Discrete Applied Mathematics 157 (2009) 3615–3624

Contents lists available at ScienceDirect

Discrete Applied Mathematics

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

About equivalent interval colorings of weighted graphs

Mathieu Bouchard [a], Mirjana Čangalović [b], Alain Hertz [a,*]

[a] École Polytechnique and GERAD, Montréal, Canada

[b] Faculty of Organizational Sciences, University of Belgrade, Belgrade, Serbia

ARTICLE INFO

Article history:
Received 20 December 2007
Received in revised form 8 April 2009
Accepted 19 April 2009
Available online 15 May 2009

Keywords:
Interval coloring problem
Weighted graphs
Equivalent colorings

ABSTRACT

Given a graph $G = (V, E)$ with strictly positive integer weights ω_i on the vertices $i \in V$, a k-interval coloring of G is a function I that assigns an interval $I(i) \subseteq \{1, \ldots, k\}$ of ω_i consecutive integers (called colors) to each vertex $i \in V$. If two adjacent vertices x and y have common colors, i.e. $I(i) \cap I(j) \neq \emptyset$ for an edge $[i, j]$ in G, then the edge $[i, j]$ is said *conflicting*. A k-interval coloring without conflicting edges is said *legal*. The interval coloring problem (ICP) is to determine the smallest integer k, called *interval chromatic number* of G and denoted $\chi_{int}(G)$, such that there exists a legal k-interval coloring of G. For a fixed integer k, the k-interval graph coloring problem (k-ICP) is to determine a k-interval coloring of G with a minimum number of conflicting edges. The ICP and k-ICP generalize *classical vertex coloring problems* where a single color has to be assigned to each vertex (i.e., $\omega_i = 1$ for all vertices $i \in V$).

Two k-interval colorings I_1 and I_2 are said *equivalent* if there is a permutation π of the integers $1, \ldots, k$ such that $\ell \in I_1(i)$ if and only if $\pi(\ell) \in I_2(i)$ for all vertices $i \in V$. As for classical vertex coloring, the efficiency of algorithms that solve the ICP or the k-ICP can be increased by avoiding considering equivalent k-interval colorings, assuming that they can be identified very quickly. To this purpose, we define and prove a necessary and sufficient condition for the equivalence of two k-interval colorings. We then show how a simple tabu search algorithm for the k-ICP can possibly be improved by forbidding the visit of equivalent solutions.

1. Introduction

Given a graph $G = (V, E)$ with vertex set V and edge set E, the classical graph coloring problem is to assign a color to each vertex so that no two adjacent vertices have the same color and the total number of different colors is minimized. This is one of the most studied NP-hard combinatorial optimization problems [9] with various practical applications [16]. A number of different variations and generalizations of the classical graph coloring problem arise when modeling and solving real-life problems. For example, the number of colors assigned to a vertex can be more than one, and conditions can be imposed on the colors assigned to the vertices.

One such generalization is the so-called interval coloring problem of a vertex-weighted graph [12] where a strictly positive integer weight ω_i is associated with each vertex $i \in V$, and an interval of ω_i consecutive integers must be assigned to each vertex $i \in V$ such that the intervals assigned to adjacent vertices are disjoint. More formally, let $G = (V, E)$ be an undirected graph with vertex set V and edge set E, and with strictly positive integer weights ω_i on the vertices $i \in V$. A k-interval coloring of G is a function I that assigns an interval $I(i) \subseteq \{1, \ldots, k\}$ of ω_i consecutive integers (called colors) to each vertex $i \in V$. Without loss of generality, we will always assume that a k-interval coloring of G uses all colors in

* Corresponding author. Fax: +1 514 340 5665.
E-mail addresses: Mathieu.Bouchard@gerad.ca (M. Bouchard), canga@fon.bg.ac.yu (M. Čangalović), Alain.Hertz@gerad.ca (A. Hertz).

0166-218X/$ – see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2009.04.015

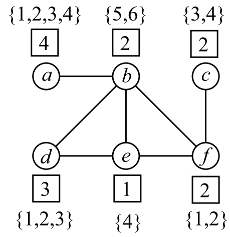

Fig. 1. A legal 6-interval coloring I of a graph G.

$\{1, \ldots, k\}$. If two adjacent vertices x and y have common colors, i.e. $I(i) \cap I(j) \neq \emptyset$ for an edge $[i, j]$ in G, then the edge $[i, j]$ is said *conflicting*. A k-interval coloring without conflicting edges is said *legal*. The interval coloring problem (ICP) is to determine the smallest integer k, called *interval chromatic number* of G and denoted $\chi_{int}(G)$, such that there exists a legal k-interval coloring of G. The special case with $\omega_i = 1$ for all vertices $i \in V$ is equivalent to the classical graph coloring problem, and a k-interval coloring is simply called k-coloring in this case. For illustration, a legal 6-interval coloring of a graph G is represented in Fig. 1, where the numbers into boxes correspond to weights on vertices. Note that $\chi_{int}(G) = 6$ for this graph since the total weight of the edge $[a, b]$ or of the triangle with vertices b, d, e is equal to 6.

For a fixed integer k, the k-interval graph coloring problem (k-ICP) is to determine a k-interval coloring of G with a minimum number of conflicting edges. If the minimum value is zero, this means that G admits a legal k-interval coloring, hence $\chi_{int}(G) \leq k$.

The ICP has a fairly long history dating back, at least to the 1970s. For example, Stockmeyer showed in 1976 that the interval-coloring problem is NP-hard, even when restricted to interval graphs and vertex weights in $\{1, 2\}$ (see problem SR2 in [9]). In 1976, Punter [18] has formulated and solved a school timetabling problem with non-preemptive multiple period lessons using an interval coloring model. Another early application of the interval coloring problem was in the compile-time memory-allocation problem [6].

While the ICP is NP-hard, it can be solved in polynomial time for special classes of graphs. For example, if G is a clique then, $\chi_{int}(G)$ is equal to $\sum_{i \in V} \omega_i$, while for a bipartite graph G, we have $\chi_{int}(G) = \max_{[i,j] \in E}\{\omega_i + \omega_j\}$. More general graphs G for which $\chi_{int}(G)$ can be computed in polynomial time are studied in [12,5].

Upper bounds on the interval chromatic number $\chi_{int}(G)$ are studied in [5], while general upper and lower bounds on $\chi_{int}(G)$ are given in [15] when vertices possibly have forbidden colors. An exact algorithm for the ICP is proposed in [3] and used to solve a real-life timetabling problem with multiple period lessons. Approximation algorithms are known for special classes of graphs such as interval graphs [2,10] or chordal graphs [17]. Heuristic algorithms for the ICP are proposed for example in [4,1].

In the classical graph coloring problem, every k-coloring is equivalent, up to a permutation of the colors, to $k! - 1$ other k-colorings. In order to increase the efficiency of graph coloring algorithms, it is important to avoid visiting equivalent k-colorings when exploring the search space. A solution to the classical k-coloring problem is in fact a partition of the vertex set into k subsets called *color classes*, and the total number of non equivalent k-colorings is equal to the number of possible partitions of the vertex set into k subsets. Such considerations have inspired many researchers, including Galinier and Hao [7] who have designed a very effective genetic algorithm for the classical graph coloring problem in which new k-colorings are created from a population of k-colorings by combining color classes of two parents instead of copying color assignments. Also, the most effective local search algorithms for classical graph coloring generate neighbor k-colorings by moving a vertex from a color class to another [8].

