European Journal of Operational Research 243 (2015) 763–773

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

Inverse chromatic number problems in interval and permutation graphs ☆

Yerim Chung [a,*], Jean-François Culus [b], Marc Demange [c,d]

[a] School of Business, Yonsei University, Seoul 120-749, Korea
[b] CEREGMIA, University of Antilles-Guyane, Schoelcher, Martinique, France
[c] School of Mathematical and Geospatial Sciences, RMIT University, Melbourne, VIC, Australia
[d] LAMSADE, UMR CNRS 7243, Paris, France

ARTICLE INFO

Article history:
Received 25 November 2013
Accepted 15 December 2014
Available online 23 December 2014

Keywords:
Inverse combinatorial optimization
Graph coloring
Interval graphs
Permutation graphs

ABSTRACT

Given a graph G and a positive integer K, the inverse chromatic number problem consists in modifying the graph as little as possible so that it admits a chromatic number not greater than K. In this paper, we focus on the inverse chromatic number problem for certain classes of graphs. First, we discuss diverse possible versions and then focus on two application frameworks which motivate this problem in interval and permutation graphs: the *inverse booking problem* and the *inverse track assignment problem*. The inverse booking problem is closely related to some previously known scheduling problems; we propose new hardness results and polynomial cases. The inverse track assignment problem motivates our study of the inverse chromatic number problem in permutation graphs; we show how to solve in polynomial time a generalization of the problem with a bounded number of colors.

1. Introduction

In inverse combinatorial optimization, the goal is traditionally to modify as little as possible a given combinatorial optimization problem instance so as to make a fixed solution optimal in the modified instance. To this end, an instance Π of the combinatorial problem is associated with a parameter system w—most often a weight system—to be modified. Given such an instance $\Pi(w)$ and a feasible solution X_0, the inverse problem $I\Pi$ is defined (Ahuja & Orlin, 2001) as the problem to perturb the parameter system from w to w' such that X_0 becomes an optimal solution of the instance $\Pi(w')$, and the deviation $||w - w'||$ is minimum under a fixed norm.

Since Burton and Toint (1992) first investigated the inverse shortest path problem in the early 1990s, inverse combinatorial optimization has attracted the attention of the operations research community. It has been extensively studied for a variety of optimization problems. Many interesting results from both theoretical and practical points of view have been achieved (see, e.g., Ahuja & Orlin, 2001; Burkard, Pleschiutschnig, & Zhang, 2004; Liu & Ubhaya, 1997; Yang, Zhang, & Ma, 1997; Zhang & Cai, 1998; Zhang, Yang, & Cai, 1999). Many studies illustrate both the relevance of inverse combinatorial problems in operations research and the extent of theoretical interest in this family of difficult problems.

Some interesting variants have been considered (Chung & Demange, 2008; Chung & Demange, 2012; Demange & Monnot, 2010). $\{0, 1\}$-*inverse problems* involve parameter systems with $(0, 1)$ components in which the structure of an instance rather than just its weight system can change. For example, changing a vertex/edge weight from 1 to 0 is equivalent to deleting the related vertex/edge from the graph. Another variant, called the *inverse number problem*, specifies a target solution value instead of a solution. The goal is then to find, with minimum modification, a new instance with an optimal value not worse than the fixed target value.

In this paper we discuss an inverse framework for *minimum vertex coloring* which ranks among the most popular combinatorial optimization problems in operations research. Given a graph G, one associates colors with vertices such that every two vertices linked by an edge differ in color. The objective is to minimize the number of colors. The optimal value is called the *chromatic number of* G, denoted by $\chi(G)$. Graph coloring has many applications in various domains, particularly in scheduling (see, e.g., Demange, Ekim, & de Werra, 2009; Demange, Ekim, Ries, & Tanasescu, 2015; de Werra, 1996; Yáñez & rez, 2003). Tasks with pairwise incompatibilities are given such that any set of pairwise compatible tasks can be performed during a time slot. The aim is to execute all tasks in the fewest time slots. Tasks are associated with vertices and incompatibilities with edges in the

☆ A preliminary version of some results of this paper appeared in Proceedings of the First International Symposium in Combinatorial Optimization (ISCO 2010), Electronic Notes in Discrete Mathematics 36: 1129-1136, 2010.

* Corresponding author. Tel.: +821088073675.
E-mail addresses: yerimchung@yonsei.ac.kr (Y. Chung), jean-francois.culus@espe-martinique.fr (J.-F. Culus), marc.demange@rmit.com.au (M. Demange).

http://dx.doi.org/10.1016/j.ejor.2014.12.028
0377-2217/

incompatibility graph, and all tasks performed within a time slot correspond to a color class.

In the inverse case, fixing a solution (a fixed vertex coloring) appears to be less natural than fixing a target chromatic number for most graph coloring applications. As a consequence, the related inverse number problem (*inverse chromatic number problem*) seems more natural. In this paper, we introduce the inverse chromatic number problem, describe typical situations in applications that motivate use of this framework, and present computational complexity results. We also identify some known operations research problems that can be illustrated as inverse chromatic number models.

The paper is organized as follows. In Section 2, we formulate the problem, compare it with similar optimization problems, and discuss the notion of *inverse model*. In Section 3, we describe an operations research problem called *the inverse booking problem* that relates to the inverse chromatic number problem in interval graphs and we present some hardness and approximation results. Some polynomially solvable cases are proposed in Section 4. In Section 5, we describe the *inverse track assignment problem* that relates to the inverse chromatic number problem for permutation graphs. We briefly discuss the tractability of this problem and that of one generalization. Finally, in Section 6, some suggestions for further study are discussed.

2. Definitions and related problems

The inverse chromatic number problem can be formally defined as follows: given a graph G and a positive integer K, the graph should be modified as little as possible in such a way that the chromatic number of the resulting graph becomes K or less. An instance of the decision version is (G, K, M), where M is a non negative number. The related question is whether there is a modification of at most M such that the modified graph has a chromatic number of at most K.

2.1. Examples of and motivation for inverse models

In the *inverse model*, we specify how to modify the instance as well as how to measure the deviation. Different inverse models for the same problem obviously produce different inverse problems. Weight systems are most often modified (see, e.g., Ahuja & Orlin, 2001; Burton & Toint, 1992; Demange & Monnot, 2010). This inverse model is obviously not relevant to the usual graph coloring problems, which involve no weight system.

For graph coloring, various types of graph modifications may be considered. Some examples of inverse models are discussed below. In addition, any kind of norm may be used to measure the deviation. Many studies in this domain have shown that the choice of norm drastically influences the results. Cost functions that are not norms could also be considered. The only condition is that the minimum value of the deviation measure must correspond to the unchanged instance. In this paper, we mainly restrict our focus to the L_1 norm[1], unless otherwise specified. Other norms, in particular the L_∞ norm[2], may be considered in future studies. When restricting to a specific class of graphs, one should take care that it is stable under the considered modification.

2.1.1. Vertex deletion

At first glance, vertex deletion is one of the most natural models for the inverse chromatic number problem. Revisiting the previously mentioned scheduling application of minimum graph coloring, it consists in removing a minimum number of tasks such that the remaining tasks can be performed in K time slots. Then, the removed tasks can be externalized, with additional costs. This problem is equivalent to the maximum K-colorable subgraph problem which has been well studied. Given a graph G, the vertices not belonging to a maximum K-colorable subgraph of G are exactly the vertices to be deleted from G. Since the maximum K-colorable subgraph problem is polynomially solvable for comparability graphs, co-comparability graphs (Frank, 1980; Greene, 1976; Greene & Kleitman, 1976), and for split graphs if K is fixed (Yannakakis & Gavril, 1987), we obtain the following result:

Proposition 1.

(1) The inverse chromatic number problem with vertex deletion can be solved in polynomial time for comparability graphs and co-comparability graphs, and the same holds for split graphs if K is a fixed integer.
(2) The inverse chromatic number problem with vertex deletion is NP-hard for split graphs if K is not fixed.

The polynomial tractability of the inverse chromatic number problem for permutation and interval graphs immediately follows:

Corollary 2. *The inverse chromatic number problem with vertex deletion is polynomially solvable for permutation and interval graphs.*

2.1.2. Edge deletion

The inverse chromatic number problem with edge deletion consists in minimizing the number of conflicting edges (edges with both extremities of the same color) for a fixed number of colors. Conflicting edges must be removed in the inverse chromatic number problem with edge deletion. This problem is used in several studies using heuristic approaches to graph coloring (see, e.g., Galiniera & Hertz, 2006; Hertz & de Werra, 1990; Marmion, Blot, Jourdan, & Dhaenens, 2013).

Both vertex- and edge-deletion models are particular cases of the so called vertex and edge deletion problems studied in Yannakakis (1978). From the computational point of view, these problems were shown to be very hard, even in very restrictive graph classes. We describe below alternative inverse models. In contrast to the previous examples, these models cannot be immediately described as usual graph optimization problems. The framework of inverse optimization is better suited for this purpose.

2.1.3. Intersection graphs

Given a base set X and a family $\mathcal{F}$ of parts of X, $\mathcal{F}$ is the vertex set of the related intersection graph, with edges between two vertices if the related parts intersect each other. For such a graph, any modification strategy on the subsets of the base set brings a modification strategy on the graph. We present two illustrating examples below.

Two well-known applications of graph coloring in some intersection graph classes are frequency allocation in telecommunication networks (see, e.g., Eisenblätter, Grötsche, & Koster, 2002) and wavelength assignment in optical networks (see, e.g., Zang, Jue, & Mukherjee, 2000). In the former problem, transmitters/receiver units are spread over a region. A frequency must be assigned to each unit, taking into account that the total number of frequencies is limited and that interferences may occur if the same frequency is assigned to close transmitters. In the most basic model, frequencies are integer numbers, and each transmitter is associated with a disk centered at the location of the unit with a radius representing its signal power. Then, two units corresponding to intersecting disks must be assigned different frequencies to avoid interference for a receptor located in the intersection. The resulting interference graph is the intersection graph of these disks, called a *disk graph*, and the problem is exactly the minimum coloring problem in this class. In the latter application, one is given an optical network (represented by a host graph) and a collection of paths, each linking two nodes in the network. Each path must be associated with a wavelength such that two paths with the same wavelength cannot share a common edge. Thus, assigning

[1] The L_1 norm is defined by $||x||_1 = \sum_{i=1}^{n} |x_i|$ for a vector x of dimension n.
[2] The L_∞ norm is defined by $||x||_\infty = \max\{|x_1|, \ldots, |x_n|\}$ for a vector x of dimension n.

wavelengths to paths is a coloring problem in the edge-intersection graph of the paths: each path is a vertex, and two vertices are linked by an edge if the related paths have at least one edge in common.

These two applications induce natural situations that can be described in terms of the inverse chromatic number problem. For the frequency allocation problem, suppose that the network of transmitters/receiver units requires more frequencies than are available. The signal power of some units may then be limited so as to avoid interference and make frequency allocation possible. However, this reduction in signal power should not affect regional coverage. Thus, a natural inverse model arises for which the radius system must be modified with an additional constraint on the coverage area. Using the L_1-norm, the total power reduction is the primary objective, while using the L_∞-norm minimizes the number of units with reduced signal power.

