Minimum sum set coloring of trees and line graphs of trees[☆]

Flavia Bonomo[a,b,1], Guillermo Durán[a,c,d,2], Javier Marenco[e,b,3], Mario Valencia-Pabon[f]

[a] *CONICET, Argentina*
[b] *Departamento de Computación, FCEN, Universidad de Buenos Aires, Argentina*
[c] *Departamento de Matemática, FCEN, Universidad de Buenos Aires, Argentina*
[d] *Departamento de Ingeniería Industrial, FCFM, Universidad de Chile, Chile*
[e] *Instituto de Ciencias, Universidad Nacional de General Sarmiento, Argentina*
[f] *LIPN, Université Paris-Nord, France.*

Abstract

In this paper, we study the Minimum Sum Set Coloring (MSSC) problem which consists in assigning a set of $x(v)$ positive integers to each vertex v of a graph so that the intersection of sets assigned to adjacent vertices is empty and the sum of the assigned set of numbers to each vertex of the graph is minimum. The MSSC problem occurs in two versions: *non-preemptive* and *preemptive*. We show that the MSSC problem is strongly NP-hard both in the preemptive case on trees and in the non-preemptive case in line graphs of trees. Finally, we give exact parameterized algorithms for these two versions on trees and line graphs of trees.

Key words: graph coloring, minimum sum coloring, set-coloring, trees, line graphs of trees.

1. Introduction

A *vertex coloring* of a graph $G = (V, E)$ is an assignment of colors to the vertices in V such that adjacent vertices receive different colors. We assume that the colors are positive integers. A vertex k-coloring of a graph G is a coloring where the color of each vertex in V is taken from the set $\{1, 2, \ldots, k\}$. Given a vertex coloring of a graph G, the *sum* of the coloring is the sum of the colors assigned to the vertices. The *chromatic sum* $\Sigma(G)$ of G is the smallest sum that can be achieved by any proper coloring of G. In the *Minimum Sum Coloring* (MSC) problem we have to find a coloring of G with sum $\Sigma(G)$.

[☆]Partially supported by ANPCyT PICT-2007-00518 (Argentina), BQR-2008 Université Paris-Nord Grant (France), and Math-AmSud Project 10MATH-04 (France-Argentina-Brazil).

Email addresses: fbonomo@dc.uba.ar (Flavia Bonomo), fbonomo@dc.uba.ar (Guillermo Durán), fbonomo@dc.uba.ar (Javier Marenco), valencia@lipn.univ-paris13.fr (Mario Valencia-Pabon)

[1]Partially supported by ANPCyT PICT-2007-00533, and UBACyT Grants X069, X606 and 20020090300094 (Argentina).

[2]Partially supported by FONDECyT Grant 1080286 and Millennium Science Institute "Complex Engineering Systems" (Chile), and UBACyT Grant X069 (Argentina).

[3]Partially supported by UBACyT Grant X069 (Argentina).

The MSC problem was introduced by Kubicka [13]. The problem is motivated by applications in scheduling [1, 2, 10, 11] and VLSI design [18, 21]. The computational complexity of determining the vertex chromatic sum of a simple graph has been studied extensively since then. In [14] it is shown that the problem is NP-hard in general, but polynomial time solvable for trees. The dynamic programming algorithm for trees can be extended to partial k-trees and block graphs [12]. Furthermore, the MSC problem is NP-hard even when restricted to some classes of graphs for which finding the chromatic number is easy, such as bipartite or interval graphs [2, 21]. A number of approximability results for various classes of graphs were obtained in the last ten years [1, 7, 10, 11, 5].

