Scheduling of unit-length jobs with cubic incompatibility graphs on three uniform machines

Hanna Furmańczyk a,*, Marek Kubale b

a Institute of Informatics, University of Gdańsk, Wita Stwosza 57, 80-952 Gdańsk, Poland
b Faculty of Electronics, Telecommunications and Informatics, Gdańsk University of Technology, Narutowicza 11/12, 80-233 Gdańsk, Poland

A B S T R A C T

In the paper we consider the problem of scheduling $n$ identical jobs on 3 uniform machines with speeds $s_1$, $s_2$, and $s_3$ to minimize the schedule length. We assume that jobs are subjected to some kind of mutual exclusion constraints, modeled by a cubic incompatibility graph. We show that if the graph is 2-chromatic then the problem can be solved in $O(n^2)$ time. If the graph is 3-chromatic, the problem becomes NP-hard even if $s_1 > s_2 = s_3$. However, in this case there exists a $10/7$-approximation algorithm running in $O(n^3)$ time. Moreover, this algorithm solves the problem almost surely to optimality if $3s_1/4 \leq s_2 = s_3$.

© 2016 Elsevier B.V. All rights reserved.

1. Introduction

Imagine you have to arrange a dinner for, say 30, people and you have at your disposal 3 round tables with different numbers of seats (not greater than 15). You know that each of your guests is in bad relations with exactly 3 other people. Your task is to assign the people to the tables in such a way that no two of them being in bad relations seat at the same table. In the paper we show how to solve this and related problems.

Our problem can be expressed as the following scheduling problem. Suppose we have $n$ identical jobs $j_1, \ldots, j_n$, so we assume that they all have unit execution times, in symbols $p_i = 1$, to be processed on three non-identical machines $M_1$, $M_2$, and $M_3$. These machines run at different speeds $s_1$, $s_2$, and $s_3$, respectively. However, they are uniform in the sense that if a job is executed on machine $M_i$, it takes $1/s_i$ time units to be completed. It refers to the situation where the machines are of different generations, e.g. old and slow, new and fast, etc.

Our scheduling model would be trivial if all the jobs were compatible. Therefore we assume that some pairs of jobs cannot be processed on the same machine due to some technological constraints. More precisely, we assume that each job is in conflict with exactly three other jobs. Thus the underlying incompatibility graph $G$ whose vertices are jobs and edges correspond to pairs of jobs being in conflict is cubic. For example, all graphs in our figures are cubic. The number of jobs $n$ must be even, since the sum of degrees of all vertices in $G$, i.e. $3n$, must be even. A load $l_i$ on machine $M_i$ requires the

* This project has been partially supported by Narodowe Centrum Nauki under contract DEC-2011/02/A/ST6/00201.
* Corresponding author.

E-mail addresses: hanna@inf.ug.edu.pl (H. Furmańczyk), kubale@eti.pg.gda.pl (M. Kubale).

http://dx.doi.org/10.1016/j.dam.2016.01.036
0166-218X/© 2016 Elsevier B.V. All rights reserved.
processing time \( P(l_i) = \frac{|l_i|}{s_i} \), and all the jobs are ready for processing at the same time. By definition, each load forms an independent set (color) in \( G \). Therefore, in what follows we will be using the terms job/vertex and color/independent set interchangeably. Since all tasks have to be executed, the problem is to find a 3-coloring, i.e. a decomposition of \( G \) into 3 independent sets \( l_1, l_2, \) and \( l_3 \) such that the schedule length \( C_{\text{max}} = \max\{P(l_i) : i = 1, 2, 3\} \) is minimized, in symbols \( Q3|p| = 1, G = \text{cubic} \). We assume \( G \) for any cubic graph. Moreover, from (1) it follows that for any cubic graph \( K \), the only cubic graph for which the chromatic number is equal to 4 is \( K_4 \). The rest of this paper is split into two parts depending on the chromaticity of cubic graphs. In Section 2 we consider 2-chromatic graphs. In particular, we give an \( O(n^2) \)-time algorithm for optimal scheduling of such graphs. Section 3 is devoted to 3-chromatic graphs. In particular, we give an \( \text{NP-hardness} \) proof and an approximation algorithm with good performance guarantee. Our algorithm runs in \( O(n^2) \) time to produce a solution of value less than 10/7 times optimal, provided that \( s_1 > s_2 = s_3 \). Moreover, this algorithm solves the problem almost surely to optimality if \( 3s_1/4 \leq s_2 = s_3 \). Finally, we discuss possible extensions of our model to arbitrary job lengths, to disconnected graphs, and to more than three machines.