For the wavelength assignment problem, if the number of different wavelengths is limited and no assignment is feasible, then modification of the routing of requests (paths) may be necessary in order to make it feasible. In the inverse model, the paths must be modified and the difference between two paths measured. For instance, using the Hamming distance allows us to minimize the number of links to be changed.

In this paper, we examine interval graphs, one of the simplest classes of intersection graphs. We present in Section 3 an application of the minimum inverse chromatic number problem with these graphs.

2.1.4. Other models

There are many other situations involving graph classes defined from a discrete structure like the set system for intersection graphs. Any method of modifying this structure induces an inverse model on the resulting graph. *Permutation graphs*, or the more general class of *overlap graphs* (see, e.g., Golumbic, 1980), provide an interesting example that we will consider in Section 5. These graphs can be defined from an interval system, as detailed in Section 5. Permutation graphs can be defined equally well from a permutation or a sequence of numbers. We present a motivating application that corresponds to a natural inverse model.

2.2. Some close problems

For most graph coloring problems, the aim is to compute the chromatic number of a given graph. however, several problems have been considered with the aim of defining or modifying the graph so as to guarantee some properties of the chromatic number. The inverse chromatic number problem clearly belongs in this class. Another problem of this type is the so-called *minimum selective coloring problem* (Demange et al., 2015; Demange, Monnot, Petrica, & Ries, 2014). An instance of this problem is a graph whose vertex set is partitioned into p *clusters*, and the objective is to select one vertex per cluster so as to minimize the chromatic number of the subgraph induced by the selected vertices. In the decision version, the question is whether it is possible to select one vertex per cluster such that the induced subgraph has a chromatic number no more than k, for a fixed k. A weighted version can be defined by associating a cost with each vertex. The aim is to find a k-colorable graph of minimum cost with one vertex per cluster.

In some cases, the inverse chromatic number problem can be reduced to a weighted selective graph coloring problem. Consider, for example, an inverse model based on some intersection graphs, as described above, with a modification strategy for each set of the original set system. For each vertex v of the intersection graph corresponding to a set X_v, we add all vertices associated with all possibilities of modifications for X_v. For instance, in the previously described model involving disk graphs, each possible radius of the disk associated with v will correspond to a new vertex. We draw an edge between two vertices if the related sets intersect. Then we consider as a cluster all vertices defined from v and associate each of them with a corresponding weight. Then, modifying the original graph can be accomplished by selecting one vertex per cluster in the new graph. The drawback of this method is that it requires enumerating all modifications of the original graph. Consequently, any inverse chromatic number problem may be transformed into a huge instance of weighted minimum selective coloring, even an infinite one. This reduction will be illustrated in Proposition 10 (Section 3).

Note that similar reductions could be established for many modification strategies, but not for all strategies, for instance, the edge deletion model.

In Demange et al. (2015), the minimum selective coloring problem is motivated by several applications in various domains including wavelength assignment, frequency allocations, berth allocation, and scheduling. The two examples of the application of the minimum chromatic number problem described above (disk graphs and edge intersection graphs of paths) are inspired by applications of minimum selective coloring. Most applications of minimum selective coloring have variants using the inverse chromatic number formulation.

A second close problem is the notion of *blockers* for the chromatic number introduced in Bazgan, Bentz, Picouleau, and Ries (2015). Given a graph and two integers d, ℓ, the problem is to decide whether it is possible to remove ℓ edges such that the chromatic number decreases by at least d. This problem is close to the inverse chromatic number problem within the edge deletion model. The only difference is that the aim is to decrease the chromatic number by a fixed value d instead of fixing a target chromatic number. These problems are equivalent in any graph classes for which minimum graph coloring is polynomial. For instance, the problem is polynomially solvable in split graphs and in complements of bipartite graphs if d is a fixed constant, and NP-hard in complements of bipartite graphs if d is not fixed Bazgan et al. (2015). We deduce immediately:

Proposition 3. *The inverse chromatic number problem with edge deletion is NP-hard in complements of bipartite graphs and polynomially solvable in both complements of bipartite graphs and split graphs under the additional condition that $|K - \chi(G)|$ is bounded by a constant.*

2.3. A remark about complexity

The decision version of the minimum graph coloring problem (K-colorability) polynomially Karp reduces to the decision version of the inverse chromatic number problem. From an instance (G, K) of the former problem (G is a graph and K a number), one can construct an equivalent instance $(G, K, M = 0)$ of the latter: the vertex coloring instance admits a chromatic number of at most K if and only if no modification is needed in the related instance of the inverse problem. Clearly, this reduction holds for any type of transformation and for any deviation measure satisfying the following two conditions: the measure has nonnegative values and is null if and only if nothing is changed in the instance. This is obviously the case in particular for any norm. Consequently, no matter how an instance is modified or measured, the inverse chromatic number problem is NP-hard in any class of graphs for which the graph coloring problem is NP-hard.

Revisiting the examples described for the intersection graph, we note that graph coloring is known to be NP-hard in disk graphs, even if all radii are equal to 1 (Gräf, Stumpf, & Weißenfels, 1998). This is also true in edge intersection graphs of paths even if the host graph is a tree (Golumbic & Jamison, 1985). Both results hold even if the intersection model is given, which is natural in most applications involving these intersection graph classes. As a consequence, the related inverse models correspond to hard inverse chromatic number problems. The same holds in overlap graphs. However, graph coloring is well known to be polynomial in interval and permutation graphs. For these two

cases, the status of the inverse chromatic number problem in terms of hardness cannot be so easily deduced. This issue will be addressed in Sections 3 and 5.

3. The inverse booking problem

We consider a set of intervals $I_1 = [a_1, b_1[, \ldots, I_n = [a_n, b_n[$ of length $p_1, \ldots, p_n$, each representing a reservation request for a resource[3] between two dates a_i and b_i, $i \in \{1, \ldots, n\}$. It is assumed that there are more than K reservations overlapping at a single time slot, while only K identical resources are available. The task is then to rearrange the given intervals by translating some of them so that:

(i) all intervals can be legally assigned to K identical parallel lines with no intersection between the intervals assigned to the same line, and
(ii) the translation cost under the L_1 norm—the total discrepancy between the original and the new interval position vectors—is minimum.

We call this problem the *inverse booking problem*, denoted by $I\text{Book}_K$. When interval translations are allowed only to the right-hand side, the problem is denoted by $I\text{Book}_{K,\rightarrow}$. In addition, when interval lengths are restricted (e.g., unitary intervals ($p_i = 1$) or equal length intervals ($p_i = p$)), we use the following notation: $I\text{Book}_{K,p_i=1}$, $I\text{Book}_{K,p_i=1,\rightarrow}$, $I\text{Book}_{K,p_i=p}$, and $I\text{Book}_{K,p_i=p,\rightarrow}$. The case with $K = 1$ is denoted by $I\text{Book}_{K=1}$ with similar notations for all other versions.

By definition, the inverse booking problem is a special case of the inverse chromatic number problem in interval graphs with specific modification rules (shifting intervals). The K parallel lines associated with resources represent the K different colors that are allowed in the graph coloring formulation.

Inverse booking problems are closely related to job scheduling problems. This gives a new insight into this well-known class of scheduling problems. For $K = 1$, $I\text{Book}_{K=1}$ can be seen as a special case of the so-called *total discrepancy problem* (Garey, Tarjan, & Wilfong, 1988), which is a single machine job scheduling problem with tardiness and earliness costs. An interval $I_i = [a_i, b_i[$ corresponds to a non-preemptive job J_i with a processing time $p(J_i) = b_i - a_i = p_i$, a due date $d(J_i) = b_i$, and a completion time $c(J_i) = b'_i$. Translation of an interval to the right-hand side corresponds to tardiness $T_i = b'_i - b_i$ of the job, and translation to the left-hand side means its earliness $E_i = a_i - a'_i$. So, using common scheduling notations, $I\text{Book}_{K=1}$ can be denoted by $1||\sum_i(E_i + T_i)$. This problem was studied by Garey et al. (1988) and its complexity was recently revisited in Wan and Yuan (2013).

Proposition 4.

(1) (Wan and Yuan (2013)) $1||\sum_i(E_i + T_i)$ (thus, $IBook_{K=1}$) is NP-hard in the strong sense.
(2) (Garey et al. (1988)) $1||\min\max_i(E_i + T_i)$ is polynomially solvable.

The second item in Proposition 4 deals with the inverse booking problem under the L_∞ norm and will be useful in what follows. In addition, $I\text{Book}_{K=1,\rightarrow}$ is equivalent to the minimum tardiness scheduling problem, denoted by $1|r_i|\sum_i T_i$, for non-preemptive jobs with arbitrary release dates. Indeed, the left endpoint a_i of each interval I_i can be seen as the release date r_i of the job J_i: translations are permitted only to the right-hand side, meaning that no job can be started before its release date. For this problem, the following result is known (Kan, 1976):

Proposition 5 (Kan, 1976). *$1|r_i|\sum_i T_i$ (thus, $IBook_{K=1,\rightarrow}$) is strongly NP-hard.*

[3] Depending on the underlying context, resources can include, for instance, machines, processors, rooms in a hotel booking system, etc.

For any bounded integer K, $I\text{Book}_K$ and $I\text{Book}_{K,\rightarrow}$ respectively correspond to the multiprocessor scheduling problem with tardiness and earliness costs, $K||\sum_i(E_i + T_i)$, and the problem with release dates and tardiness costs, $K|r_i|\sum_i T_i$.

$I\text{Book}_{K=1}$ can also be viewed as the unweighted version of the so-called *single machine just-in-time scheduling problem with earliness and tardiness costs*, denoted by $1||\sum_i(\eta_i E_i + \mu_i T_i)$. In Müller-Hannemann and Sonnikow (2009), it is shown that no approximation ratio of the form $O(c^n)$ for a constant c can be guaranteed for this problem unless $\mathbf{P} = \mathbf{NP}$. As the weights play a crucial role in the proof, the approximation behavior of the unweighted case is left as an open question. In what follows, we answer this question by showing that $I\text{Book}_{K=1}$ and $I\text{Book}_{K=1,\rightarrow}$ are both n-approximable, but not $n^{1-\varepsilon}$-approximable for any $\varepsilon > 0$.

Proposition 6. *$I\text{Book}_{K=1}$ and $I\text{Book}_{K=1,\rightarrow}$ are n-approximable.*

Proof. This is an easy consequence of Proposition 4, which stated that $I\text{Book}_{K=1}$ under the L_∞-norm can be solved in polynomial time. Since we have $|v|_\infty \le |v|_1 \le n|v|_\infty$ for every vector v of dimension n, we conclude that $I\text{Book}_{K=1}$ is n-approximable. The same result can be obtained for $I\text{Book}_{K=1,\rightarrow}$. □

A natural question is whether this approximation ratio of n can be improved. We provide a negative answer in Theorem 7. This result corroborates the NP-completeness results in the strong sense for $I\text{Book}_{K=1}$ (Wan & Yuan, 2013) and $I\text{Book}_{K=1,\rightarrow}$ (Kan, 1976), as mentioned in Corollary 8.