The edge coloring version of the MSC problem, the *Minimum Sum Edge Coloring* (MSEC) problem, has been defined in an analogous way. The MSEC problem is NP-hard for bipartite graphs [8], even if the graph is also planar and has maximum degree 3 [15]. Furthermore, in [15] it is also shown that the MSEC is NP-hard for 3-regular planar graphs and for partial 2-trees. A problem generalizing MSC is the *Optimum Cost Chromatic Partition* (OCCP) problem, where there is a finite set of colors, each color has a cost and the aim is to minimize the total sum of the costs. Independently of the result given by Jansen in [12] concerning the polynomial time complexity of an even more general optimization problem (i.e. the Generalized Optimum Cost Chromatic Partition Problem) on block graphs, it has been shown in [8, 20, 22] that the MSEC problem can be solved in polynomial time on trees by a dynamic programming algorithm that uses weighted bipartite matching as a subroutine (in fact, notice that the class of block graphs includes trees and line graphs of trees). In [4], it has been shown that this problem is also polynomial time solvable for multicycles (i.e. cycles with parallel edges). For general multigraphs, a 1.829-approximation algorithm for the MSEC problem is presented in [11]. For bipartite graphs there exist better approximation ratios: a 1.796-approximation algorithm is given in [10], and a 1.414-approximation algorithm is proposed recently in [6].

An interesting application of the MSEC problem is to model dedicated scheduling of biprocessor jobs. The vertices correspond to the processors and each edge $e = uv$ corresponds to a job that requires a time unit of simultaneous work on the two preassigned processors u and v. The colors correspond to the available time slots. A processor cannot work on two jobs at the same time, this corresponds to the requirement that a color can appear at most once on the edges incident to a vertex. The objective is to minimize the average time before a job is completed. When there can be $x(e)$ instances of the same job, it arises the notion of *set-coloring* of the corresponding conflict graph. In general, the main application of the MSC and MSSC problems is the problem of resource allocation with constraints imposed by *conflicting resource requirements*. In such a problem, the constraints are given by a conflict graph G, in which the nodes represent processors, the edges indicate competition on resources (i.e., two nodes are adjacent if the corresponding processors cannot run their jobs simultaneously), and the objective is to minimize the average response time of the system (see [1] for more details).

Formally, given a simple graph $G = (V, E)$ and a demand function $x : V \to \mathbb{Z}^+$, a *vertex set-coloring* of (G, x) consists in assigning to each vertex $v \in V$ a set of $x(v)$ colors in such a way that adjacent vertices will be assigned disjoint sets of colors. Given a vertex

set-coloring of a graph G with demand function x, the *sum* of the set-coloring is the sum of the colors in the set assigned to each one of the vertices. The *chromatic set-sum* $\Sigma(G, x)$ of (G, x) is the smallest sum that can be achieved by any proper set-coloring of (G, x). In the *Minimum Sum Set Coloring* (MSSC) problem we have to find a set-coloring of (G, x) with sum $\Sigma(G, x)$. Clearly, when $x(v) = 1$ for each vertex v of the graph, the MSSC problem becomes the MSC problem. The dedicated scheduling of biprocessor jobs with multiple instances can be modeled as a MSSC problem on the line graph of the conflict graph. We consider two variants of the MSSC problem. In the *preemptive* (PMSSC) problem, each vertex may get any set of colors, while in the *non-preemptive* (NPMSSC) problem, the colors assigned to each vertex have to be consecutive. The non-preemptive case arises when each job requires a high cost setup on the processors, and thus the objective is to minimize the average time before a job is completed, within the solutions minimizing the setup costs. The preemptive version corresponds to the scheduling approach commonly used in modern operating systems, where jobs may be interrupted during their execution and resumed at a later time.

A similar set-coloring problem with a different objective function is the *Sum Multicoloring* (SMC) problem. The SMC problem has been introduced in [3] by Bar-Noy et al. Formally, given a graph G and a demand function x on the vertices of G, the SMC problem looks for a set-coloring of (G, x) which minimizes the sum of the largest color assigned to each vertex of G. The SMC problem models scheduling problems where each job v requires $x(v)$ unit execution times to be completed and the goal is to minimize the average completion time of the jobs (or equivalently, the sum of the completion times). In [9], it has been shown that the non-preemptive case of the SMC problem (NPSMC) can be solved in polynomial time for trees. However, in [17] Marx shows that the preemptive case of the SMC problem (PSMC) is strongly NP-hard even for binary trees. More recently, Marx has shown in [16] that both the PSMC and the NPSMC problems are strongly NP-hard problems on line graphs of trees.