In the next section, we generalize the above equivalence relation to k-interval colorings. We then prove a necessary and sufficient condition for the equivalence of two k-interval colorings. Such a condition makes it easy to recognize equivalent k-interval colorings. We will use the following terminology. A *clique* in a graph $G = (V, E)$ is a subset $W \subseteq V$ of pairwise adjacent vertices. An *interval graph* [13,12] is the intersection graph of a set of intervals. It has one vertex for each interval in the set, and an edge between every pair of vertices corresponding to intervals that intersect. Subsets $C_1, \ldots, C_p$ of V define a *cover* of V if $\bigcup_{i=1}^{p} C_i = V$. Moreover, if the sets C_i of the cover are mutually disjoint, they define a *partition* of V.

2. Equivalent *k*-interval colorings

Intuitively, two k-interval colorings are equivalent if one can be obtained from the other by permuting the colors $1, \ldots, k$. More formally, the equivalence of k-interval colorings can be defined as follows.

Definition 1. Two k-interval colorings I_1 and I_2 are said equivalent if there is a permutation π of the integers $1, \ldots, k$ such that $\ell \in I_1(i)$ if and only if $\pi(\ell) \in I_2(i)$ for all vertices $i \in V$.

Note that given a k-interval coloring I of graph $G = (V, E)$ and a permutation π of the integers $1, \ldots, k$, it may happen that $\bigcup_{\ell \in I(i)} \pi(\ell)$ is not an interval for some vertex $i \in V$. For example, considering the graph of Fig. 1, permutation π with $\pi(1) = 6, \pi(6) = 1$ and $\pi(\ell) = \ell$ for $\ell \neq 1, 6$ gives colors 2, 3, 4 and 6 to vertex a, and this is not an interval.

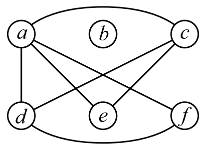

Fig. 2. The interval graph $H_{G,I}$ associated with the k-interval coloring of Fig. 1.

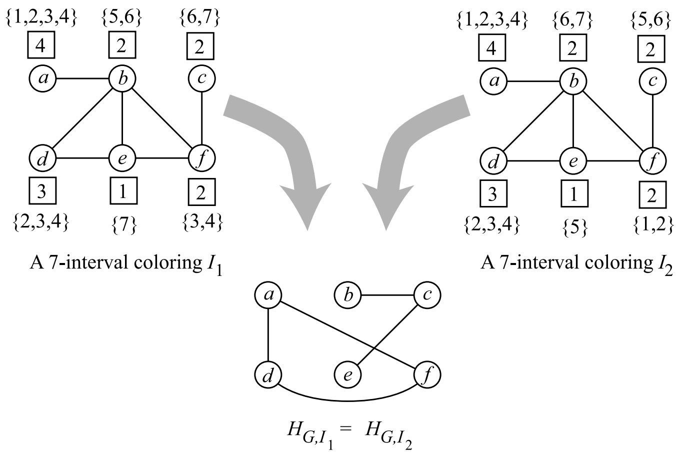

Fig. 3. Two non-equivalent 7-interval colorings of a graph G, and their identical associated interval graph.

For the classical graph coloring problem (i.e., when $\omega_i = 1$ for all $i \in V$), Definition 1 means that two k-colorings are equivalent if their corresponding partition into color classes are identical. A concept similar to color classes in the case of interval coloring is what we call *interval color classes*, with the following formal definition.

Definition 2. Given a k-interval coloring I of a graph $G = (V, E)$, a subset $W \subseteq V$ of vertices is an interval color class for I if

(a) $\bigcap_{i \in W} I(i) \neq \emptyset$, and
(b) $\bigcap_{i \in W} I(i) \cap I(j) = \emptyset$ for all $j \notin W$.

The above definition can be interpreted in terms of graphs. Indeed, given an k-interval coloring I of a graph $G = (V, E)$, let $H_{G,I}$ be the interval graph with vertex set V and where two vertices i and j are linked by an edge if and only if $I(i) \cap I(j) \neq \emptyset$. Then, the interval color classes for I correspond to the maximal cliques in $H_{G,I}$. For example, the graph of Fig. 2 is the interval graph $H_{G,I}$ associated with the k-interval coloring of Fig. 1. It contains 4 maximal cliques (i.e. interval color classes), namely $W_1 = \{a, c, d\}$, $W_2 = \{a, c, e\}$, $W_3 = \{a, d, f\}$ and $W_4 = \{b\}$.

When $\omega_i = 1$ for all $i \in V$ (i.e., for the classical graph coloring problem), the interval graph $H_{G,I}$ is made of vertex-disjoint cliques, each one corresponding to a color class. Observe that the color classes in classical graph coloring induce a partition of the vertex set, while the interval color classes for a k-interval coloring induce a cover of the vertex set, which means that some vertices possibly belong to several interval color classes. In the example of Fig. 2, vertex a belongs to three different interval color classes, and vertex c belongs to two of them.

Since two k-colorings in classical graph coloring are equivalent if and only if they induce the same partition of the vertex set into color classes, it is tempting to think that two k-interval colorings I_1 and I_2 of a graph G are equivalent if and only if they have exactly the same interval color classes, i.e. if their associated interval graphs H_{G,I_1} and H_{G,I_2} are equal. Fig. 3 illustrates with an example that such a statement is not correct. Indeed, the two 7-interval colorings I_1 and I_2 on this figure have the same associated interval graph. However, if I_1 and I_2 were equivalent, then $I_2(f) = \{1, 2\} = \{\pi(3), \pi(4)\}$, which means that $\{1, 2\} \subset I_2(d)$, a contradiction.

Hence, in order to determine whether two k-interval colorings of a graph $G = (V, E)$ are equivalent, it is not sufficient to compare their corresponding cover of V with interval color classes. For a k-interval coloring I of G, let us associate a weight $|I(i) \cap I(j)|$ to each edge $[i, j]$ in the interval graph $H_{G,I}$. The following theorem states that two k-interval colorings I_1 and I_2 of a graph G are equivalent if and only if the corresponding weighted graphs H_{G,I_1} and H_{G,I_2} are identical (i.e., they have the same edge set and the same weights on the edges). For example, the two weighted interval graphs associated with the 7-interval colorings of Fig. 3 are represented in Fig. 4. Since the weight of the edge $[d, f]$ is 2 in H_{G,I_1} and 1 in H_{G,I_2}, the two 7-interval colorings are not equivalent.

Theorem 1. *Two k-interval colorings I_1 and I_2 of a graph $G = (V, E)$ are equivalent if and only if*

$$|I_1(i) \cap I_1(j)| = |I_2(i) \cap I_2(j)| \quad \text{for all } i, j \text{ in } V. \tag{1}$$

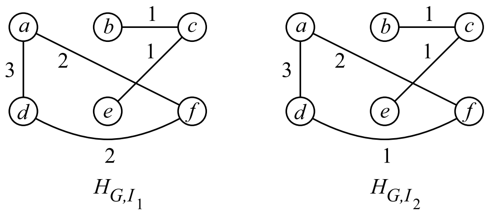

Fig. 4. The two weighted interval graphs associated with the 7-interval colorings of Fig. 3.

