}}(\mathscr{M})$ in this case. Let $M_{id'}$ be the ideal machine for $\{M_2, \ldots, M_m\}$. By Lemma 2, we have

$$\frac{\max C_S(\{M_2, \ldots, M_m\})}{C_{id'}(Col_1)} \leq 1 + \frac{m-2}{|Col_1|}.$$

We also have

$$\frac{C_{id'}(Col_1)}{\max C_{S_{opt}}(\mathscr{M})} \leq \frac{10}{9}\frac{|Col_1|}{n}.$$

By combining both inequalities we obtain,

$$\frac{\max C_S(\{M_2, \ldots, M_m\})}{\max C_{S_{opt}}(\mathscr{M})} \leq \frac{10}{9}\left(\frac{|Col_1|}{n} + \frac{m-2}{n}\right).$$

Without loss of generality we may assume that $n \geq m$. By this assumption and by the fact that $\frac{4}{5}n \geq \alpha(G) \geq |Col_1|$ we obtain

$$\frac{\max C_S(\{M_2, \ldots, M_m\})}{\max C_{S_{opt}}(\mathscr{M})} \leq 2.$$

For the last case we may bound the ratio of $\max C_S(K)$ to $C_{id}(\mathscr{J})$ and that of $\max C_S(\mathscr{M} \setminus K)$ to $C_{id}(\mathscr{J})$, where M_{id} is the ideal machine for $\mathscr{M}$. By the way k is established, we have $\frac{14}{20}s_{total} > \sum_{M \in K} s(M) \geq \frac{9}{20}s_{total}$ and $\frac{11}{20}s_{total} \geq \sum_{M \in \mathscr{M} \setminus K} s(M) > \frac{6}{20}s_{total}$. These inequalities are obtained using the condition that $\frac{1}{4}s_{total} > \max s(\mathscr{M})$. The value $\frac{9}{20}s_{total}$ that is present in the inequality determining k is a conveniently chosen constant that asserts that $2\left(\sum_{M \in K} s(M)\right) > \frac{4}{5}s_{total}$ and that $2\left(\sum_{M \in \mathscr{M} \setminus K} s(M)\right) > \frac{1}{2}s_{total}$. By Corollary 1, if $n \geq 10(k-1)$, then the ratio of $\max C_S(K)$ to $C_{id}(\mathscr{J})$ is at most 2. Similarly, if $n \geq 10(m-k-1)$, then the ratio of $\max C_S(\mathscr{M} \setminus K)$ to $C_{id}(\mathscr{J})$ is at most 2. If $n \geq 10(m-2)$, then all the ratios are less than or equal to 2. □

The approximation coefficient 2 is the best possible value in the sense that for jobs with the conflict graph in the shape of three double-stars $S_{3,3}$, presented in Fig. 1a, the schedule constructed by the Algorithm has the length exactly twice the optimal (cf. Fig. 1b, 1c).