It is not difficult to observe that an optimal solution for the MSSC problem is not always an optimal solution for the SMC and vice-versa. In fact, consider a path on five vertices $v_1, v_2, \ldots, v_5$ and a demand function $x(v_i) = 2$ for $i = 1, 3, 5$, and $x(v_i) = 1$, for $i = 2, 4$. The only optimal solution for the MSSC problem (preemptive and non-preemptive) on this instance is $\{1, 2\}, \{3\}, \{1, 2\}, \{3\}, \{1, 2\}$, and the only optimal solution for the SMC problem (preemptive and non-preemptive) on this instance is $\{2, 3\}, \{1\}, \{2, 3\}, \{1\}, \{2, 3\}$. Moreover, in [16] Marx shows that the set-coloring version of the OCCP problem (that generalizes the PMSSC problem), can be solved in polynomial time on line graphs of trees, but the PSMC problem is strongly NP-hard on the same class of graphs.

Nevertheless, the polynomial time dynamic programming algorithm given by Halldórsson et al. in [9] for the NPSMC problem on trees can be adapted in a straightforward way in order to solve the NPMSSC problem on trees within the same computational complexity. Therefore, the NPMSSC problem on trees can be solved in polynomial time.

In Section 3, we prove that the PMSSC problem is strongly NP-hard on trees. In Section 4, it is shown that the NPMSSC problem is strongly NP-hard on line graphs of trees. These results distinguish the MSC problem and both versions of the MSSC problem

in terms of computational complexity, since the MSC problem is polynomial time solvable on block graphs. Moreover, in this paper we give exact parameterized algorithms for these problems on these classes of graphs.

2. Definitions and notation

Assume that we are given a graph $G = (V, E)$ with a demand function $x : V \to \mathbb{Z}^+$. We will denote $n = |V|$, $d(v)$ the degree of a vertex $v \in V$, $\Delta = \max_{v \in V} d(v)$ and $p = \max_{v \in V} x(v)$. The neighborhood of a vertex v will be denoted by $N_G(v)$.

We can consider that the input of our problem is a graph $G = (V, E)$ with a demand function $x : V \to \mathbb{Z}^+$, so the input size is $|V| + |E| + \sum_{v \in V} \log(x(v))$. In the preemptive case it makes some sense to consider as the size of the problem $|V| + |E| + \sum_{v \in V} x(v)$, since it is the output size. Nevertheless, we will call pseudo-polynomial time algorithms to those that are polynomial on $|V| + |E| + \sum_{v \in V} x(v)$.

It is easy to deduce that the minimum number of colors that can be used in an optimum solution of the MSC problem on a graph is upper bounded by $\Delta + 1$. Indeed, assume that we are given a graph $G = (V, E)$ with a demand function $x : V \to \mathbb{Z}^+$. Denote by $C(G, x)$ (resp. $C'(G, x)$) the minimum number of colors that can be used in an optimum solution of the NPMSSC (resp. PMSSC) problem on G with demand function x. It is easy to generalize the bound above and get $C'(G, x) \leq p(\Delta + 1)$. We now show the following lemma.

Lemma 1. *Let G be a graph and let x be a demand function from the vertices of G to the set of positive integers. Then $C(G, x) \leq 2p\Delta - \Delta + p$.*

Proof. Let v be the vertex using the highest color. Since the goal is to minimize the total sum of colors, it uses at most the interval $[c + 1, c + x(v)]$, where c is the maximum color used by one of its neighbors. In the worst case, all of its neighbors $v_1, \ldots, v_{d(v)}$ use disjoint intervals, the smaller beginning at $x(v)$ and separated by intervals of size $x(v) - 1$ (that is, v_1 uses the interval $[x(v), x(v) + x(v_1) - 1]$, v_2 uses the interval $[2x(v) + x(v_1) - 1, 2x(v) + x(v_1) + x(v_2) - 2]$, and so on). So $c \leq \sum_{w \in N_G(v)} (x(w) + x(v) - 1) \leq (2p - 1)\Delta = 2p\Delta - \Delta$, and thus $c + x(v) \leq 2p\Delta - \Delta + p$. □