We first prove the following Lemma about k-interval colorings satisfying Property (1) of Theorem 1. For a subset $W \subseteq V$ of vertices and a k-interval coloring I, we denote $I(W) = \bigcap_{i \in W} I(i)$.

Lemma 1. *Let I_1 and I_2 be two k-interval colorings of a graph $G = (V, E)$ satisfying Property (1), and let $W \subseteq V$ be a subset of vertices. Then*

(a) $|I_1(W)| = |I_2(W)|$, *and*
(b) *W is an interval color class for I_1 if and only if it is an interval color class for I_2.*

Proof. We first prove (a). If W contains a single vertex i (i.e. $W = \{i\}$), then $\omega_i = |I_1(W)| = |I_2(W)|$. So assume $|W| \geq 2$.

- If $|I_1(W)| = 0$ then there exist at least two vertices i and j in W such that $I_1(i) \cap I_1(j) = \emptyset$. By Property (1), we then have $|I_2(i) \cap I_2(j)| = 0$, which means that $|I_2(W)| = 0$.
- If $|I_1(W)| > 0$, then $I_1(W)$ is an integer interval (since $I_1(i)$ is an integer interval for all $i \in V$), and there are at least two vertices u and v in W such that $I_1(u) \cap I_1(v) = I_1(W)$. By Property (1), we have $|I_2(u) \cap I_2(v)| = |I_1(W)|$, which means that $|I_2(W)| \leq |I_1(W)|$ since $I_2(W) \subseteq I_2(u) \cap I_2(v)$.

In all cases, we have $|I_2(W)| \leq |I_1(W)|$. By permuting the roles of I_1 and I_2, the same proof gives $|I_1(W)| \leq |I_2(W)|$. Hence, $|I_2(W)| = |I_1(W)|$.

To prove (b), assume that W is an interval color class for one of the two k-interval colorings, say I_1. Then, by definition, we have $I_1(W) \neq \emptyset$, and it follows from (a) that $I_2(W) \neq \emptyset$. If there exists a vertex $u \notin W$ such that $I_2(W \cup \{u\}) \neq \emptyset$, then we know from (a) that $I_1(W \cup \{u\}) \neq \emptyset$, which means that W is not an interval color class for I_1, a contradiction. In summary, $I_2(W) \neq \emptyset$ and $I_2(W \cup \{u\}) = \emptyset$ for all $u \notin W$, which means that W is also an interval color class for I_2. □

We now prove that Property (1) is a necessary and sufficient condition for the equivalence of two k-interval colorings.

Proof of Theorem 1. Let I_1 and I_2 be two equivalent k-interval colorings of a graph $G = (V, E)$. By Definition 1, there exists a permutation π of the integers $1, \ldots, k$ such that $\ell \in I_1(i)$ if and only if $\pi(\ell) \in I_2(i)$ for all vertices $i \in V$. Hence, for every i and j in V and for every $\ell \in \{1, \ldots, k\}$ we have $\ell \in I_1(i) \cap I_1(j)$ if and only if $\pi(\ell) \in I_2(i) \cap I_2(j)$, which means that $|I_1(i) \cap I_1(j)| = |I_2(i) \cap I_2(j)|$ for all i and j in V. Property (1) is therefore a necessary condition for the equivalence of two k-interval colorings.

We now prove that Property (1) is also a sufficient condition. The proof is by induction on the number k of colors used in I_1 and I_2. For $k = 1$, we have $I_1(i) = I_2(i) = \{1\}$ for all vertices $i \in V$ (since a k-interval coloring uses all colors in $\{1, \ldots, k\}$). Hence, permutation π with $\pi(1) = 1$ defines the equivalence between I_1 and I_2.

So, assume that $k > 1$ and Property (1) is a sufficient condition for the equivalence of two ℓ-interval colorings for all $\ell = 1, \ldots, k-1$. Consider any interval color class W for I_1. We know from Lemma 1 that $|I_1(W)| = |I_2(W)|$ and W is also an interval color class for I_2. So let π_1 be a bijective mapping from $I_1(W)$ to $I_2(W)$ (i.e., $\bigcup_{\ell \in I_1(W)} \pi_1(\ell) = I_2(W)$). For all vertices $i \in V$ and $r = 1, 2$ define

$$I'_r(i) = \begin{cases} I_r(i) - I_r(W) & \text{if } i \in W \\ I_r(i) & \text{if } i \in V - W \end{cases}$$

and

$$\omega'_i = \begin{cases} \omega_i - |I_1(W)| & \text{if } i \in W \\ \omega_i & \text{if } i \in V - W. \end{cases}$$

Since W is an interval color class for I_r ($r = 1, 2$), we have $\bigcup_{i \in V} I'_r(i) = \{1, \ldots, k\} - I_r(W)$ and $|I'_r(i)| = \omega'_i$ for all $i \in V$ and $r = 1, 2$. Note that $I'_r(i)$ is not necessarily an interval. Indeed, if the smallest color in $I_r(W)$ is strictly larger than the smallest color in $I_r(i)$ while the largest color in $I_r(W)$ is strictly smaller than the largest color in $I_r(i)$, then $I'_r(i)$ is the union of two integer intervals. So, let f_r be a function that relabels the colors in $\{1, \ldots, k\} - I_r(W)$ from 1 to $k - |I_r(W)|$ so that $f_r(i) < f_r(j)$ if and only if $i < j$ and define $I''_r(i) = \bigcup_{\ell \in I'_r(i)} f_r(\ell)$. The sets $I''_r(i)$ are integer intervals for all $i \in V$ and $r = 1, 2$. Note that if $I_r(i) = I_r(W)$ for a vertex $i \in W$, then $\omega'_i = 0$ and $I''_r(i)$ is empty.

Let $G' = (V', E')$ be the weighted graph obtained from G by removing all vertices with $\omega'_i = 0$, and by assigning weight ω'_i to all vertices $i \in V'$. By denoting $k' = k - |I_1(W)| = k - |I_2(W)|$, we have shown that I''_1 and I''_2 are two k'-interval colorings of G' with $k' < k$. In order to use the induction hypothesis, we now show that I''_1 and I''_2 satisfy Property (1).

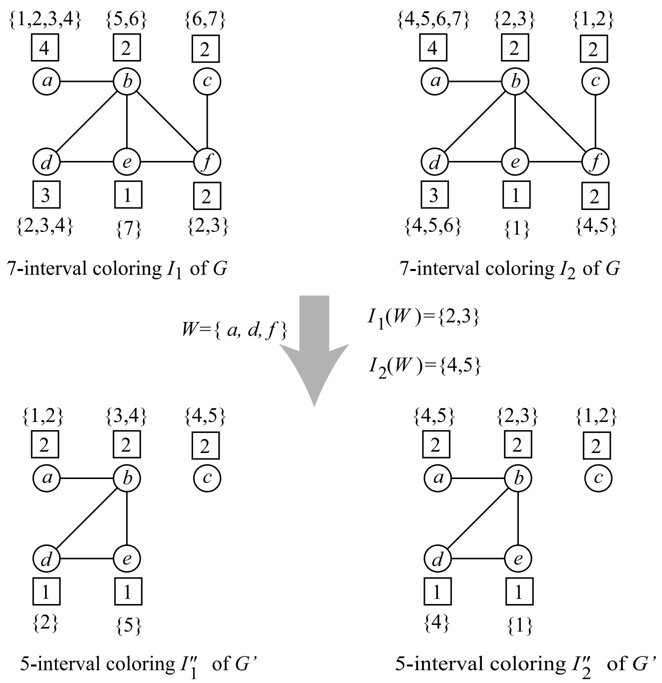

Fig. 5. Illustration of the proof of Theorem 1.