Theorem 7. *Suppose that all interval lengths are bounded by a polynomial function.*

(1) There is no polynomial time approximation algorithm for $I\text{Book}_{K=1}$ or for $I\text{Book}_{K=1,\rightarrow}$ guaranteeing an approximation ratio of $O(n^{1-\epsilon})$, $\epsilon > 0$, unless $\mathbf{P} = \mathbf{NP}$.
(2) The same result holds even if the interval graph corresponding to the initial set of intervals is bipartite.[4]

Proof. We use a polynomial-time reduction from *3-Partition*. An instance of *3-Partition* is given by a bound $B \in \mathbb{N}^+$ and a finite set $X = \{x_1, \ldots, x_{3m}\}$ of $3m$ elements, each of size $s(x_i) = d_i \in \mathbb{N}^+$ for $i \in \{1, \ldots, 3m\}$ such that $B/4 < d_i < B/2$ and $\sum_{i=1}^{3m} d_i = mB$. A 3-partition instance is said to be positive if the x_is can be partitioned into m disjoint groups of three elements, each summing exactly to B. We assume that the d_is and B are bounded by a polynomial function. Under this assumption, *3-Partition* is still NP-complete since *3-Partition* is strongly NP-complete (Garey & Johnson, 1979).

For the conciseness of the proof, we first consider $I\text{Book}_{K=1}$, the proof for $I\text{Book}_{K=1,\rightarrow}$ being very similar.

1. Let us fix two constants, c and $\epsilon > 0$, and show that the considered problem cannot be approximated within a ratio of $cn^{1-\epsilon}$. We assume that some interval lengths are given as rational numbers. Given an instance of *3-Partition* with d_is and B, we construct an instance of $I\text{Book}_{K=1}$ as follows (see Fig. 1).

The instance is made up of $m + 1$ interval blocks, $L^0, L^1, \ldots, L^m$, and $3m$ intervals, $I_1, \ldots, I_{3m}$.

- Each of the interval blocks L^j, $j \in \{0, 1, \ldots, m\}$, consists of $T = 3Q(m)m^2B$ contiguous intervals, $L^j_i = [l^j_i, r^j_i[$, $i \in \{1, \ldots, T\}$, of equal length. $Q(m)$ is a fixed polynomial function to be defined later.
- The first and last interval blocks L^0 and L^m are of total length $T = 3Q(m)m^2B$, i.e., each interval belonging to L^0 or L^m has a length of 1, while the total length of the other blocks $L_1, \ldots, L_{m-1}$ equals B, i.e., each of $L_1, \ldots, L_{m-1}$ consists of T contiguous intervals of length $\frac{1}{3Q(m)m^2}$.

[4] Even in the case where two machines are enough to process all jobs, the single machine scheduling problem is still difficult, and cannot be approximated within a ratio of $O(n^{1-\epsilon})$, $\epsilon > 0$.

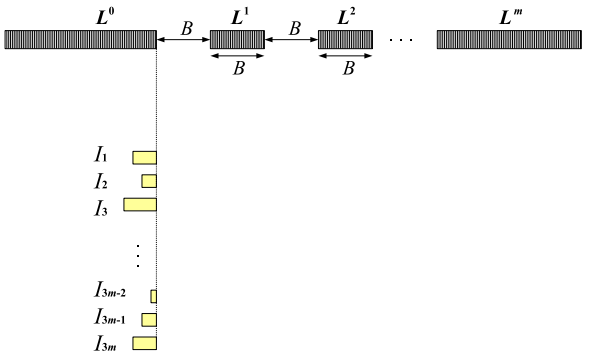

Fig. 1. The instance of $I\text{Book}_{K=1}$ associated with an instance of 3-Partition.

- The $m+1$ blocks $L^0, L^1, \ldots, L^m$ are distributed along a horizontal line in this order with a space of length B between every two consecutive interval blocks. We set the right endpoint of L^0 to 0: $r_T^0 = 0$. Each interval block can be written as follows:
  - for $i \in \{1, \ldots, T\}$, $L_i^0 = [-3Q(m)m^2B + (i-1), -3Q(m)m^2B + i[$,
  - for $j \in \{1, \ldots, m-1\}$ and $i \in \{1, \ldots, T\}$, $L_i^j = [(2j-1)B + \frac{i-1}{3Q(m)m^2}, (2j-1)B + \frac{i}{3Q(m)m^2}[$,
  - for $i \in \{1, \ldots, T\}$, $L_i^m = [(2m-1)B + (i-1), (2m-1)B + i[$.
- The $3m$ elements of the *3-Partition* instance are represented by $3m$ intervals $I_i = [-d_i, 0[$, $i \in \{1, \ldots, 3m\}$, of length d_i.
- Since B is bounded by a polynomial function, the total number N of intervals clearly satisfies $N \leq P(m)Q(m)$ for a fixed polynomial function P. We then choose $Q > c^{1/\epsilon}P^{(1-\epsilon)/\epsilon}$, implying $cN^{1-\epsilon} < Q(m)$.

We will show that, in the case where the 3-partition instance admits a positive answer, we have $\beta \leq 3m^2B$, where β is the optimal value of the constructed $I\text{Book}_{K=1}$ instance. In the opposite case, we have $\beta \geq 3Q(m)m^2B$. We then conclude that no polynomial-time algorithm guarantees the ratio $\rho(n)$, where n is the number of intervals.

Let us first consider the case where a 3-partition exists for the original instance: the set X can be partitioned into m disjoint 3-sets $X_1, \ldots, X_m$ such that for any $j \in \{1, \ldots, m\}$, $\sum_{x \in X_j} s(x) = B$. By renumbering the intervals I_is ($i \in \{1, \ldots, 3m\}$), we may assume without loss of generality that for $j \in \{1, \ldots, m\}$, three elements of X_j correspond to intervals I_{3j-2}, I_{3j-1} and I_{3j}, respectively. Let us denote by $B_j = [r_T^{j-1}, l_1^j[$, $j \in \{1, \ldots, m\}$, the space between L^{j-1} and L^j of length B. Each B_j can be filled up with three intervals I_{3j-2}, I_{3j-1} and I_{3j} for any $j \in \{1, \ldots, m\}$. Translating I_{3j-2}, I_{3j-1} and I_{3j} to the right-hand side at least by $(2j-2)B$ and at most by $(2j-1)B$ yields a feasible solution for $I\text{Book}_{K=1}$ of a value not greater than $3m^2B$. Hence, we have $\beta \leq 3m^2B$.

Suppose now that there is no 3-partition for X. In this case, the $3m$ intervals I_is cannot all be inserted within the spaces B_js. In order to have all the intervals stand in line, at least one of the following two actions should be conducted: (i) shifting some I_is either to the left-hand side of L^0 or to the right-hand side of L^m, and (ii) widening some B_js at least by 1 (since $d_i \in \mathbb{N}$ for $i \in \{1, \ldots, 3m\}$). Since the interval blocks L^0 and L^m are of total length $T = 3Q(m)m^2B$, action (i) yields a cost of (at least) $3Q(m)m^2B$. To widen some B_js, one needs to shift some block L^j, $j \in \{0, \ldots, m\}$, at least by a distance of 1 to the left- or right-hand side. Since each interval block is composed of $T = 3Q(m)m^2B$ small intervals, such an action yields a cost of at least $3Q(m)m^2B$. Hence, we have $\beta \geq 3Q(m)m^2B$.

Assume that there is a polynomial-time approximation algorithm for $I\text{Book}_{K=1}$ which guarantees the approximation ratio $\rho(n) \leq cn^{1-\epsilon}$ for any $\epsilon > 0$, where c is a constant and n is the number of intervals of the instance. In case of a 3-partition ($\beta \leq 3m^2B$), this algorithm delivers a solution for the $I\text{Book}_{K=1}$ instance with N intervals of value $\lambda \leq 3m^2B\rho(N) \leq 3cN^{1-\epsilon}m^2B < 3Q(m)m^2B$ (recall that $cN^{1-\epsilon} < Q(m)$). In the opposite case where no 3-partition exists, the algorithm returns a solution of value $\lambda \geq \beta \geq 3Q(m)m^2B$. This implies the existence of a polynomial-time algorithm which solves *3-Partition*, contradicting the strong NP-completeness of *3-Partition*.

2. In the previous construction, the related interval graph (see Fig. 1) is $(3m+1)$-colorable. Based on the same structure, we construct a 2-colorable instance of $I\text{Book}_{K=1}$ by repositioning the $3m$ intervals $I_i = [a_i, b_i[$, $1 \leq i \leq 3m$, in such a way that $b_{i+1} = a_i$, $i \in \{1, \ldots, 3m-1\}$ and $b_1 = 0$. We assume without loss of generality that $d_1 \leq d_2 \leq \cdots \leq d_{3m}$, where $d_i = b_i - a_i$ for $i \in \{1, \ldots, 3m\}$.

Taking into account these changes,[5] we assume that each of the interval blocks $L^1, \ldots, L^{m-1}$ consists of $5Q(m)m^2B$ intervals of length $\frac{1}{5Q(m)m^2}$, and that block L^m consists of $5Q(m)m^2B$ intervals of length 1. To ensure that no interval I_i, $1 \leq i \leq 3m$ will be shifted to the left-hand side of the block L^0, we increase the total length of L^0 to $8Q(m)m^2B$. As in the previous case, the polynomial function Q is chosen so that the number N of intervals satisfies $cN^{1-\epsilon} < Q(m)$. If there is a polynomial-time approximation algorithm for $I\text{Book}_{K=1}$, guaranteeing the ratio $O(n^{1-\epsilon})$, $\epsilon > 0$, then it would deliver a solution of value $\lambda < 5Q(m)m^2B$ for the case where a 3-partition exists ($\beta \leq 3m^2B + X_0 \leq 5m^2B$) and a solution of value $\lambda \geq 5Q(m)m^2B$ for the opposite case. Such an algorithm would solve *3-Partition*, which is impossible unless **P=NP**. Hence, $I\text{Book}_{K=1}$ with the bipartite constraint cannot be approximated within the ratio of $O(n^{1-\epsilon})$, $\epsilon > 0$.

Recall that some interval lengths are defined as rational numbers. By multiplying all interval lengths by the same number (e.g., $3Q(m)m^2$ and $5Q(m)m^2$ for Proofs (1) and (2), respectively), we can easily obtain an instance with integer data. The value of each solution should also be multiplied by this common value.