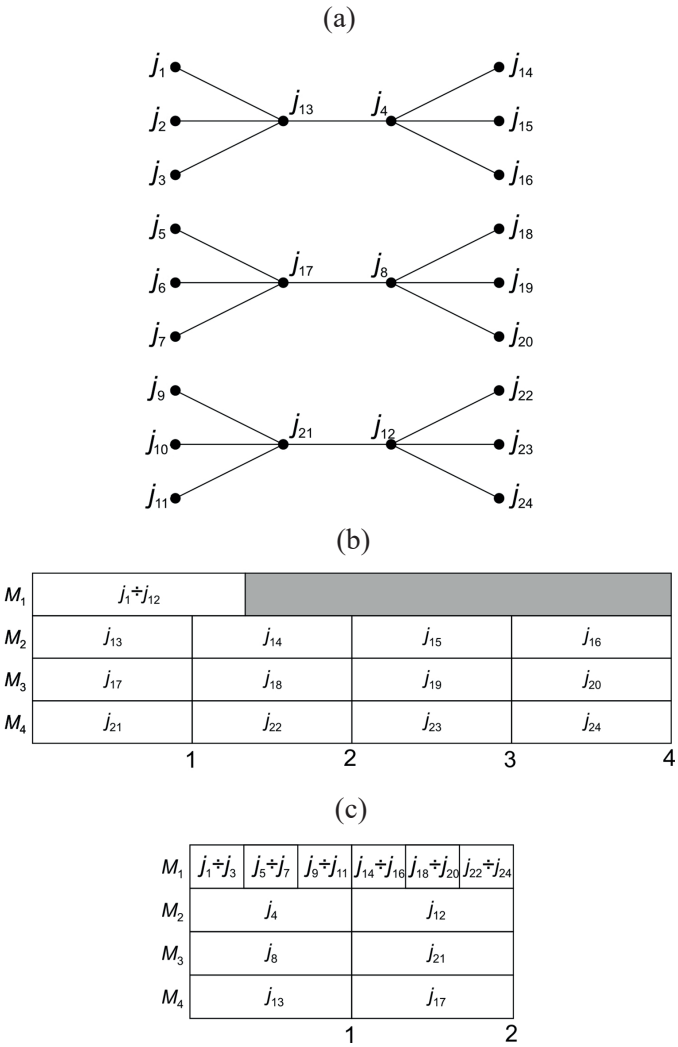

Fig. 1. The worst-case example for Algorithm 2: a) a graph $3S_{3,3}$ and machines with speeds $s(M_1) = 9$, $s(M_2) = s(M_3) = s(M_4) = 1$; b) Gantt chart for the suboptimal schedule; c) Gantt chart for an optimal schedule

4. 4-approximation Algorithm for the problem $Qm|p_j=1, G=bisubquartic|\Sigma C_j$, where $m \in \{2, 3, 4\}$

We begin this section with the following.

Lemma 3. Let $K \subseteq \mathscr{M}$ and $N \subseteq \mathscr{J}$. $\Sigma_{id}(N)$ is a lower bound on the total completion time of any subschedule for N on K, where M_{id} is the ideal machine for K.

Proof. We prove the lemma for a 2-element set $K = \{M_i, M_j\}$. Let S be any subschedule for N on K and N_i, N_j be the sets of the jobs assigned to the corresponding machines in S. We show that $\Sigma_{id}(N) \leq \Sigma_S(N)$. We first transform the inequality

$$(s(M_i)|N_j| - s(M_j)|N_i|)^2 \geq 0$$

to the following form

$$2s(M_i)s(M_j)|N_i||N_j| \leq s(M_i)^2|N_j|^2 + s(M_j)^2|N_i|^2 \leq$$
$$\leq s(M_i)^2|N_j|(|N_j|+1) + s(M_j)^2|N_i|(|N_i|+1).$$

By adding $s(M_i)s(M_j)(|N_i|^2 + |N_j|^2 + |N_i| + |N_j|)$ to both sides and by a simple transformation we obtain

$$s(M_i)s(M_j)(|N_i|+|N_j|)(|N_i|+|N_j|+1) \leq$$
$$\leq s(M_i)+s(M_j))(s(M_j)|N_i|(|N_i|+1) + s(M_i)|N_j|(|N_j|+1)).$$

Dividing both the sides by $2s(M_i)s(M_j)\big(s(M_i)+s(M_j)\big)$ we get

$$\Sigma_{id}(N) = \frac{(|N_i|+|N_j|)(|N_i|+|N_j|+1)}{2(s(M_i)+s(M_j))}$$
$$\leq \frac{s(M_j)|N_i|(|N_i|+1) + s(M_i)|N_j|(|N_j|+1)}{2s(M_i)s(M_j)}$$
$$\leq \Sigma_S(N).$$

If the number of machines is greater than 2, the thesis follows from induction on $|K|$. □

Algorithm 3 A 4-approximation Algorithm for the problem $Qm|p_j=1, G=bisubquartic|\Sigma C_j$, where $m \in \{2, 3, 4\}$

Input: A bisubquartic graph G.