- If i and j are two vertices in $V' \cap W$, then $I_r(W) \subseteq I_r(i) \cap I_r(j)$ for $r = 1, 2$, which means that $|I''_r(i) \cap I''_r(j)| = |I_r(i) \cap I_r(j)| - |I_r(W)|$. Since $|I_1(i) \cap I_1(j)| = |I_2(i) \cap I_2(j)|$ and $|I_1(W)| = |I_2(W)|$, we have $|I''_1(i) \cap I''_1(j)| = |I''_2(i) \cap I''_2(j)|$.
- If i and j are two vertices in V' with at least one not in W, then $I_r(W) \cap I_r(i) \cap I_r(j) = \emptyset$ for $r = 1, 2$, which means that $|I''_r(i) \cap I''_r(j)| = |I_r(i) \cap I_r(j)|$. Since $|I_1(i) \cap I_1(j)| = |I_2(i) \cap I_2(j)|$, we have $|I''_1(i) \cap I''_1(j)| = |I''_2(i) \cap I''_2(j)|$.

By induction hypothesis, we know that there exists a permutation π_2 of the colors in $\{1, \ldots, k'\}$ such that $\ell \in I''_1(i)$ if and only if $\pi_2(\ell) \in I''_2(i)$ for all vertices $i \in V'$. Consider finally permutation π of the colors in $1, \ldots, k$ such that

$$\pi(\ell) = \begin{cases} \pi_1(\ell) & \text{if } \ell \in I_1(W) \\ f_2^{-1}(\pi_2(f_1(\ell))) & \text{if } \ell \in \{1, \ldots, k\} - I_1(W). \end{cases}$$

For a vertex $i \in V$ and a color $\ell \in I_1(i)$, we have proved that

- if $\ell \in I_1(W)$, then $\pi(\ell) = \pi_1(\ell) \in I_2(W) \subseteq I_2(i)$
- if $\ell \notin I_1(W)$, then $f_1(\ell) \in I''_1(i)$. Since $\pi_2(f_1(\ell)) \in I''_2(i)$, we have $\pi(\ell) = f_2^{-1}(\pi_2(f_1(\ell))) \in I_2(i)$.

In summary, we have $\ell \in I_1(i)$ if and only if $\pi(\ell) \in I_2(i)$, which proves that I_1 and I_2 are equivalent. □

An illustration of the above construction of permutation π for two equivalent k-interval colorings is given in Fig. 5. Vertex set $W = \{a, d, f\}$ is an interval color class for both 7-interval colorings I_1 and I_2 of G. We have $I_1(W) = \{2, 3\}$ and $I_2(W) = \{4, 5\}$. We can therefore consider π_1 such that $\pi_1(2) = 4$ and $\pi_1(3) = 5$. Hence, $f_1(1) = 1, f_1(4) = 2, f_1(5) = 3, f_1(6) = 4, f_1(7) = 5$ and $f_2(1) = 1, f_2(2) = 2, f_2(3) = 3, f_2(6) = 4, f_2(7) = 5$. Since $\omega'_f = 0$, vertex f does not belong to G'. All vertices in G' have the same weight as in G, except a and d for which there is reduction of two units. While $I'_1(a) = \{1, 4\}$ is not an interval, $I''_1(a) = \{f_1(1), f_1(4)\} = \{1, 2\}$. The two 5-interval colorings I''_1 and I''_2 of G' are equivalent, which can be observed with permutation π_2 such that $\pi_2(1) = 5, \pi_2(2) = 4, \pi_2(3) = 3, \pi_2(4) = 2$ and $\pi_2(5) = 1$. A proof of the equivalence of I_1 and I_2 in G is provided by permutation π with $\pi(1) = 7, \pi(2) = 4, \pi(3) = 5, \pi(4) = 6, \pi(5) = 3, \pi(6) = 2$ and $\pi(7) = 1$. For example, $\pi(4) = f_2^{-1}(\pi_2(f_1(4))) = f_2^{-1}(\pi_2(2)) = f_2^{-1}(4) = 6$. The weighted interval graph associated with the two 7-interval colorings of Fig. 5 is represented in Fig. 6.

3. Comparison of two algorithms for the ICP

In order to illustrate how the theoretical results of the previous section can help in the design of efficient algorithms for the ICP, we have developed two tabu search algorithms, the second one being based on Theorem 1 for avoiding the visit of equivalent solutions. We will show on a limited set of instances that the second tabu search algorithm possibly finds better solutions than the first one. The computational experiments are not meant to be exhaustive, but rather indicative and should help to orient future research on the development of more elaborate algorithms for the ICP.

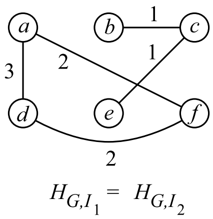

Fig. 6. The weighted interval graph associated with the 7-interval colorings of Fig. 5.

Tabu search
Generate an initial solution $s \in S$, set $T \leftarrow \emptyset$ and $s^* \leftarrow s$;
while no stopping criterion is met **do**
  Determine a solution $s' \in N(s)$ with minimum value $f(s')$ such that the move from s to s' does not belong to T or $f(s') < f(s^*)$;
  if $f(s') < f(s^*)$ **then**
    set $s^* \leftarrow s'$;
  Set $s \leftarrow s'$ and update T.

Fig. 7. General scheme of a tabu search algorithm.

3.1. Two tabu search algorithms for the ICP

Let S be the set of solutions to a combinatorial optimization problem, and f a function to be minimized over S. For a solution $s \in S$, let $N(s)$ denote the *neighborhood* of s which is defined as the set of solutions in S obtained from s by performing a local change, called *move*. A local search is an algorithm that generates a sequence $s_0, s_1, \ldots, s_r$ of solutions in S, where s_0 is an initial solution and each s_i $(i \geq 1)$ belongs to $N(s_{i-1})$. Tabu search is one of the most famous local search algorithms. In order to avoid cycling, tabu search uses a *tabu list* T that contains forbidden moves. More precisely, a move m from s_{i-1} to s_i is forbidden, and s_i is called a *tabu solution*, if m belongs to the tabu list T and $f(s_i) \geq f(s^*)$, where s^* is the best solution encountered so far. At each iteration, the algorithm moves from the current solution s_{i-1} to the best non tabu neighbor $s_i \in N(s_{i-1})$, even if $f(s_i) > f(s_{i-1})$. The general scheme of a tabu search algorithm is given in Fig. 7. For more details on tabu search, the reader may refer to [11].