Finally, an instance of $I\text{Book}_{K=1,\rightarrow}$ can be constructed in a similar way (but in this case, L^0 can be replaced by a single unit interval). The same arguments hold for this case, which concludes the proof of the theorem. □

The strong NP-hardness of $1||\sum_i(E_i + T_i)$ ($I\text{Book}_{K=1}$) (Wan & Yuan, 2013) and of $1|r_i|\sum_i T_i$ ($I\text{Book}_{K=1,\rightarrow}$) (Kan, 1976) can be obtained as a direct corollary:

Corollary 8. *For every $K \geq 1$, $I\text{Book}_K$ and $I\text{Book}_{K,\rightarrow}$ are strongly NP-hard.*

Proof. In the proof of Theorem 7, we show that the 3-partition problem with d_is and B bounded by a polynomial function polynomially reduces to $I\text{Book}_{K=1}$ and $I\text{Book}_{K=1,\rightarrow}$. In addition, all numbers involved in the constructed instances of $I\text{Book}_{K=1}$ and $I\text{Book}_{K=1,\rightarrow}$ are bounded by a polynomial function. This implies that both problems are NP-hard in a strong sense. Obviously, this result also holds for any fixed value of $K \geq 1$. □

As a natural extension, we investigate the particular case of $I\text{Book}_{K,\rightarrow}$ where there is only a constant number of different interval lengths. For brevity, the proof for the following proposition is given in Appendix A.

Proposition 9. *If all interval lengths belong to the set $\{1, 2, 5, 7, 9, L, L+3, 8L+5\}$, where $L \in \mathbb{N}$ is a polynomial function depending on n, then $I\text{Book}_{K,\rightarrow}$ is* NP*-hard.*

4. Some polynomial cases for the inverse booking problem

Previously, we pointed out the equivalence between inverse booking problems and scheduling problems. Indeed, $I\text{Book}_{K=1}$ is equivalent to the total discrepancy problem $K = 1||\sum_i(E_i + T_i)$, and

[5] In comparison with the previous construction, intervals I_is are shifted to the left-hand side by $D = d_1 + (d_1 + d_2) + \cdots + (d_1 + \cdots + d_{3m-1}) = (3m-1)d_1 + (3m-2)d_2 + \cdots + 2d_{3m-2} + d_{3m-1} \leq \frac{3}{2}m^2B$.

$IBook_{K\geq 1,\rightarrow}$ is equivalent to the multiprocessor scheduling problem with release dates, $K|r_i|\sum_i T_i$. These equivalences lead to some polynomial results for the inverse booking problem and its variants. In particular, if job processing times are all equal, then the problem $K=1|p_i=p|\sum_i(E_i+T_i)$ (hence, $IBook_{K=1,p_i=p}$) can be solved in polynomial time (Garey et al., 1988), even with rational value starting dates (Verma & Dessouky, 1998).

In addition, Baptiste (2000) showed that when all job processing times are equal, $K|p_i=p,r_i|\sum_i T_i$ (and hence $IBook_{K\rightarrow,p_i=p}$) can be solved in polynomial time for any fixed $K\geq 1$. The related complexity is $(O(|V|^{3K+2})$, which was improved in Brucker and Kravchenko (2005) using a linear programming approach. Moreover, if all processing times are equal to 1, it is well known that a common flow approach can be applied for $K|p_i=1,r_i|\sum_i T_i$ (see, e.g., Baptiste & Brucker, 2004), even in a weighted case or one with more general objectives that can be expressed as the sum of non-decreasing functions of completion times. A similar approach can be taken for $K|p_i=1,r_i|\sum_i(E_i+T_i)$, and thus for $IBook_{K,p_i=1}$ (see Brucker, 1995). These results are now very familiar. Explicit proof is provided in Proposition 10, which allows us to evaluate the related complexity. Note that even a weighted version of the results with specific translation costs (not considered here) can be obtained by the same method.

For the rest of this section, we restrict our investigation to intervals of length 1. Thus, an interval system is entirely defined by a *position vector* $\mathbf{l}_0=(a_1,\ldots,a_n)$, each component $a_i, i\in\{1,\ldots,n\}$ denoting the left endpoint of the ith interval. Without loss of generality, we assume that $a_1\leq\cdots\leq a_i\leq\cdots\leq a_n$.

Proposition 10 (See Baptiste & Brucker, 2004; Brucker, 1995)**.** *If interval lengths are all equal to 1 and their endpoints have integer values, then for any K, $IBook_{K,p_i=1}$ and $IBook_{K,p_i=1,\rightarrow}$ can be solved in time $O(n^4\log n)$. Equivalently, this holds for $K|p_i=1|\sum_i(E_i+T_i)$ and $K|p_i=1,r_i|\sum_i T_i$.*

Proof. We sketch the proof using a very classical method. We denote by $\mathcal{K}=\{1,\ldots,K\}$ the set of K lines. Without loss of generality, we can assume $K<n$ since, in the opposite case, the optimal value of the instance is zero. Consider a set $\mathcal{I}$ of n intervals $I_i=[a_i,a_i+1[, i\in\{1,\ldots,n\}$, such that $a_i\in\mathbb{Z}, i\in\{1,\ldots,n\}$, and $a_1\leq\cdots\leq a_i\leq\cdots\leq a_n$. Let $(a_1,\ldots,a_n)$ be the related position vector. Every optimal solution $(a_1^*,\ldots,a_n^*)$ for $IBook_{K,p_i=1}$ satisfies $a_i^*\in\mathbb{Z}, i\in\{1,\ldots,n\}$. A feasible solution with integer endpoints can be defined by specifying a *state* (i,q,k) for every interval $i\in\{1,\ldots,n\}$, where $q\in\mathbb{Z}$ and $k\in\mathcal{K}$. The state (i,q,k) indicates that the ith interval I_i is translated by distance $q\in\mathbb{Z}$ and is assigned to the kth line for $k\in\mathcal{K}$.

Note that in any optimal solution, no interval is translated by more than $\lceil\frac{n}{2K}\rceil$. Indeed, suppose that an interval $I_i=[a_i,a_i+1[$ is translated by distance $D\in\mathbb{Z}$. Then, excluding I_i, there are $2D-1$ intervals $[x_i,x_i+1[$ with $x_i\in\{a_i-D+1,\ldots,a_i+D-1\}$ on each of the K lines; otherwise one can obtain a better solution by translating $I_i=[a_i,a_i+1[$ by distance $D'<D$. So, the total number n of intervals is at least $K(2D-1)+1$, which implies that $D\leq\lceil\frac{n}{2K}\rceil$. Consequently, we can restrict ourselves to the set of states $\mathcal{S}=\mathcal{I}\times\{-\lceil\frac{n}{2K}\rceil,\ldots,\lceil\frac{n}{2K}\rceil\}\times\mathcal{K}$.

So, for every interval $I_i\in\mathcal{I}$, we denote by $A_i=\{a_i+q, q\in\{-\lceil\frac{n}{2K}\rceil,\ldots,\lceil\frac{n}{2K}\rceil\}\}$ the set of all possible left endpoints of I_i after translation. Let $A=\bigcup_{i\in\{1,\ldots,n\}}A_i$. We construct an edge-weighted bipartite graph $\widetilde{B}=(\mathcal{I}\cup(A\times\mathcal{K}),\widetilde{E}_B)$ with complete connections between I_i and $A_i\times\mathcal{K}$. There is a one-to-one correspondence between edges in $\widetilde{E}_B$ and all possible states in $\mathcal{S}$. As $K\leq n$, $|\mathcal{S}|=|\widetilde{E}_B|=O(n^2)$. We then define an edge-weight function, assigning the weight $|q|$ to the edge associated with state (i,q,k). This corresponds to the translation cost of the related interval.

Then, the problem $IBook_{K,p_i=1}$ reduces to the computation of a minimum weight matching of size n in $\widetilde{B}$, which can be solved by the Hungarian method in time $O(|V|(|E|+|V|\log|V|))$, where $|V|$ is the number of vertices and $|E|$ is the number of edges in the bipartite graph (Schrijver, 2003). Since $\widetilde{B}$ has at most $O(n^2)$ vertices and $O(n^2)$ edges, $IBook_{K,p_i=1}$ can be solved in time $O(n^4\log n)$. For the case where translations are only allowed to the right-hand side, we restrict the allowable translation distance q to the values in $\{0,\ldots,\lfloor\frac{n}{K}\rfloor\}$, obtaining the same result for $IBook_{K,p_i=1,\rightarrow}$. □

As the main focus of this section, we provide a polynomial case that generalizes the cases mentioned earlier. We show that the problem $IBook_{K,\rightarrow}$ with a fixed number ℓ of different interval lengths $p_1,\ldots,p_\ell$ that are all bounded by a polynomial function can be solved in polynomial time for a fixed number K of lines. To our knowledge, this polynomial case is heretofore unknown. The main difference from the hard case stated in Proposition 9 is that the number of lines (machines) is bounded.

We consider $p_1,\ldots,p_\ell$ to be the possible values of interval lengths, all being bounded by a polynomial function with respect to the number n of intervals. For any $j\in\{1,\ldots,\ell\}$, let n_j be the number of intervals of length p_j. We then have $n=\sum_{j=1}^{\ell}n_j$. For any $j\in\{1,\ldots,\ell\}$ and $i\in\{1,\ldots,n_j\}$, let a_i^j be the initial left endpoint of interval I_i^j of length p_j, and we denote by $(a_i^j)_{i\in\{1,\ldots,n_j\},j\in\{1,\ldots,\ell\}}$ the initial position vector. We assume that $a_1^j\leq\cdots\leq a_i^j\leq\cdots\leq a_{n_j}^j$ for every $j\in\{1,\ldots,\ell\}$. We suppose that all entries $a_i^j, i\in\{1,\ldots,n_j\}.j\in\{1,\ldots,\ell\}$, and lengths $p_j.j\in\{1,\ldots,\ell\}$ are integers. Without loss of generality, we assume that for any $j\in\{1,\ldots,\ell\}$ and $i\in\{1,\ldots,n_j\}$, $a_i^j\in\mathbb{N}$, $p_j\in\mathbb{N}$, and $\min_{i,j}a_i^j=0$.

Similarly, we denote by $(f_i^j)_{j\in\{1,\ldots,\ell\},i\in\{1,\ldots,n_j\}}$ any position vector associated with a feasible solution for the given instance. By definition of the problem $IBook_{K,\rightarrow}$, we have $f_i^j\geq a_i^j$ for every $j\in\{1,\ldots,\ell\}$ and $i\in\{1,\ldots,n_j\}$.

Definition 1. A feasible position vector $(f_i^j)_{j\in\{1,\ldots,\ell\},i\in\{1,\ldots,n_j\}}$ is said to be monotonous if $f_1^j\leq\cdots\leq f_i^j\leq\cdots\leq f_{n_j}^j$ for every $j\in\{1,\ldots,\ell\}$ and $i\in\{1,\ldots,n_j\}$.

Note that for any non-monotonous feasible solution, if for a given $j_0\in\{1,\ldots,\ell\}$ we have $a_{i_1}^{j_0}\leq a_{i_2}^{j_0}$ and $f_{i_1}^{j_0}>f_{i_2}^{j_0}$, then defining $\tilde{f}_{i_1}^{j_0}=f_{i_2}^{j_0}$, $\tilde{f}_{i_2}^{j_0}=f_{i_1}^{j_0}$, and $\tilde{f}_i^j=f_i^j$ for all other $(i,j)\neq(i_1,j_0),(i_2,j_0)$, we get a new feasible solution of better value. So we have:

Lemma 11. *If there exists an optimal solution for $IBook_{K,\rightarrow}$, then there is a monotonous optimal solution.*

In addition, we can assume the following restrictions on position vectors.