Output: A schedule

1. $(Col_1, Col_2) = Non\text{-}Equitable\ Coloring(G)$.
2. $M_1 \leftarrow Col_1,\ M_2, \ldots, M_m \leftarrow Col_2$.

Theorem 2. Algorithm 3 is 4-approximate for the problem $Qm|p_j=1, G=bisubquartic|\Sigma C_j$, where $m \in \{2, 3, 4\}$.

Proof. Let us begin our analysis with the case of a problem with 4 machines. Let S be a schedule obtained by Algorithm 3 and let S_{opt} be any schedule with the minimal total completion time. We may divide $\mathscr{J}$ into sets N_{M_1}, N_{M_2}, N_{M_3} and N_{M_4}, where N_{M_i} is the set of the jobs assigned to M_i in S_{opt}.

- If $|Col_2| \geq |N_{M_2} \cup N_{M_3} \cup N_{M_4}|$, then by Lemma 1, $|Col_2| \leq 2|N_{M_2} \cup N_{M_3} \cup N_{M_4}|$, hence

$$\Sigma_S(Col_2) < 4\Sigma_{S_{opt}}\big(N_{M_2} \cup N_{M_3} \cup N_{M_4}\big). \tag{4}$$

In this case $|Col_1| \leq |N_{M_1}|$, so we have

$$\Sigma_S(Col_1) \leq \Sigma_{S_{opt}}\big(N_{M_1}\big) \leq 4\Sigma_{S_{opt}}\big(N_{M_1}\big).$$

Therefore

$$\Sigma_S(\mathscr{J}) = \Sigma_S(Col_1) + \Sigma_S(Col_2)$$
$$\leq 4\Sigma_{S_{opt}}(N_{M_1}) + 4\Sigma_{S_{opt}}(N_{M_2} \cup N_{M_3} \cup N_{M_4})$$
$$= 4\Sigma_{S_{opt}}(\mathscr{J}).$$

Inequality (4) comes from the following observation. Let us assume that $|Col_2| \leq 2|N_{M_2} \cup N_{M_3} \cup N_{M_4}|$. The total completion time in a greedy assignment is not greater than in the assignment in which we schedule exactly $2|N_{M_2}|$ jobs on M_2, $2|N_{M_3}|$ jobs on M_3 and $2|N_{M_4}|$ jobs on M_4. For each of the machines, in this assignment the total completion time of the jobs assigned to a machine is less than four times the total completion time of the jobs assigned to this machine in S_{opt}. Thus the inequality follows.

- Assume $|Col_2| < |N_{M_2} \cup N_{M_3} \cup N_{M_4}|$. Let us divide the set $N_{M_2} \cup N_{M_3} \cup N_{M_4}$ into 2 subsets. Let $N_{|Col_2|}$ be any of its subsets with cardinality $|Col_2|$. Let $N_r = \big(N_{M_2} \cup N_{M_3} \cup N_{M_4}\big) \setminus N_{|Col_2|}$. Then we have $|Col_1| = |N_{M_1}| + |N_r|$ and

$$\Sigma_{S_{opt}}\big(\mathscr{J}\big) = \Sigma_{S_{opt}}\big(N_{M_1}\big) + \Sigma_{S_{opt}}\big(N_r\big) + \Sigma_{S_{opt}}\big(N_{|Col_2|}\big).$$