In order to illustrate how the theoretical results of the previous section can help in the design of efficient algorithms for the ICP, we have developed two tabu search algorithms, the second one being based on Theorem 1 for avoiding the visit of equivalent solutions. Both tabu search algorithms are heuristic methods for the k-ICP. They are used to solve the ICP with the following scheme.

1. Determine an upper bound k on $\chi_{int}(G)$.
2. Apply tabu search for the $(k-1)$-ICP; if the output is a legal $(k-1)$-interval coloring then set $k \leftarrow k-1$ and repeat step 2, else return k.

Tabucol [14] is a tabu search algorithm for the classical k-coloring problem (i.e., for the k-ICP with $\omega_i = 1$ for all vertices $i \in V$). The search space S is the set of (not necessary legal) k-colorings of G. A solution $c \in S$ is therefore a partition of the vertex set into k subsets $V_1, \ldots, V_k$. The evaluation function f measures the number of conflicting edges. Hence, for a solution $c = (V_1, \ldots, V_k)$ in S, $f(c)$ is equal to $\sum_{i=1}^{k} |E_i|$, where E_i denotes the set of edges with both endpoints in V_i. The goal of *Tabucol* is to determine a k-coloring c such that $f(c) = 0$. Given a k-coloring c, a neighbor k-coloring $c' \in N(c)$ is obtained by choosing an endpoint i of a conflicting edge and assigning a new color $c'(i) \neq c(i)$ to i. When modifying the color $c(i)$ of a vertex i, the tabu list stores the ordered triple $(i, c(i), f(c))$, which means that for some number of iterations, all moves to a solution c' with $f(c') \geq f(c)$ and $c'(i) = c(i)$ have a tabu status.

The first proposed tabu search algorithm, called TABU_1, is a simple adaptation of *Tabucol* to the k-ICP. The search space S is the set of (not necessary legal) k-interval colorings of G. The evaluation function f measures the total overlap of intervals on adjacent vertices. More precisely, given a k-interval coloring I of G, we define

$$f(I) = \sum_{[i,j] \in E} |I(i) \cap I(j)|.$$

Hence, a k-interval coloring is legal if and only if $f(I) = 0$. Given a k-interval coloring I, the neighborhood of I is defined as the set of solutions I' which can be obtained from I by choosing an endpoint i of a conflicting edge and assigning a new interval $I'(i) \neq I(i)$ to i. We denote $N_1(I)$ the set of such neighbors of I. Let $\min_I(i)$ denote the smallest integer in the interval $I(i)$. When modifying the interval $I(i)$ of a vertex i, the tabu list TL_1 stores the ordered triple $(i, \min_I(i), f(I))$, which means that for some number of iterations, a move to a solution I' with $f(I') \geq f(I)$ and $\min_{I'}(i) = \min_I(i)$ (i.e., with $I(i) = I'(i)$) has a tabu status.

Consider again the *Tabucol* algorithm for the classical k-coloring problem. Let $c' \in N(c)$ be a neighbor solution of a k-coloring c obtained by modifying the color of vertex i. Then c and c' are equivalent if and only if $c(j) \notin \{c(i), c'(i)\}$ for all vertices $j \neq i$. Indeed, if there is a vertex $j \neq i$ with $c(i) = c(j)$ then i and j belong to the same color class in c but not in c'. Similarly, if $c'(i) = c(j)$ with $j \neq i$ then i and j belong to the same color class in c' but not in c. In summary, c and c' are equivalent if and only if i is the unique vertex with color $c(i)$ and the new color $c'(i)$ assigned to i is not used by any vertex $j \neq i$, which is most unlikely. The situation is totally different for the k-ICP. For example, consider the 8-interval coloring I at the top of Fig. 8. It contains three conflicting edges, namely $[c, f]$, $[d, e]$ and $[e, f]$. Hence, a neighbor of I is obtained by changing the interval associated with vertex c, d, e or f, which gives a total of 21 neighbors in $N_1(I)$. Four of these neighbors are equivalent to I and represented at the bottom of Fig. 8.

The second tabu search algorithm, called TABU_2, has only two differences with TABU_1. The first difference is on the definition of the neighborhood of a k-interval coloring. In order to avoid visiting equivalent solutions, the neighborhood $N_2(I)$ of a k-interval colorings I is defined as the subset of solutions $I' \in N_1(I)$ such that there exists at least one conflicting edge $[i, j]$ with $|I(i) \cap I(j)| \neq |I'(i) \cap I'(j)|$. Hence, given a k-interval coloring I, a neighbor solution $I' \in N_2(I)$ is obtained by choosing a vertex $i \in V$ that is the endpoint of a conflicting edge and assigning a new interval $I'(i) \neq I(i)$ to i such that there is at least one vertex j adjacent to i with $|I(i) \cap I(j)| > 0$ and $|I(i) \cap I(j)| \neq |I'(i) \cap I(j)|$. According to Theorem 1 this is sufficient to ensure that no solution in $N_2(I)$ is equivalent to I. For the 8-interval coloring I at the top of Fig. 8, $N_1(I)$ contains 21 neighbors while $N_2(I)$ contains only 13 neighbors obtained by setting $\min_{I'}(c) = 1$ or 2, $\min_{I'}(d) = 1, 2, 3, 4$ or 6, $\min_{I'}(e) = 1, 2, 3, 5$ or 7, or $\min_{I'}(f) = 1$. The four neighbors $I' \in N_1(I)$ represented at the bottom of Fig. 8 do not belong to $N_2(I)$ since they are equivalent to I. In addition, $N_2(I)$ does not contain the neighbors I' with $\min_{I'}(c) = 5, 6$ or 7 or $\min_{I'}(f) = 2$ since these values do not change the size of the overlap of the intervals on the endpoint of a conflicting edge.

The second difference between TABU_1 and TABU_2 is on the definition of a tabu move. We consider a second tabu list TL_2, and a move is declared tabu if both tabu lists assign a tabu status to the move. The second tabu list is defined as follows. When modifying the interval associated with a vertex i for moving from a solution I to a neighbor solution $I' \in N_2(I)$, we consider all vertices j such that $|I(i) \cap I(j)| > 0$ and $|I(i) \cap I(j)| \neq |I'(i) \cap I(j)|$, and for each such vertex we insert the ordered quadruple $(i, j, |I(i) \cap I(j)|, f(I))$ in a second tabu list TL_2. The move from a solution I to a solution I' is considered as tabu according to TL_2 if there exists $(i, j, q, r) \in TL_2$ such that $|I(i) \cap I(j)| \neq |I'(i) \cap I'(j)| = q$ and $f(I') \geq r$. Notice that an ordered quadruple (i, j, q, r) is introduced in TL_2 only if $q > 0$, which means that we never impose that two intervals should not overlap since this would forbid the visit of too many solutions. For illustration, if we move from the 8-interval coloring I at the top of Fig. 8 to a neighbor solution I' by setting $I'(f) = \{1, 2, 3, 4, 5, 6\}$, then $(a, f, 6, 4)$ and $(e, f, 2, 4)$ are introduced in the tabu list. While $|I(b) \cap I(f)| \neq |I'(b) \cap I'(f)|$, the ordered quadruple $(b, f, 0, 4)$ does not enter the tabu list since $|I(b) \cap I(f)| = 0$.