Lemma 12. *Without loss of generality, we can restrict ourselves to position vectors satisfying, for every $j\in\{1,\ldots,\ell\}$ and $i\in\{1,\ldots,n_j\}$, $f_i^j\in\{a_i^j,a_i^j+1,\ldots,a_i^j+(n-1)L\}$, where n is the total number of intervals and L is the maximum interval length.*

Proof. Consider indeed a feasible position vector $(f_i^j)_{j\in\{1,\ldots,\ell\},i\in\{1,\ldots,n_j\}}$ with $f_{i_0}^{j_0}>a_{i_0}^{j_0}+(n-1)L$ for some i_0 and j_0 (the interval $I_{i_0}^{j_0}$ is translated by more than $(n-1)L$). Suppose that $I_{i_0}^{j_0}$ is assigned to line k. Denote by A the set of all intervals I_s^t allocated to line k and such that $f_s^t\geq a_{i_0}^{j_0}, (s,t)\neq(i_0,j_0)$. It would be less costly to insert $I_{i_0}^{j_0}$ on line k at the first available position after shifting all intervals in A by $p_{j_0}\leq L$. More precisely, denote by $b=\max\{f_i^j+p_j|I_i^j$ assigned to line k and $f_i^j<a_{i_0}^{j_0}\}$ the largest right-hand point of intervals assigned to line k and positioned before $a_{i_0}^{j_0}$. We then define $\tilde{f}_{i_0}^{j_0}=\max(a_{i_0}^{j_0},b)$; for every $I_s^t\in A, \tilde{f}_s^t=f_s^t+p_{j_0}$ and for all other intervals I_i^j, we pose $\tilde{f}_i^j=f_i^j$. The new position vector $\tilde{f}$ is feasible and less costly than f. □

As an immediate consequence, one can assume that the distance between two consecutive a_i^js is no more than $(n-1)L$. If this is not

the case, then the instance can be split into two independent sub-instances.

A consequence of Lemma 12 is that $\forall j \in \{1,\dots,\ell\}, i \in \{1,\dots,n_j\}$ we have $f_i^j \le n^2L$. Indeed, assuming that (i) $\min_{i,j}(a_i^j) = 0$, (ii) the distance between two consecutive a_i^js is at most $(n-1)L$ and (iii) $\forall i,j, f_i^j \le a_i^j + (n-1)L$, we get $\max_{i,j} f_i^j \le n(L+(n-1)L) = n^2L$.

We then design a dynamic programming algorithm as follows. For $k_j \le n_j$, $q_r \in \{0,\dots,(n^2+1)L\}$ and $t_r \in \{1,\dots,\ell\}$, with $j \in \{1,\dots,\ell\}$ and $r \in \{1,\dots,K\}$, we denote by $S(k_1,\dots,k_\ell,q_1,\dots,q_K,t_1,\dots,t_K)$ the set of all (partial) solutions where, for every $j \in \{1,\dots,\ell\}$, the k_j intervals I_i^j, $i \in \{1,\dots,k_j\}$ of length p_j are monotonously positioned on K lines, and the right-most interval assigned to line $r \in \{1,\dots,K\}$ ends at q_r and has length p_{t_r}.

We denote by $S(k_1,\dots,k_\ell,q_1,\dots,q_K,t_1,\dots,t_K)$ the state of the interval system such that the first $k_j \le n_j$ intervals of length p_j are already positioned.

If $S(k_1,\dots,k_\ell,q_1,\dots,q_K,t_1,\dots,t_K) = \emptyset$, then we set $V(k_1,\dots,k_\ell,q_1,\dots,q_K,t_1,\dots,t_K) = +\infty$. With these notations, the optimal value of the considered instance is then:

$$\min_{(q_1,\dots,q_K,t_1,\dots,t_K)\in\{0,\dots,n^2L\}^K\times\{1,\dots,\ell\}^K} V(n_1,\dots,n_\ell,q_1,\dots,q_K,t_1,\dots,t_K)$$

In the dynamic programming process, the induction is maid on $\sum_{j=1}^{\ell} k_j$. The main step is explained in Lemma 13.

Let $S_N = \{S(k_1,\dots,k_\ell,q_1,\dots,q_K,t_1,\dots+t_K) | k_1+\dots,k_\ell \le N\}$ be the set of states for which the total number of intervals already positioned is at most N. We have: $|S_N| \le N^\ell((n^2+1)L)^K\ell^K \le O(L^K n^{\ell+2K})$ for fixed constants ℓ and K.

Lemma 13. *Suppose that $V(k'_1,\dots,k'_\ell,q'_1,\dots,q'_K,t'_1,\dots,t'_K)$ is known for all $(k'_1,\dots,k'_\ell)$ such that $k'_1+\dots+k'_\ell \le N$, and for all $(q'_1,\dots,q'_K,t'_1,\dots,t'_K)$. Then, Algorithm 1 computes all values of $V(k_1,\dots,k_\ell,q_1,\dots,q_K,t_1,\dots,t_K)$ for $k_1+\dots+k_\ell = N+1$. It runs in time $O(n^2L)$ if K and ℓ are fixed constants.*

Algorithm 1 Main step of the dynamic programming algorithm.

Require: Values of $V(k'_1,\dots,k'_\ell,q'_1,\dots,q'_K,t'_1,\dots,t'_K)$ with $k'_1+\dots+k'_\ell \le N$ and a state $(k_1,\dots,k_\ell,q_1,\dots,q_K,t_1,\dots,t_K)$ with $k_1+\dots+k_\ell = N+1$.

Ensure: $V(k_1,\dots,k_\ell,q_1,\dots,q_K,t_1,\dots,t_K)$.

1: $\bar{r} \leftarrow argmax\{q_1,\dots,q_K\}$

2: **if** $k_{t_{\bar{r}}} \ne 0$ and $a_{k_{t_{\bar{r}}}}^{t_{\bar{r}}} + p_{t_{\bar{r}}} \le q_{\bar{r}}$ **then**

3: $V(k_1,\dots,k_\ell,q_1,\dots,q_K,t_1,\dots,t_K) \leftarrow \min_{\substack{q\in\{0,\dots,q_{\bar{r}}-p_{t_{\bar{r}}}\}\\ t\in\{1,\dots,\ell\}}} [V(k_1,\dots,\underset{\uparrow\, t_{\bar{r}}}{k_{t_{\bar{r}}}-1},\dots,k_\ell,q_1,\dots,\underset{\uparrow\, \bar{r}}{q},\dots,q_K, t_1,\dots,\underset{\uparrow\, \bar{r}}{t},\dots,t_K) + (q_{\bar{r}} - p_{t_{\bar{r}}} - a_{k_{t_{\bar{r}}}}^{t_{\bar{r}}})]$

4: **else**

5: $V(k_1,\dots,k_\ell,q_1,\dots,q_K,t_1,\dots,t_K) \leftarrow +\infty$

6: **end if**

Proof. Let $\bar{r}$, $1 \le \bar{r} \le K$ be the line such that $q_{\bar{r}} = \max\{q_1,\dots,q_K\}$. Algorithm 1 considers every possible state obtained from $S(k_1,\dots,k_\ell,q_1,\dots,q_K,t_1,\dots,t_K)$ by removing the interval $I_{k_{t_{\bar{r}}}}^{t_{\bar{r}}}$, and computes (see Line 3) the best solution value $V(k_1,\dots,k_\ell,q_1,\dots,q_K,t_1,\dots,t_K)$. Regarding the complexity, Line 1 requires a constant time and Line 3 requires $O(n^2L)$ time if K and ℓ are fixed constants. □

Using Lemma 13, an algorithm is immediately obtained that computes all values of $V(k_1,\dots,k_\ell,q_1,\dots,q_K,t_1,\dots,t_K)$. For any given ℓ and K, the related complexity is $O(\sum_{N=1}^{n}|S_N|n^2L) = O(n^{\ell+2K+3}L^{1+K})$. So, if ℓ and K are fixed constants and L is a polynomial function of n, then the related complexity is polynomial. This leads to the following theorem:

Theorem 14. *$Book_{K,\to}$ can be solved in polynomial time under the following conditions: all parameters have integer values, the number of different interval lengths and the number of parallel lines are fixed, and the maximum interval length is bounded by a polynomial function.*

5. The inverse track assignment problem: An example in permutation graphs

Let us consider the following track assignment problem described in Brucker and Nordmann (1994): in a train depot, trains should be stored during the night on some tracks. Each track may contain several trains organized as a stack according to the *First In Last Out* rule: the train entering last on the track must leave first the next morning. So, on each track, trains must be stored in the order of their arrival time and in the reverse order of their departure time. Such an assignment is called *legal*. It is assumed that, at a given time (e.g., midnight), all trains are stored on tracks. This is called the *midnight condition*. The minimum track assignment problem consists in finding a legal assignment using the minimum number of parallel tracks.

For an inverse version of this problem, suppose that there are n trains and K tracks. The arrival and departure times of each train are known. If no legal assignment exists, then the departure times must be modified as little as possible so that n trains with new departure times can be legally assigned to K tracks.

One can represent each train on a two-dimensional *lattice* plan with the arrival times on the x-axis and the departure times in reverse order or the complements of the departure times with respect to a fixed time (e.g., 12 a.m. in Fig. 2) on the y-axis. An example schedule and the lattice representation are given in Table 1 and Fig. 2, respectively.

Given this schedule and its lattice representation, there is a legal assignment of n trains to K tracks if and only if the different points in the lattice representation can be partitioned into K non-decreasing subsequences. So, the inverse track assignment problem involves modifying as little as possible the values of y-coordinates so as to obtain at most K non-decreasing subsequences in the lattice representation.

This problem can be represented using a *permutation graph*. A permutation graph (Golumbic, 1980) is usually constructed from

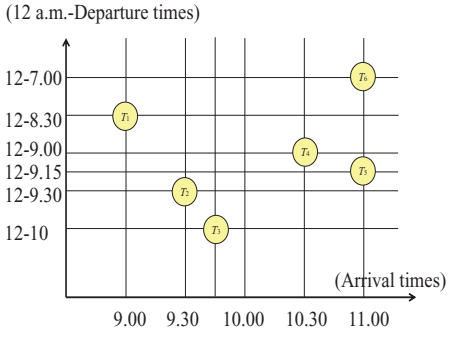

Fig. 2. The lattice representation of train times.

Table 1
Train arrival and departure times.

Train	Arrival time	Departure time
T_1	9 p.m.	8.30 a.m.
T_2	9.30 p.m.	9.30 a.m.
T_3	9.45 p.m.	10 a.m.
T_4	10.30 p.m.	9 a.m.
T_5	11 p.m.	9.15 a.m.
T_6	11 p.m.	7 a.m.