2. Scheduling of 2-chromatic graphs

We begin with introducing some basic notions concerning graph coloring. A graph \( G = (V, E) \) is said to be \( k \)-colorable if and only if its vertex set can be partitioned into independent sets \( V_1, \ldots, V_k \subset V \) such that \( |V_i| - |V_j| \leq 1 \) for all \( i, j = 1, \ldots, k \). The smallest \( k \) for which \( G \) admits such a coloring is called the \( \chi \) (G) is the chromatic number of \( G \) and denoted \( \chi(G) \). A graph \( G \) has a \( k \)-colorable \( k \)-coloring if there exists a partition of its vertices into independent sets \( V_1, \ldots, V_k \subset V \) such that one of these subsets, say \( V_i \), is of size \( |V_i| \leq \left\lceil |V| / k \right\rceil \), and the remaining subgraph \( G - V_i \) is \( (k - 1) \)-colorable. In the following we will say that graph \( G \) has \( (V_1, \ldots, V_k) \)-coloring to express explicitly a partition of \( V \) into \( k \) independent sets. If, however, only cardinalities of color classes are important, we will use the notation \( \left\lceil |V| / k \right\rceil \).

Let us recall some basic facts concerning colorability of cubic graphs. It is well known from Brooks theorem [3] that for any cubic graph \( G \neq K_4 \) we have \( \chi(G) \leq 3 \), where \( \chi(G) \) is the classical chromatic number of \( G \) and \( K_4 \) is the complete graph on four vertices. On the other hand, Chen et al. [4] proved that every 3-chromatic cubic graph can be equitably colored without introducing a new color. Moreover, since a connected cubic graph \( G \) with \( \chi(G) = 2 \) is a bipartite graph with partition sets of equal size, we have the equivalence of the classical and equitable chromatic numbers for 2-chromatic cubic graphs. Since the only cubic graph for which the chromatic number is equal to 4 is \( K_4 \), we have

\[
2 \leq \chi(G) = \chi(G) \leq 4
\]

for any cubic graph. Moreover, from (1) it follows that for any cubic graph \( G \neq K_4 \), we have

\[
n/3 \leq \alpha(G) \leq n/2
\]

where \( \alpha(G) \) is the independence number of \( G \). Note that the upper bound is tight only if \( G \) is bipartite.

Let \( Q_k \) denote the class of connected \( k \)-chromatic cubic graphs and let \( Q_k(n) \subset Q_k \) stand for the subclass of cubic graphs on \( n \) vertices, \( k = 2, 3, 4 \). Clearly, \( Q_4 = \{K_4\} \). In what follows we will call the graphs belonging to \( Q_2 \) bicubic, and the graphs belonging to \( Q_3 \) tricubic.

As mentioned, if \( G \) is bicubic then any 2-coloring of it is equitable and there may be no equitable 3-coloring (cf. \( K_{3,3} \) shown in Fig. 1). On the other hand, all graphs in \( Q_2(n) \) have a semi-equitable 3-coloring of type \([n/2], [n/4], [n/4]\) (or \([n/4], [n/4], [n/2]\)). Moreover, they are easy colorable in linear time while traversing them in a depth-first search (DFS) manner.

Let \( s_i \) be the speed of machine \( M_i \), for \( i = 1, 2, 3 \), and let \( s = s_1 + s_2 + s_3 \). Without loss of generality we assume that \( s_1 \geq s_2 \geq s_3 \). If there are just 6 jobs to schedule then the incompatibility graph \( G = K_{3,3} \) and there is only one decomposition of it into 3 independent sets shown in Fig. 1(a), as well as there is only one decomposition of \( G \) into 2 independent sets shown in Fig. 1(b), of course up to isomorphism. The length of minimal schedule is \( \min\{3s_1/2, 2s_1/2, 1s_1/3, 3s_1/2\} \). Therefore, we assume that our graphs have at least 8 vertices.

By an ideal schedule we mean a schedule in which:

(i) machine \( M_1 \) performs as many jobs as possible and \( M_2 \) and \( M_3 \) finish at the same time, if \( s_1 \geq s_2 + s_3 \), or
(ii) machines \( M_1, M_2, \) and \( M_3 \) all finish at the same time, if \( s_1 < s_2 + s_3 \).

An example of ideal schedule is shown in Fig. 2(a).
دریافت فوری
متن کامل مقاله

امکان دانلود نسخه تمام متن مقالات انگلیسی
امکان دانلود نسخه ترجمه شده مقالات
پذیرش سفارش ترجمه تخصصی
امکان جستجو در آرشیو جامعی از صدها موضوع و هزاران مقاله
امکان دانلود رایگان ۲ صفحه اول هر مقاله
امکان پرداخت اینترنتی با کلیه کارت های عضو شتاب
دانلود فوری مقاله پس از پرداخت آنلاین
پشتیبانی کامل خرید با بهره مندی از سیستم هوشمند رهگیری سفارشات