3.2. Computational experiments

We first report computational experiments on 52 DIMACS benchmark graphs having up to 191 vertices. These instances have also been considered in [1], and therefore constitute a test set on which comparisons can be made with other algorithms. For a detailed description of these instances, the reader can refer to [20].

For measuring the performance of the proposed algorithms, we also report known lower and upper bounds on the optimal solution. More precisely, Čangalović and Schreuder in [3] have described an exact algorithm for finding the interval chromatic number. It is based on the Branch-and-Bound principle. An initial lower bound $LB(G)$ on $\chi_{int}(G)$ is obtained by determining a clique of maximum total weight, using a variation of the algorithm proposed in [19]. Also, an initial upper bound $UB(G)$ on $\chi_{int}(G)$ is obtained by using the heuristic algorithm proposed in [4]. Moreover, two truncated Branch-and-Bound algorithms for the *ICP* are proposed in [1]. Both algorithms are run with a time limit of one hour. Their output is either the optimal value (i.e., the interval chromatic number), or an upper bound on $\chi_{int}(G)$.

When using TABU_1 or TABU_2, we start the search with $k = UB(G)$. Then, as explained at the beginning of Section 3, we decrease k by one unit if a legal $(k-1)$-interval coloring is found, and this process is repeated until a time limit of 20 min is reached. There are actually 73 instances having weights associated to nodes in the DIMACS benchmarks [20]. For 21 of these instances we got $LB(G) = UB(G)$ which means that $\chi_{int}(G) = LB(G)$ and there is nothing to optimize for these graphs. We therefore only report results on the 52 other instances for which $LB(G) < UB(G)$. All tests were performed on an Intel(R) Core(TM)2 cpu 6400/2.13 GHz. According to preliminary experiments, the duration of a tabu status for the first tabu list is randomly chosen at each iteration in the interval $[\sqrt{k|V|}, 3\sqrt{k|V|}\ ]$ while the interval $[\frac{1}{2}|V|\sqrt{\max_{i\in V}\omega_i}\ ,\ \frac{3}{2}|V|\sqrt{\max_{i\in V}\omega_i}\ ]$ is used for the second tabu list. A better tuning of these parameters is certainly possible, but the chosen values turned to be the best for our limited test set.

The three first columns of Table 1 contain the name of the instances, their number n of vertices, and their number m of edges. The next column indicates the largest vertex weight (column labeled "max ω") which also corresponds to the number of different vertex weights. We then report the value of the lower and upper bounds $LB(G)$ and $UB(G)$ (columns labeled "LB" and "UB") mentioned above. Column labeled "Trunc BB" contains the best upper bound obtained in [1] with one of the two truncated Branch-and-Bound algorithms. When a proof of optimality was obtained, we use bold numbers. The next three columns contain the results obtained using TABU_1. We ran TABU_1 five times on each graph, and columns "Best", "Worse" and "Average" contain the best, the worse and the average solution values we have reached. Again, we use bold numbers when a solution produced with TABU_1 is known to be optimal (because it reaches the lower bound $LB(G)$ or is equal to an

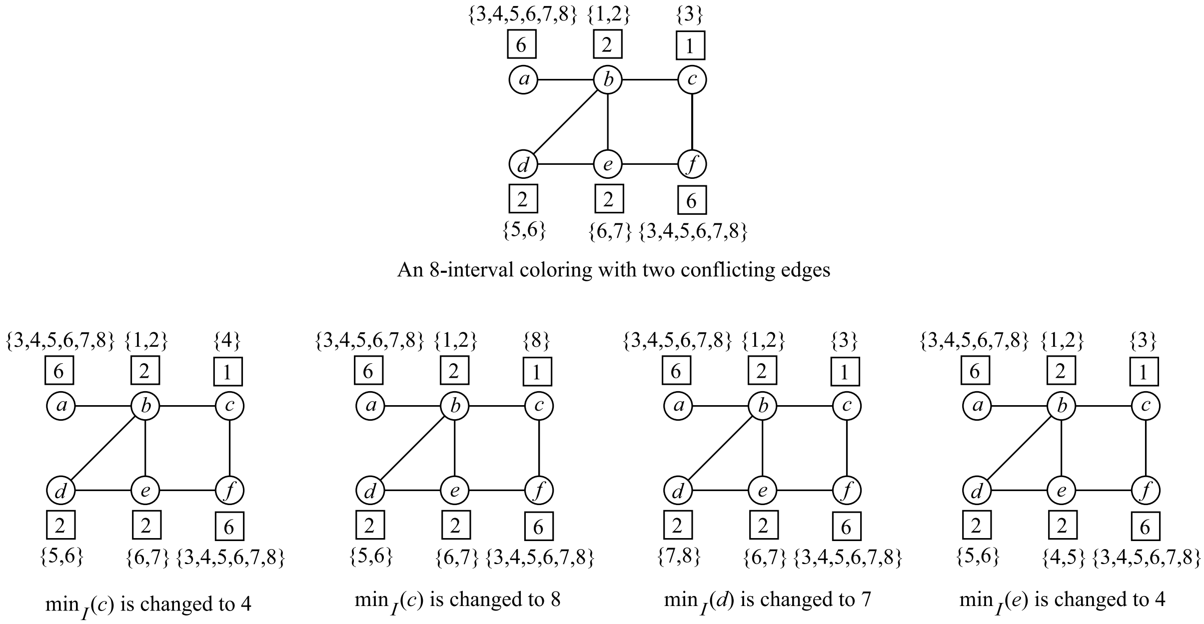

Fig. 8. An 8-interval coloring with 4 equivalent neighbors.

optimal value reported in column "Trunc BB"). The last three columns contain the same information for $TABU_2$. The last line contains average numbers for each column.

We observe that both algorithms produce better results than those reported in [1] (column "Trunc BB"). We cannot conclude from these 52 benchmark problems which among $TABU_1$ and $TABU_2$ is the best algorithm since they give in average the same best results. We observe however a larger variance for $TABU_2$: the average difference between the worse and the best solution produced by $TABU_2$ is $95.54 - 94.46 = 1.08$, while it is equal to 0.85 for $TABU_1$. Note that the use of Theorem 1 for avoiding the visit of equivalent solutions has helped to determine an optimal solution for instances queen10_10gb and queen11_11g. $TABU_2$ could however never reach a 135-interval coloring for queen9_9gb, while such an optimal solution was obtained with $TABU_1$.

While experimenting $TABU_1$ and $TABU_2$ on other instances, we have noticed that $TABU_2$ tends to produce better results than $TABU_1$ only on instances in which the vertices have almost all the same weights. To illustrate this fact, we now report results obtained on a second set of instances and then give a possible explanation of this observation.

Given a positive integer n, a real number $p \in [0, 1]$ and a set Ω of positive integers, a random graph $G_{n,p,\Omega}$ contains n vertices, all $n(n-1)/2$ ordered pairs of vertices have a probability p of being linked by an edge, and the weight ω_i of a vertex i is chosen randomly according to a uniform distribution in Ω. We have generated such random graphs with $n = 100$, 125 and 250, $p = 0.5$, and with three different weight sets $\Omega_1 = \{7, 9\}$, $\Omega_2 = \{5, 7, 9, 11\}$ and $\Omega_3 = \{1, 3, 5, 7, 9, 11, 13, 15\}$. For each triplet (n, p, Ω_i), four graphs were created and we have then run $TABU_1$ and $TABU_2$ five times on each graph. Average results are reported in Fig. 9. More precisely, for each set Ω_i we give the average number of colors used by $TABU_1$ and $TABU_2$ on the sixty runs (five runs on each of the twelve graphs), the plain line being for $TABU_1$ and the dotted one for $TABU_2$. We observe that $TABU_2$ is better than $TABU_1$ with the weight sets Ω_1 and Ω_2, and worse with Ω_3 which contains many different weights.