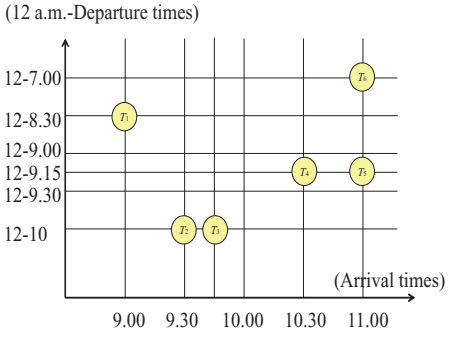

Fig. 3. An optimal modification for $K = 2$.

a permutation, but it can also be constructed from a sequence $(\pi = (\pi_1, \ldots, \pi_n))$ of numbers: each vertex corresponds to one number in the sequence, and two vertices corresponding to π_i and π_j are connected by an edge if $i < j$ and $\pi_i > \pi_j$. Such a graph can be equivalently represented by a list $\mathcal{L} = (P_1, \ldots, P_n)$ of points, $P_i = (x_i, y_i)$, with x-coordinate x_i and y-coordinate y_i in the two-dimensional plan.

In this permutation graph representation, a stable set is a set of vertices in $\mathcal{L}$ that are in non-decreasing order (vertices along a vertical or horizontal line are listed in non-decreasing order), while a clique is either a single vertex or vertices in decreasing order. The graph is K-colorable if and only if the set $\mathcal{L}$ can be divided into at most K non-decreasing subsequences. Using this model, the inverse track assignment problem corresponds to a particular case of the inverse chromatic number problem in permutation graphs. The task consists in modifying the y-coordinates in a fixed lattice representation. By assuming that y-coordinates can only be decreased, one can define a variant of the inverse track assignment problem where train departures can only be delayed. Here, we use the L_1-norm to measure a modification of the instance, but other norms can also be used. For the example corresponding to Table 1 and $K = 2$, an optimal modification is given in Fig. 3 that corresponds to delaying the departure time of train T_2 by 30 minutes and the departure time of train T_4 by 15 minutes. The total cost is then 45 minutes.

If the midnight condition is not assumed, then the problem becomes an inverse chromatic number problem in a larger class of graphs, called *overlap graphs*. Given n intervals associated with n trains, an overlap graph (Golumbic, 1980) is defined as follows: each vertex represents an interval during which the related train must be stored in the depot, and two vertices are joined by an edge if and only if the two corresponding intervals overlap, but they do not properly contain each other. If the midnight condition is satisfied (all intervals share a common point), then the resulting overlap graph is a permutation graph. The minimum k-colorability problem is known to be NP-complete in overlap graphs (Garey, Johnson, Miller, & Papadimitriou, 1980). As mentioned in Introduction, this implies that the related inverse chromatic number problem is NP-hard.

Corollary 15. *The inverse track assignment problem without the midnight condition is NP-hard.*

5.1. The inverse (p, k)-colorability problem on permutation graphs

Here, we consider a generalization of the inverse chromatic number problem called the *inverse* (p, k)*-colorability problem*, and denoted by $ICOL_{(p,k)}$. The case with $p = 0$ and $k = K$ corresponds to the inverse track assignment problem. Given a permutation graph defined by a lattice representation $\mathcal{L}$, the problem consists in modifying as little as possible the y-coordinates, from $\mathcal{L}$ to $\mathcal{L}'$, in such a way that $\mathcal{L}'$ can be partitioned into p decreasing subsequences and k non-decreasing subsequences. A single vertex can be considered as either a decreasing or non-increasing subsequence. Equivalently, the graph $G[\mathcal{L}']$ associated with $\mathcal{L}'$ can be partitioned into p cliques and k stable sets (i.e., it admits a (p, k)-coloring).

We propose a dynamic programming algorithm for solving $ICOL_{(p,k)}$. Given an instance defined by a list $\mathcal{L} = (P_1, \ldots, P_n)$, with $P_i = (x_i, y_i)$, we denote by Y the set of y-coordinates. Without loss of generality we assume that $\mathcal{L}$ is in non-decreasing lexicographic order: $x_1 \leq \cdots, \leq x_n$, and if $x_i = x_{i+1}$, $1 \leq i < n$, then $y_{i+1} \geq y_i$. We then see that there is always an optimal solution with modified y-coordinates in Y.

For any integer $t \leq n$, let $\mathcal{L}[t] = (P_1, \ldots, P_t)$ $(\mathcal{L} = \mathcal{L}[n])$ and let $\mathcal{L}'[t] = (P'_1, \ldots, P'_t)$, with $P'_i = (x_i, y'_i)$ and $y'_i \in Y$ denoting any modified list. We denote by C^i, $i \in \{1, \ldots, p\}$, a decreasing subsequence of $\mathcal{L}[t]$ and by S^j, $j \in \{1, \ldots, k\}$, a non-decreasing subsequence of $\mathcal{L}[t]$. For any subset $\tau \subset \mathcal{L}[t]$ of $\mathcal{L}[t]$, we denote by $\min(\tau)$ and $\max(\tau)$ the minimum and maximum y-coordinate of the elements in τ, respectively. In addition, by denoting $\mathcal{L}'[t] = \bigcup_{1 \leq i \leq p} C^i \cup \bigcup_{1 \leq j \leq k} S^j$, we mean that $\mathcal{L}'[t]$ is covered by p decreasing subsequences $C^1, \ldots, C^p$ and k non-decreasing subsequences $S^1, \ldots, S^k$. Let $\mathcal{T} = \{(A_p, B_k) = (a_1, \ldots, a_p, b_1, \ldots, b_k) \in Y^{p+k}\}$ be the set of vectors of dimension $(p + k)$ with coordinates in Y. We have $|\mathcal{T}| = O(n^{p+k})$. With any decomposition $\mathcal{L}'[t] = \bigcup_{1 \leq i \leq p} C^i \cup \bigcup_{1 \leq j \leq k} S^j$ of $\mathcal{L}[t]$, we associate the vector $(A_p, B_k) \in \mathcal{T}$ such that $\forall i \in \{1, \ldots, p\}$, $a_i = \min(C^i)$ and $\forall j \in \{1, \ldots, k\}$, $b_j = \max(S^j)$.

For any integer t such that $p + k \leq t \leq n$, and any fixed vector $(A_p, B_k) \in \mathcal{T}$, we consider the problem $P_t(A_p, B_k)$, defined as follows:

$$P_t(A_p, B_k) \begin{cases} \min \sum_{1 \leq h \leq t} |y_h - y'_h| \\ s.t.\ \mathcal{L}'[t] = C^1 \cup \ldots \cup C^p \cup S^1 \cup \ldots \cup S^k \\ a_i = \min(C^i),\ \forall i \in \{1, \ldots, p\} \\ b_j = \max(S^j),\ \forall j \in \{1, \ldots, k\} \end{cases}$$

where y_h is the y-coordinate of the hth element in $\mathcal{L}[t]$ and y'_h is the y-coordinate of the hth element in $\mathcal{L}'[t]$. Let $v(P_t(A_p, B_k))$ be the minimum value of $P_t(A_p, B_k)$. For any $i \in \{1, \ldots, p\}$ and $j \in \{1, \ldots, k\}$, let us define $f(i)$ and $g(j)$ as follows:

$$f(i) = |y_{t+1} - a_i| + \min_{\{u_i \geq a_i\}} v(P_t(a_1, \ldots, a_{i-1}, u_i, a_{i+1}, \ldots, a_p, b_1, \ldots, b_k))$$

$$g(j) = |y_{t+1} - b_j| + \min_{\{v_j \leq b_j\}} v(P_t(a_1, \ldots, a_p, b_1, \ldots, b_{j-1}, v_j, b_{j+1}, \ldots, b_k)).$$

Then, for any fixed vector $(A_p, B_k) \in \mathcal{T}$, the inductive relation between $P_t(A_p, B_k)$ and $P_{t+1}(A_p, B_k)$ can be expressed as follows:

$$v(P_{t+1}(A_p, B_k)) = \min\{\min_{1 \leq i \leq p} f(i),\ \min_{1 \leq j \leq k} g(j)\},$$

$$v(P_t(A_p, B_k)) = 0 \text{ if } t \leq p + k.$$

Then, the optimal solution value of $ICOL_{(p,k)}$ is given by $v(P_n) = \min_{(A_p, B_k) \in \mathcal{T}} v(P_n(A_p, B_k))$.

Let us evaluate the worst-case running time required to compute the optimum value $v(P_n)$ for a list $\mathcal{L}$ of n vertices. The values of $v(P_t(A_p, B_k))$ are stored in a table of size $O(n^{p+k+1})$ (n lines and $|\mathcal{T}| = O(n^{p+k})$ columns). For any fixed integer $t \leq n$ and any fixed vector $(A_p, B_k) \in \mathcal{T}$, the optimal solution of $P_{t+1}(A_p, B_k)$ is obtained from a solution of $P_t(A'_p, B'_k)$, $(A'_p, B'_k) \in \mathcal{T}$, computed at the previous step. For this, we merely need the minimum among $f(i)$, $1 \leq i \leq p$ and $g(j)$, $1 \leq j \leq k$, which requires $O(n^2)$ running time (since $(p + k) \leq n$). So, for any fixed integer $t \leq n$, computing all the tth line requires $O(n^{p+k+2})$ running time. Since the table contains n lines, the overall running time is bounded by $O(n^{p+k+3})$.

Proposition 16. *If p and k are constant, then $ICOL_{(p,k)}$ can be solved in polynomial time in permutation graphs.*

For $p = 0$, we get:

Corollary 17. *For any constant K, the inverse track assignment problem on K tracks under the midnight condition can be solved in polynomial time.*

6. Concluding remarks

In this paper, we introduced an inverse version of minimum graph coloring called the inverse chromatic number problem. We focused on two particular cases in interval and permutation graphs. For both cases, we described possible application frameworks and proposed some complexity results. We selected these two examples for two main reasons:

(1) Since the inverse chromatic number problem is as hard as the k-colorability problem, we first focused on classes of graphs where minimum coloring is polynomial. The results in interval graphs demonstrate that, even in this case, the inverse chromatic number problem may be intractable.
(2) These two examples illustrate inverse models that cannot directly be represented by usual graph problems. Instead of modifying the given graph, we modify the interval system or the lattice representation associated with the given graph so that the resulting graph has a chromatic number of K or less.

The inverse booking problem proved to be closely related to well-known scheduling problems. We presented improved hardness results in approximation and with a fixed number of interval sizes. Moreover, their presentation as inverse problems makes some new specific cases and generalizations relevant. In future studies, modifications of the interval system may be considered in which the interval length can be reduced at minimal cost, or intervals can be split into a fixed number of segments.

As mentioned in Section 2, the inverse booking problem examined here immediately suggests similar problems in other classes of intersection graphs like disk graphs, edge-intersection graphs of paths, and intersection graphs of rectangles (see, e.g., the berth allocation problem presented in Demange et al., 2015) or circular arc graphs (i.e., intersection graphs of arcs). Since these classes induce hard coloring problems (Garey et al., 1980; Golumbic & Jamison, 1985; Gräf et al., 1998; Lee & Leung, 1984), one may expect them to generate very hard versions of the inverse chromatic number problem.