Let M_{id} be the ideal machine for $\mathscr{M}$. By the inequality

$$\frac{\Sigma_S(Col_1)}{\Sigma_{id}(Col_1)} = \frac{\frac{|Col_1|(|Col_1|+1)}{2s(M_1)}}{\frac{|Col_1|(|Col_1|+1)}{2\sum_{M\in\mathscr{M}} s(M)}} \leq 4.$$

and by Lemma 3, the total completion time of Col_1 on M_1 is at most 4 times greater than the the total completion time of $|Col_1|$ jobs in any other schedule. Hence

$$\Sigma_S(Col_1) \leq 4\big(\Sigma_{S_{opt}}\big(N_{M_1}\big) + \Sigma_{S_{opt}}\big(N_r\big)\big).$$

Thus we obtain

$$\Sigma_S(\mathscr{J}) = \Sigma_S(Col_1) + \Sigma_S(Col_2)$$
$$\leq \Sigma_S(Col_1) + 4\Sigma_S(Col_2)$$
$$\leq 4\Sigma_{S_{opt}}(N_{M_1}) + 4\Sigma_{S_{opt}}(N_r) + 4\Sigma_S(Col_2)$$
$$\leq 4\Sigma_{S_{opt}}(N_{M_1}) + 4\Sigma_{S_{opt}}(N_r) + 4\Sigma_{S_{opt}}(N_{|Col_2|})$$
$$= 4\Sigma_{S_{opt}}(\mathscr{J}).$$

In both cases we have $\Sigma_S\big(\mathscr{J}\big) \leq 4\Sigma_{S_{opt}}\big(\mathscr{J}\big)$, hence Algorithm 3 is 4-approximate. Similar observations may be used to establish that the Algorithm is 4-approximate for the problems $Q2|p_j=1, G=bisubquartic|\Sigma C_j$ and $Q3|p_j=1, G=bisubquartic|\Sigma C_j$. □

The approximation coefficient 4 is the best possible value in the sense that for any $\varepsilon > 0$ we may find such integers l and k that for a graph G consisting of l copies of $3S_{3,3}$, the graph presented in Fig. 1a, and for machines with speeds $s(M_1) = ks(M_4)$ and $s(M_2) = s(M_3) = s(M_4)$ the Algorithm constructs a schedule S with $\Sigma_S(\mathscr{J})$ that is at least $4 - \varepsilon$ times greater than the minimal one. Let $l \geq (10 - 4\varepsilon)/(6\varepsilon)$ and let $k \geq 180l$. Then we have

$$\frac{10}{6l+4} \leq \varepsilon .$$

The optimal assignment is to assign the jobs corresponding to an independent set of vertices with cardinality $\alpha(G) = 18l$ to the fastest machine and to assign the remaining jobs, with cardinality equal to $6l$, to the other machines in a balanced way. Let us denote this schedule as S_{opt}. Then

$$\begin{aligned}\Sigma_{S_{opt}}(\mathscr{J}) &= \frac{18l(18l+1)}{2ks(M_4)} + 3\frac{2l(2l+1)}{2s(M_4)} \\ &\leq \frac{180l^2 + (6l^2+3l)k}{ks(M_4)} \leq \frac{6l^2+4l}{s(M_4)}.\end{aligned}$$

Algorithm 3 constructs a schedule with $12l$ jobs assigned to M_1 and $12l$ jobs assigned to M_2, M_3, M_4. So we have

$$\Sigma_S(\mathscr{J}) = \frac{12l(12l+1)}{2ks(M_4)} + 3\frac{4l(4l+1)}{2s(M_4)} \geq \frac{24l^2+6l}{s(M_4)}.$$

Hence we get

$$4 - \varepsilon \leq 4 - \frac{10}{6l+4} = \frac{24l^2+6l}{6l^2+4l} \leq \frac{\Sigma_S(\mathscr{J})}{\Sigma_{S_{opt}}(\mathscr{J})}.$$

Observation 3. Algorithms 1 and 3 run in $O(n)$ time for bisubquartic graphs. Algorithm 2 has the overall complexity of $O\left(nm^{10(m-2)}\right)$ for bisubquartic graphs.

In fact, the algorithms are devoted to sparse graphs so it is reasonable to implement them using a data structure suitable for such graphs.