Given a k-interval coloring I, the solutions I' in $N_2(I)$ are obtained from I by assigning a new interval $I'(i)$ to an endpoint i of a conflicting edge so that $|I(i) \cap I(j)| \neq |I'(i) \cap I'(j)|$ for at least one conflicting edge $[i, j]$. As observed in Section 3.1, this is a sufficient condition to ensure that there is no solution equivalent to I in $N_2(I)$. The condition is however not necessary since it may happen that $|I(i) \cap I(j)| = |I'(i) \cap I'(j)|$ for all conflicting edges $[i, j]$ while the existence of a vertex j' with $|I(i) \cap I(j')| \neq |I'(i) \cap I'(j')|$ makes I' not equivalent to I. In summary, it may happen that many solutions in $N_1(I)$ but not in $N_2(I)$ are not equivalent to I and we now show that this turns to be particularly true when the weights on the vertices have many different values.

Consider any conflicting edge $[i, j]$ and assume first that $I(i) \not\subset I(j)$ and $I(j) \not\subset I(i)$. Then there is only one change of $I(i)$ to $I'(i)$ which gives $|I(i) \cap I(j)| = |I'(i) \cap I'(j)|$ (if $\min_I(i) \neq \min_I(j)$). For example, if $I(i) = \{1, 2, 3\}$ and $I(j) = \{3, 4, 5, 6, 7, 8\}$, then $|I(i) \cap I(j)| = |I'(i) \cap I'(j)|$ only if $I'(i) = \{8, 9, 10\}$. The situation is different when $I(i) \subset I(j)$ or $I(j) \subset I(i)$. Indeed, there are then $|\omega_i - \omega_j|$ possible ways of changing $I(i)$ to $I'(i)$ so that $|I(i) \cap I(j)| = |I'(i) \cap I'(j)|$. If $[i, j]$ is the unique conflicting edge, then none of these solutions belong to $N_2(I)$ while there all belong to $N_1(I)$ and are possibly all non equivalent to I. For example, if $I(i) = \{4, 5, 6\}$ and $I(j) = \{3, 4, 5, 6, 7, 8\}$, then the solutions obtained by setting $\min_I(i) = 3$, 5 or 6 do not belong to $N_2(I)$ while the existence of a vertex j' not adjacent to i with $I(j') = \{6, 7, 8\}$ is sufficient to prove that I' is not equivalent to I if $\min_I(i)$ is set equal to 3, 5 or 6.

Table 1
Results for DIMACS benchmark graphs.

Instance			max ω	LB	UB	Trunc BB	TABU$_1$			TABU$_2$		
Name	n	m					Average	Best	Worse	Average	Best	Worse
DSJC125.1gb	125	736	20	67	86	73	71.6	71	72	73	72	74
DSJC125.1g	125	736	5	19	23	**19**	**19**	**19**	**19**	**19**	**19**	**19**
DSJC125.5gb	125	3891	20	125	246	237	202.8	200	205	204	202	206
DSJC125.5g	125	3891	5	40	72	68	63.2	63	64	62.8	62	63
DSJC125.9gb	125	6961	20	425	608	608	545.2	543	548	548.4	543	552
DSJC125.9g	125	6961	5	122	166	163	154	153	155	152.8	151	154
GEOM100b	100	1050	3	30	31	**30**	**30**	**30**	**30**	**30**	**30**	**30**
GEOM100	100	547	10	60	61	**60**	**60**	**60**	**60**	**60**	**60**	**60**
GEOM110	110	638	10	62	67	**62**	**62**	**62**	**62**	**62**	**62**	**62**
GEOM120a	120	1434	10	93	98	98	**93**	**93**	**93**	**93**	**93**	**93**
GEOM120b	120	1491	3	34	35	**34**	**34**	**34**	**34**	**34**	**34**	**34**
GEOM120	120	773	10	63	68	65	64	64	64	64	64	64
GEOM50b	50	249	3	17	18	**17**	**17**	**17**	**17**	**17**	**17**	**17**
GEOM60a	60	339	10	65	66	**65**	**65**	**65**	**65**	**65**	**65**	**65**
GEOM60b	60	366	3	22	23	**22**	**22**	**22**	**22**	**22**	**22**	**22**
GEOM70b	70	488	3	22	23	**22**	**22**	**22**	**22**	**22**	**22**	**22**
GEOM80a	80	612	10	68	72	69	**68**	**68**	**68**	**68**	**68**	**68**
GEOM80b	80	663	3	25	26	**25**	**25**	**25**	**25**	**25**	**25**	**25**
GEOM90a	90	789	10	65	70	**65**	**65**	**65**	**65**	**65**	**65**	**65**
GEOM90	90	441	10	51	52	**51**	**51**	**51**	**51**	**51**	**51**	**51**
myciel5gb	47	236	20	37	65	**53**	**53**	**53**	**53**	**53**	**53**	**53**
myciel5g	47	236	5	10	19	**17**	**17**	**17**	**17**	**17**	**17**	**17**
myciel6gb	95	755	20	39	92	79	78	78	78	78.8	78	79
myciel6g	95	755	5	10	25	**20**	**20**	**20**	**20**	**20**	**20**	**20**
myciel7gb	191	2360	20	40	98	90	87.8	87	88	89.2	88	91
myciel7g	191	2360	5	10	28	24	24	24	24	24	24	24
queen10_10gb	100	1470	20	136	159	146	138.4	137	139	137.4	**136**	138
queen10_10g	100	1470	5	38	43	39	**38**	**38**	**38**	**38**	**38**	**38**
queen11_11gb	121	1980	20	140	170	165	149.2	148	151	150.2	150	151
queen11_11g	121	1980	5	41	48	44	42	42	42	41.2	**41**	42
queen12_12gb	144	2596	20	163	192	179	165.8	165	167	164.4	164	165
queen12_12g	144	2596	5	42	52	49	46.6	46	47	46	46	46
queen8_8gb	64	728	20	113	120	113	113	113	113	113	113	113
queen8_8g	64	728	5	28	34	29	29	29	29	29	29	29
queen9_9gb	81	1056	20	135	157	145	136	**135**	137	137.4	136	139
queen9_9g	81	1056	5	35	39	**35**	**35**	**35**	**35**	**35**	**35**	**35**
R100_1gb	100	509	20	56	78	65	64	64	64	64.4	64	65
R100_1g	100	509	5	15	19	**17**	**17**	**17**	**17**	**17**	**17**	**17**
R100_5gb	100	2456	20	132	222	208	184.4	183	186	186	185	187
R100_5g	100	2456	5	35	58	54	50	50	50	50	50	50
R100_9gb	100	4438	20	390	512	512	459.8	458	462	459.4	457	462
R100_9g	100	4438	5	108	140	134	126.8	126	127	126.4	126	127
R50_5gb	50	612	20	98	131	118	112.8	112	113	113.6	113	115
R50_5g	50	612	5	27	34	32	31.6	31	32	31.6	31	32
R50_9gb	50	1092	19	228	264	264	235.6	235	237	234.6	234	235
R50_9g	50	1092	5	64	73	68	66	66	66	66	66	66
R75_1gb	70	251	20	53	69	57	57	57	57	57	57	57
R75_1g	70	251	5	14	19	**16**	**16**	**16**	**16**	**16**	**16**	**16**
R75_5gb	75	1407	20	114	184	171	154	153	155	155.8	154	157
R75_5g	75	1407	5	31	50	46	43.6	43	44	43	43	43
R75_9gb	75	2513	20	298	393	393	341.8	340	344	342.4	338	346
R75_9g	75	2513	5	85	104	101	96	96	96	95.4	95	96
Average				81.54	107.73	102.62	94.88	94.46	95.31	95.02	94.46	95.54