To our knowledge, the inverse track assignment problem is new to the literature. We proved its tractability in polynomial time, however, the complexity remains open if the number of tracks is part of the problem instance. Here also, many generalizations can be made. Instead of changing the lattice representation of the graph, the interval system in the equivalent interval representation may be changed. Some modifications can be represented with the inverse model proposed here, while others induce alternative inverse models. Our model already considers the case of overlap graphs. Here also, because the minimum coloring problem is hard in this class (Garey et al., 1980), the related inverse problem is expected to be much harder than in permutation graphs.

The minimum inverse chromatic number problem may be studied in other classes of graphs using specific graph modification strategies. The inverse number problems may be considered for other combinatorial optimization problems.

Acknowledgements

This work was supported by the National Research Foundation of Korea Grant funded by the Korean Government (NRF-2014S1A3A2044046). M. Demange also acknowledges support by the French Agency for Research under the DEFI program TODO, ANR-09-EMER-010.

Appendix A

In this section, we prove that $I\text{Book}_{K,\rightarrow}$ is NP-hard if all interval lengths belong to the set $\{1, 2, 5, 7, 9, L, L+3, 8L+5\}$ where $L \in \mathbb{N}$ is a polynomial function that depends on n. For this purpose, we consider the problem `(2,2)-Sat`:

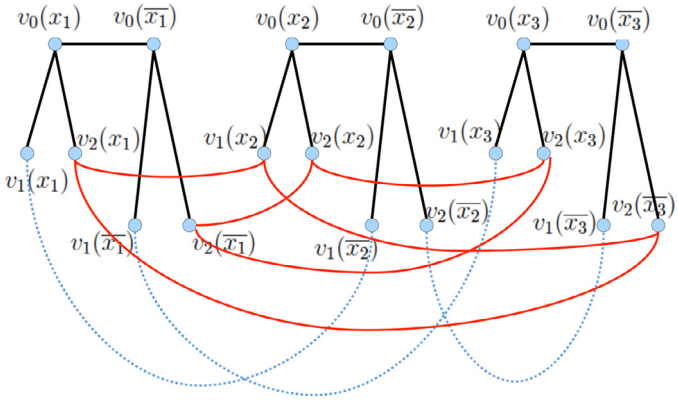

Fig. 4. The graph G_{I_0} associated with an instance I_0 of `(2,2)-Sat`.

`(2,2)-Sat`
Instance: Set X of variables, collection C of clauses over X such that each clause contains 2 or 3 literals, and every literal x and $\overline{x}$ appears exactly two times.
Question: Is there a satisfying truth assignment for C?

`(2,2)-Sat` is known to be NP-complete (Berman, Karpinski, & Scott, 2003). Our proof proceeds in two steps. Using a reduction from `(2,2)-Sat`, we first show that a particular case of Vertex Cover,[6] called Restricted Vertex Cover (`RVC` for short) is NP-complete. Then, we reduce this particular case of Vertex Cover to $I\text{Book}_{K,\rightarrow}$.

Let us first describe how we build a graph G_I from an instance I of `(2,2)-Sat`. This is the core of the reduction.

Consider an instance I of `(2,2)-Sat` with a set $X = \{x_1, \ldots, x_n\}$ of n variables, a set $X \cup \overline{X} = \cup_{1 \le i \le n}\{x_i, \overline{x_i}\}$ of $2n$ literals, and a collection $C = \{c_1, \ldots, c_{p+k}\}$ of clauses, where k is the number of 2-clauses and p the number of 3-clauses. For the simplicity of the proof, we suppose that the clauses are renumbered in such a way that $c_1, \ldots, c_k$ are 2-clauses and $c_{k+1}, \ldots, c_{p+k}$ are 3-clauses. As each literal appears exactly twice, we have $4n = 3p + 2k$.

We then construct a graph $G_I = (V, E)$ of order $6n$ as follows:

a. $V = V_0 \cup V_1$ with
   * $V_0 = \bigcup_{1 \le i \le n}\{v_0(x_i), v_0(\overline{x_i})\}$ and
   * $V_1 = \bigcup_{1 \le i \le n}\{v_1(x_i), v_2(x_i), v_1(\overline{x_i}), v_2(\overline{x_i})\}$.
b. $E = E' \cup E'' \cup E'''$, where
   * $E' = \bigcup_{1 \le i \le n}\left\{e'_{i,1}, e'_{i,2}, e'_{i,3}, e'_{i,4}, e'_{i,5}\right\}$, with $e'_{i,1} = (v_0(x_i)v_0(\overline{x_i}))$, $e'_{i,2} = (v_0(x_i)v_1(x_i))$, $e'_{i,3} = (v_0(x_i)v_2(x_i))$, $e'_{i,4} = (v_0(\overline{x_i})v_1(\overline{x_i}))$, and $e'_{i,5} = (v_0(\overline{x_i})v_2(\overline{x_i}))$.
   * E'' is a set of the edges with both extremities in V_1.
   * E''' is a set of the disjoint 3-cliques with three vertices in V_1 such that every vertex in V_1 is an extremity of either one edge of E'' or two edges of E'''.

Briefly speaking, we create for each variable $x_i \in X$ a truth-setting component consisting of two vertices $v_0(x_i)$ and $v_0(\overline{x_i})$, connected by an edge $(v_0(x_i)v_0(\overline{x_i}))$. For each literal $y \in X \cup \overline{X}$, we add two vertices $v_1(y)$ and $v_2(y)$ representing respectively the first and the second occurrence of the literal y. Each of these two vertices are connected to $v_0(y)$ by an edge.

For each clause $c_j \in C$, we create some edges. If two literals y and z are contained in a same clause c_j, we consider the related vertices $v_i(y)$ (with $i = 1$ if c_j is the first clause containing y and $i = 2$ otherwise) and $v_i(z)$, and we connect these two vertices by an edge. So, a 2-clause is represented by an edge and a 3-clause by a triangle.

Fig. 4 depicts the graph obtained from the instance I_0 of `(2,2)-Sat` defined by $X = \{x_1, x_2, x_3\}$ and $C = \{c_1, c_2, c_3, c_4, c_5\}$ where $c_1 = \overline{x_1} \vee$

[6] We remind that an instance of Vertex Cover is a graph $G = (V, E)$ and an integer M and the question is whether there is a subset $V' \subset V$ of size M or less such that for every edge $(uv) \in E$, $V' \cap \{u, v\} \neq \emptyset$.

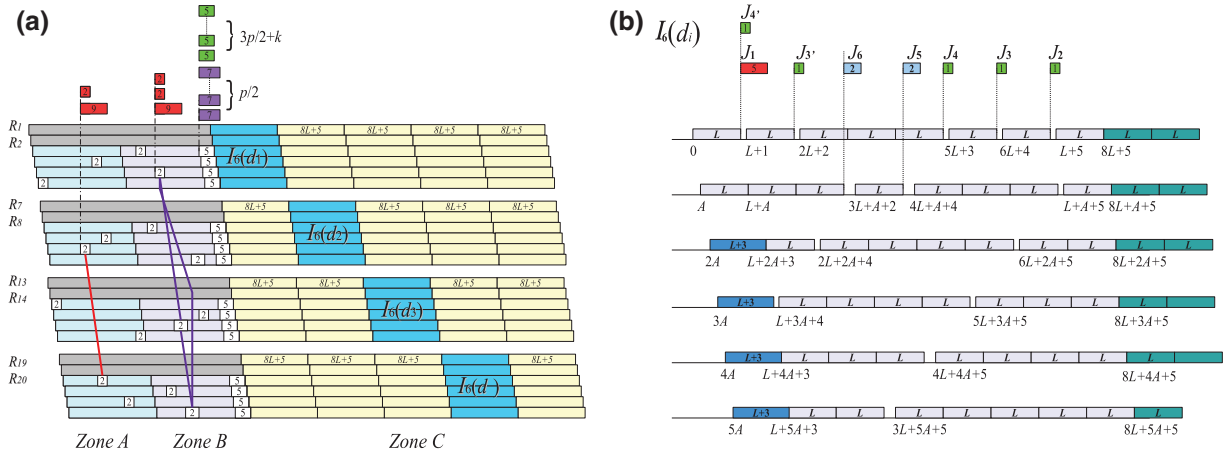

Fig. 5. (a) The whole instance $\mathcal{I}_{6n}$ with $n = 4$, $p = 2$, and $k = 5$ and (b) the sub-instance $\mathcal{I}_6(d_i)$.

x_3, $c_2 = x_1 \vee \overline{x_2}$, $c_3 = \overline{x_2} \vee \overline{x_3}$, $c_4 = x_1 \vee x_2 \vee \overline{x_3}$ and $c_5 = \overline{x_1} \vee x_2 \vee x_3$.

We define $\mathfrak{F}$ as the set $\{(I, G_I), I \in \mathfrak{I}\}$ where $\mathfrak{I}$ denotes the set of (2,2)-Sat-instances. Since the graph $G_I \in \mathfrak{F}$ is given together with the instance I, the membership to $\mathfrak{F}$ can be decided in polynomial time with respect to the size of I. Note also that this transformation can be completed in polynomial time.

We then denote by RVC (Restricted Vertex Cover) the restriction of Vertex Cover to the graphs G_I, $I \in \mathfrak{I}$:

RVC:

Instance: an instance $I \in \mathfrak{I}$ with n variables, k 2-clauses and p 3-clauses and the graph G_I.

Question: Does G_I contain a vertex cover of size at most $M = n + k + 2p$?

Lemma 18. *The RVC problem is NP-complete.*

Proof. We prove this result by using a reduction from (2,2)-Sat. The argument is very similar to the classical proof that Vertex Cover is NP-complete by a reduction from 3-SAT (see Garey & Johnson, 1979, p. 55). From an instance $I \in \mathfrak{I}$, we construct in polynomial time G_I as described previously. It is straightforward to verify that G_I has a vertex cover of size $n + k + 2p$ if and only if I is satisfiable. Since RVC is clearly in **NP** it completes the proof. □

Proposition 19. *IBook$_{K,\rightarrow}$ is* NP*-hard if all interval lengths belong to* $\{1, 2, 5, 7, 9, L, L + 3, 8L + 5\}$ *where* $L \in \mathbb{N}$ *is bounded by a polynomial function with respect to n.*

Proof. (Sketch) Starting from an instance of the RVC-problem, we construct an instance of *I*Book$_{K=6n,\rightarrow}$ composed of $6n$ parallel lines such that for any fixed $i \in \{1, \ldots, n\}$, the lines $\mathcal{R}_{6(i-1)+1}$, $\mathcal{R}_{6(i-1)+2}$, $\mathcal{R}_{6(i-1)+3}$, $\mathcal{R}_{6(i-1)+4}$, $\mathcal{R}_{6(i-1)+5}$, and $\mathcal{R}_{6(i-1)+6}$ are respectively associated with the vertices $v_0(x_i)$, $v_0(\overline{x_i})$, $v_1(x_i)$, $v_2(x_i)$, $v_1(\overline{x_i})$, and $v_2(\overline{x_i})$ of graph $G \in \mathfrak{F}$. The lines $\mathcal{R}_i$'s are loaded by some intervals of length $\{L, L + 3, 8L + 5\}$, as shown in Fig. 5, in such a way that the difference between the starting point of the leftmost interval in two lines is at least equal to a certain integer A (see Fig. 5). The value of A, which depends on n, will be fixed in such a way that exchanging two of these intervals of different lines costs more than Q.