Algorithm 1 consists of 2 phases, both of the complexity $O(n)$. Algorithm 3 consists of two parts. The first is the application of Algorithm 1. The second is the assignment of the jobs corresponding to color classes to the machines which can be implemented in linear time [3]. So the total complexity of this Algorithm is $O(n)$.

Algorithm 2 may have to find an optimal solution for small graphs. This can be done by using an exhaustive search. In general, each of the vertices can be colored using one of m colors. The correctness of a coloring can be verified in $O(n)$ time. The length of the schedule corresponding to a coloring can be calculated in $O(m)$ time. Thus the total complexity is at most $O\left((n+m)m^n\right)$. However, in our case, the brute-force algorithm is used only for graphs with $n \leq 10(m-2)$. Thus it has time complexity on the order of $O\left((n+m)m^{10(m-2)}\right)$. The remaining operations are the same as in the previous cases. However, the greedy assignment can be done in time $O(n \log m)$. Hence the total time complexity is $O\left(n + n\log m + (n+m)m^{10(m-2)}\right) = O\left(nm^{10(m-2)}\right)$, which is $O(n)$ for a fixed number of machines.

3. Final remarks

Our analysis is based on 2-coloring of a bipartite graph with maximal width. A natural extension of this approach is to consider 2-colorings with smaller widths. Such colorings seems to be better, e.g. when the machines can be split into two sets with closer sums of speeds.

It is also interesting to know if Algorithm 2 can be refined to obtain polynomial time complexity in both the number of jobs and the number of machines.

Finally, it would be interesting to incorporate in our scheduling model time window constraints imposed on the machines (cf. [8]). In this direction of study the model of graph coloring with forbidden colors would be useful [9].

Acknowledgements. The project has been partially supported by Gdańsk University of Technology, grant no. POWR.03.02.00-IP.08-00-DOK/16.

References

[1] H.L. Bodlaender and K. Jansen, "On the complexity of scheduling incompatible jobs with unit-times", *Lecture Notes in Computer Science* 711, (1993).

[2] P. Brucker, Scheduling Algorithms, Fifth edition, Springer-Verlag Berlin Heidelberg, 2007.

[3] M.I. Dessouky, B.J. Lageweg, J.K. Lenstra, and S.L. van de Velde, "Scheduling of identical jobs on uniform parallel machines", *Statistica Neerlandica* 44, 115–123 (1990).

[4] S. Duraj, P. Kopeć, M. Kubale, T. and Pikies, "Scheduling of identical jobs with bipartite incompatibility graphs on uniform machines. Computational experiments", *Decision Making in Manufacturing and Services* 11, 53–61 (2017).

[5] H. Furmańczyk and M. Kubale, "Scheduling of unit-length jobs with bipartite incompatibility graphs of four uniform machines", *Bull. Pol. Ac.: Tech.* 65, 29–34 (2017).

[6] H. Furmańczyk and M. Kubale, "Scheduling of unit-length jobs with cubic incompatibility graphs on three uniform machines", *Discrete Applied Mathematics* 234, 210–217 (2018).

[7] K. Giaro, R. Janczewski, M. Kubale, and M. Małafiejski, "A 27/26-approximation algorithm for the chromatic sum coloring of bipartite graphs", *Lecture Notes in Computer Science* 2462, 135–145 (2002).

[8] K. Giaro, M. Kubale, and P. Obszarski, "A graph coloring approach to scheduling of multiprocessor tasks on dedicated machines with availability constraints", *Discrete Applied Mathematics* 157, 3625–3630 (2009).

[9] M. Kubale, "Interval vertex-coloring of a graph with forbidden colors", *Discrete Mathematics* 74, 125–136 (1989).

[10] D.B. West, Introduction to Graph Theory, Pearson Education, Delhi India, 2002.

[11] W. Willis, "Bounds for the independence number of a graph", Virginia Commonwealth University, 2011.