In summary, the number of solutions I' in $N_1(I)$ not equivalent to I and not in $N_2(I)$ is proportional to the values $|\omega_i - \omega_j|$ of the conflicting edges $[i, j]$. Such values are small only when the weights on the vertices are almost all the same, which gives a possible explanation of the curves in Fig. 9.

A better neighborhood for tabu search would contain all solutions in $N_1(I)$ which are not equivalent to I, but this would increase the computational complexity. Indeed, for the endpoint i of a conflicting edge, one can test in $O(|V|)$ whether a change of $I(i)$ to a new interval $I'(i)$ gives a solution I' equivalent to I. Hence, such a neighborhood could be generated with a time complexity in $O(F(I)|V|k)$, where $F(I)$ denotes the number of endpoints of conflicting edges in a k-interval coloring I. While $F(I) \in O(|V|)$ (which gives a theoretical time complexity in $O(|V|^2k)$, $F(I)$ typically contains only few vertices which gives a time complexity in $O(|V|k)$ in practice. For comparison, the generation of $N_2(I)$ can be performed in time complexity in $O(F'(I)k)$, where $F'(I)$ denotes the number of conflicting edges in I. This is theoretically in $O(|E|k)$ but typically in $O(k)$ since the number of conflicting edges in $TABU_2$ reduces very quickly to a few units.

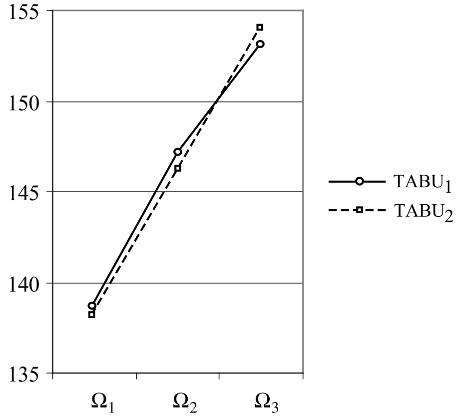

Fig. 9. Experiments on random graphs with various weight sets.

4. Conclusion

Two k-interval colorings I_1 and I_2 are said *equivalent* if there is a permutation π of the integers $1, \ldots, k$ such that $\ell \in I_1(i)$ if and only if $\pi(\ell) \in I_2(i)$ for all vertices $i \in V$. We have shown that a necessary and sufficient condition for such an equivalence is to have $|I_1(i) \cap I_1(j)| = |I_2(i) \cap I_2(j)|$ for all vertices i and j. Hence, equivalent solutions to the k-ICP are easy to recognize and we have shown that a tabu search algorithm for the k-ICP can possibly be improved by forbidding the visit of equivalent solutions.

While the two proposed tabu search algorithms produce reasonably good results in comparisons with those published in [1], we do not argue that they constitute the best possible algorithms for the ICP. The experiments reported in Section 3.2 should help to orient future research on the development of more elaborate algorithms for the ICP.

References

[1] M. Bouchard, M. Čangalović, A. Hertz, On a reduction of the interval coloring problem to a series of bandwidth coloring problems, Technical report G-2007-69, Les Cahiers du GERAD, Montréal, Canada, 2007.
[2] A.L. Buchsbaum, H. Karloff, C. Kenyon, N. Reingold, M. Thorup, OPT versus LOAD in dynamic storage allocation, in: Proceedings of the 35th Annual ACM Symposium on Theory of Computing, 2003, pp. 632–646.
[3] M. Čangalović, J.A.M. Schreuder, Exact colouring algorithm for weighted graphs applied to timetabling problems with lectures of different lengths, European Journal of Operational Research 51 (1991) 248–258.
[4] A.T. Clementson, C.H. Elphick, Approximate coloring algorithms for composite graphs, Journal of Operational Research Society 34 (6) (1983) 503–509.
[5] D. de Werra, A. Hertz, Consecutive colorings of graphs, Zeitschrift für Operations Research 32 (1) (1988) 1–8.
[6] J. Fabri, Automatic storage optimization, in: ACM SIGPLAN Notices: Proceedings of the 1979 SIGPLAN symposium on Compiler construction, vol. 148, 1979, pp. 83–91.
[7] P. Galinier, J.K. Hao, Hybrid evolutionary algorithms for graph colorings, Journal of Combinatorial Optimization 3 (1999) 379–397.
[8] P. Galinier, A. Hertz, A survey of local search methods for graph coloring, Computers & Operations Research 33 (2006) 2547–2562.
[9] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freman and Company, NY, 1979.
[10] J. Gergov, Algorithms for compile-time memory optimization, in: Proceedings of the 10th ACM-SIAM Symposium on Discrete Algorithms, 1999. pp. 907–908.
[11] F. Glover, M. Laguna, Tabu Search, Kluwer Academic Publishers, Boston, 1997.
[12] M. Golumbic, Algorithmic Graph Theory and Perfect Graphs, Academic Press, NY, 1980.
[13] G. Hajós, Über eine Art von Graphen, Internationale Mathematische Nachrichten 11 (1957).
[14] A. Hertz, D. de Werra, Using tabu search techniques for graph coloring, Computing 39 (1987) 345–351.
[15] M. Kubale, Interval vertex-coloring of a graph with forbidden colors, Discrete Mathematics 74 (1989) 125–136.
[16] P.M. Pardalos, T. Mavridou, J. Xue, The graph coloring problem: A bibliographic survey, in: Handbook of Combinatorial Optimization, vol. 2, Kluwer Academic Publishers, 1998, pp. 331–395.
[17] S.V. Pemmaraju, S. Penumatcha, R. Raman, Approximating interval coloring and max-coloring in chordal graphs, ACM Journal of Experimental Algorithmics 10 (2005) 1–19.
[18] A. Punter, Systems for timetabling by computer based on graph coloring, Ph.D. Thesis, C.N.A.A., Hatfield Polytechnic, 1976.
[19] T. Sakaki, K. Nakashima, Y. Hattori, Algorithms for finding in the lump both bounds of the chromatic number of a graph, The Computer Journal 19 (1976) 329–332.
[20] M.A. Trick, Computational symposium: Graph coloring and its generalizations. Cornell University, Ithaca, NY, 2002 http://mat.gsia.cmu.edu/COLOR04.