These $6n$ lines are partitioned into three zones: zone $\mathcal{A}$, zone $\mathcal{B}$ and zone $\mathcal{C}$:

- Zone $\mathcal{A}$ represents the edges associated with the 2-clauses. It contains k intervals of length L. If the line is associated with a vertex $v_i(\ell_j)$, which appears in a 2-clause, then we add a blank of length 2 in this zone.
- Zone $\mathcal{B}$ represents the 3-cliques associated with the 3-clauses. It contains p intervals of length L and, if the line is associated with an extremity of a 3-clique, we add a blank of size 2 in this zone. Between the zone $\mathcal{B}$ and the last zone $\mathcal{C}$, we have a blank of size 5.
- Zone $\mathcal{C}$ represents the structures $S(x_i)$, $i \in \{1, \ldots, n\}$ (see Fig. 5(a)). Each line of zone $\mathcal{C}$ is loaded first by n intervals of length $8L + 5$ and then by Q intervals of length L. The ith intervals of lines $\bigcup_{r=1}^{6}\{\mathcal{R}_{6(i-1)+r}\}$ for any $i \in \{1, \ldots, n\}$ are replaced by the same sub-instance given in Fig. 5(b).

Then, each line of $\bigcup_{i=1}^{n}\{\mathcal{R}_{6(i-1)+3}, \mathcal{R}_{6(i-1)+4}, \mathcal{R}_{6(i-1)+5}, \mathcal{R}_{6(i-1)+6}\}$ contains a blank of size 2 in zone $\mathcal{A}$ (respectively, in zone $\mathcal{B}$) if the line corresponds to a literal composing a 2-clause (respectively, 3-clause), and a blank of size 5 in zone $\mathcal{B}$. For the blanks in zone $\mathcal{A}$, we add two intervals of lengths 2 and 5, respectively. In zone $\mathcal{B}$, we add three intervals, two of length 2 and the other of length 5 for each of three blanks of length 2. We also add $\frac{3p}{2} + k$ intervals of length 5 and $\frac{p}{2}$ intervals of length 7. We then observe that the blanks of size 2 in zones $\mathcal{A}$ and $\mathcal{B}$ are located alternatively (due to the structure $S(x_i)$, no line contains two blanks of size 2 in these zones).

In $\mathcal{I}_6(d_i)$, the interval J_1 should be placed on line $\mathcal{R}_1$ or line $\mathcal{R}_2$ in order not to exceed the translation cost Q. For such solutions, the other small intervals will be inserted, according to the position of J_1, in the blanks either on lines $\mathcal{R}_2$, $\mathcal{R}_3$, and $\mathcal{R}_4$ (Solution 1) or on lines $\mathcal{R}_2$, $\mathcal{R}_5$, and $\mathcal{R}_6$ (Solution 2). The blanks not used here will be used later to make spaces to insert the intervals of lengths 5 and 7, added in zones $\mathcal{A}$ and $\mathcal{B}$. This instance is constructed in such a way that all intervals can be legally assigned on K lines with a translation cost less than Q if and only if the vertices corresponding to the lines to which we assign the intervals of length 2 that we added in zones $\mathcal{A}$ and $\mathcal{B}$ constitute a vertex cover of size $M = n + 2p + k$ of G if and only if there is a truth assignment which satisfies all clauses of the (2,2)-SAT instance. □

References

Ahuja, R. K., & Orlin, J. B. (2001). Inverse optimization. *Operations Research*, *49*(5), 771–783.

Baptiste, P. (2000). Scheduling equal-length jobs on identical parallel machines. *Discrete Applied Mathematics*, *103*(1–3), 21–32.

Baptiste, P., & Brucker, P. (2004). Scheduling equal processing time jobs. In J. Y.-T. Leung (Ed.), *Handbook of scheduling—Algorithms, models and performance analysis* (pp. 14-1–14-37). Chapman & Hall/CRC (Chapter 14).

Bazgan, C., Bentz, C., Picouleau, C., & Ries, B. (2015). Blockers for the stability number and the chromatic number. *Graphs and Combinatorics*, *31*(1), 73–90.

Berman, P., Karpinski, M., & Scott, A. D. (2003). *Approximation hardness of short symmetric instances of max-3sat*. ECCC Technical Report (TR03–049).

Brucker, P. (1995). *Scheduling algorithms*. Berlin: Springer-Verlag.

Brucker, P., & Kravchenko, S. A. (2005). *Scheduling jobs with release times on parallel machines to minimize total tardiness*. Technical Report Reihe P Preprints, Heft 258. Osnabrcker Schriften zur Mathematik, Fachbereich Mathematik/Informatik, Universitt Osnabrck.

Brucker, P., & Nordmann, L. (1994). The *k*-track assignment problem. *Computing*, *52*(2), 97–122.

Burkard, R. E., Pleschiutschnig, C., & Zhang, J. (2004). Inverse median problems. *Discrete Optimization*, *1*(1), 23–39.
Burton, D., & Toint, Ph. L. (1992). On an instance of the inverse shortest paths problem. *Mathematical Programming*, *53*, 45–61.
Chung, Y., & Demange, M. (2008). The 0-1 inverse maximum stable set problem. *Discrete Applied Mathematics*, *156*(13), 2501–2516.
Chung, Y., & Demange, M. (2012). Some inverse traveling salesman problems. *4OR*, *10*(2), 193–209.
Demange, M., Ekim, T., & de Werra, D. (2009). A tutorial on the use of graph coloring for some problems in robotics. *European Journal of Operational Research*, *192*(1), 41–55.
Demange, M., Ekim, T., Ries, B., & Tanasescu, C. (2015). On some applications of the selective graph coloring problem. *European Journal of Operational Research*, *240*, 307–314.
Demange, M., & Monnot, J. (2010). An introduction to inverse combinatorial problems. In V. T. Paschos (Ed.), *Paradigms of combinatorial optimization (problems and new approaches)* (pp. 547–586). ISTE–WILEY.
Demange, M., Monnot, J., Petrica, P., & Ries, B. (2014). On the complexity of the selective graph coloring problem in some special classes of graphs. *Theoretical Computer Science*, *540-541*, 89–102.
de Werra, D. (1996). Extensions of coloring models for scheduling purposes. *European Journal of Operational Research*, *92*, 474–492.
Eisenblätter, A., Grötsche, M., & Koster, A. M. C. A. (2002). Frequency planning and ramifications of coloring. *Discussiones Mathematicae Graph Theory*, *22*, 51–88.
Frank, A. (1980). On chain and antichain families of a partially ordered set. *Journal of Combinatorial Theory, Series B*, *29*, 176–184.
Galiniera, P., & Hertz, A. (2006). A survey of local search methods for graph coloring. *Computers & Operations Research*, *33*, 2547–2562.
Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability—A guide to the theory of NP-completeness*. San Francisco: Freeman.
Garey, M. R., Johnson, D. S., Miller, G. L., & Papadimitriou, C. H. (1980). The complexity of coloring circular arcs and chords. *SIAM Journal on Algebraic and Discrete Methods*, *1*(2), 216–227.
Garey, M. R., Tarjan, R. E., & Wilfong, G. T. (1988). One-processor scheduling with symmetric earliness and tardiness penalties. *Mathematics of Operations Research*, *13*, 330–348.
Golumbic, M. C. (1980). *Algorithmic graph theory and perfect graphs*. New York: Academic press.
Golumbic, M. C., & Jamison, R. E. (1985). The edge intersection graphs of paths in a tree. *Journal of Combinatiorial Theory, Series B*, *38*, 8–22.
Gräf, A., Stumpf, M., & Weißenfels, G. (1998). On coloring unit disk graphs. *Algorithmica*, *20*, 277–293.
Greene, C. (1976). Some partitions associated with a partially ordered set. *Journal of Combinatorial Theory. Series A*, *20*(1), 69–79.
Greene, C., & Kleitman, D. J. (1976). The structure of sperner k-families. *Journal of Combinatorial Theory. Series A*, *20*(1), 41–68.
Hertz, A., & de Werra, D. (1990). The tabu search metaheuristic: How we used it. *Annals of Mathematics and Artificial Intelligence*, *1*, 111–121.
Rinnooy Kan, A. H. G. (1976). *Machine scheduling problem: Classification, complexity and computation*. The Hague: Nijhoff.
Lee, D. T., & Leung, J. Y.-T. (1984). On the 2-dimensional channel assignment problem. *IEEE Transactions on Computers*, *33*, 2–6.
Liu, M.-H., & Ubhaya, V. A. (1997). Integer isotone optimization. *SIAM Journal on Optimization*, *7*(4), 1152–1159.
Marmion, M.-E., Blot, A., Jourdan, L., & Dhaenens, C. (2013). Neutrality in the graph coloring problem. *Learning and Intelligent Optimization. Lecture notes in computer Science 2013* (pp. 125–130).
Müller-Hannemann, M., & Sonnikow, A. (2009). Non-approximability of just-in-time scheduling. *Journal of Scheduling*, *12*, 555–562.
Schrijver, A. (2003). *Combinatorial optimization: Polyhedra and efficiency*. Springer.
Verma, S., & Dessouky, D. (1998). Single-machine scheduling of unit-time jobs with earliness and tardiness penalties. *Mathematics of Operations Research*, *23*, 930–943.
Wan, L., & Yuan, J. (2013). Single-machine scheduling to minimize the total earliness and tardiness is strongly np-hard. *Operations Research Letters*, *41*, 363–365.
Yang, C., Zhang, J., & Ma, Z. (1997). Inverse maximum flow and minimum cut problems. *Optimization*, *40*, 147–170.
Yannakakis, M. (1978). Node-and edge-deletion np-complete problems. In *Proceedings of the tenth annual ACM symposium on theory of computing (STOC'78)* (pp. 253–264). New York, NY: ACM.
Yannakakis, M., & Gavril, F. (1987). The maximum k-colorable subgraph problem for chordal graphs. *Information Processing Letters*, *24*(2), 133–137.
Yáñez, J., & Ramírez, J. (2003). The robust coloring problem. *European Journal of Operational Research*, *148*, 546–558.
Zang, H., Jue, J. P., & Mukherjee, B. (2000). A review of routing and wavelength assignment approaches for wavelength routed optical wdm networks. *Optical Networks Magazine*, *1*, 47–60.
Zhang, J., & Cai, M. C. (1998). Inverse problem of minimum cuts. *Mathematical Methods of Operations Research*, *47*(1), 51–58.
Zhang, J., Yang, X. G., & Cai, M. C. (1999). The complexity analysis of the inverse center location problem. *Journal of Global Optimization*, *15*(2), 213–218.