Let $P = V_1, \ldots, V_t$ be a partition of the vertices of a graph G with demand x. We will call *P-good* to a coloring of (G, x) where each V_i, $i = 1, \ldots, t$, is colored with sum $\Sigma(G[V_i], x)$. Clearly, a P-good coloring is optimum for the MSSC problem, and if G admits a P-good coloring, then every optimum coloring of G must be P-good.

3. The PMSSC problem on trees

We will show that the PMSSC problem on trees is in general NP-hard, even considering $\sum_{v \in V(T)} x(v)$ as the input size.

Theorem 1. *The PMSSC problem on trees is strongly NP-hard.*

Proof. The reduction is from 3-SAT. First we give some definitions and we introduce some families of special trees that will be used as gadgets in the NP-hardness proof.

For a, b positive integers, $a \leq b$, let $T_{[a,b]}$ be a tree with root r with demand $x(r) = b-a+1$ and, if $a > 1$, two children v_1, v_2 with demand $x(v_1) = x(v_2) = a-1$ (see Figure 1).

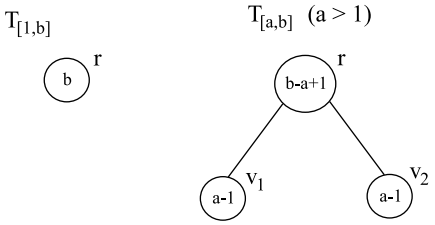

Figure 1: The tree $T_{[a,b]}$ for $1 \leq a \leq b$.

The trees $T_{[a,b]}$ admit P-good colorings for suitable partitions P: if $a = 1$ then the partition P is trivial, if $a > 1$ then the partition $P = \{r, v_1\}, \{v_2\}$ is such that $T_{[a,b]}$ admits a P-good coloring. Moreover, in every P-good coloring of $T_{[a,b]}$, vertex v_2 should receive colors $1, \ldots, a-1$ and therefore vertex r should receive colors $a, \ldots, b$ (and vertex v_1 colors $1, \ldots, a-1$ as well).

Let $\{a_1, \ldots, a_k\}$ be a set of positive integers, $a_1 < \ldots < a_k$, and let $S = \sum_{i=1}^{k} a_i - \binom{k+1}{2} + 1$. We will define the tree $T_{\{a_1,\ldots,a_k\}}$. The root r has demand $x(r) = 1$. The children of r are the following: a child v with demand $x(v) = k-1$; S children each of them being the root of $T_{[1,a_1-1]}$, when $a_1 > 1$; S children each of them being the root of $T_{[a_i+1,a_{i+1}-1]}$, when $a_{i+1} > a_i + 1$ for each $i = 1, \ldots, k-1$. Besides, vertex v has S children each of them being the root of $T_{[1,a_1-1]}$, when $a_1 > 1$; S children each of them being the root of $T_{[a_i+1,a_{i+1}-1]}$, when $a_{i+1} > a_i+1$ for each $i = 1, \ldots, k-1$ (see Figure 2). We will analyze now the possible solutions to the PMSSC problem on $T_{\{a_1,\ldots,a_k\}}$. If all the trees $T_{[a,b]}$ are colored in an optimum way, then r and v should receive colors $\{a_1, \ldots, a_k\}$, and the overall sum is $\sum_{i=1}^{k} a_i + D$ where D is the sum of $\Sigma(T_{[a,b]}, x)$ over all the trees $T_{[a,b]}$ involved in $T_{\{a_1,\ldots,a_k\}}$. Indeed, suppose that r and v receive a set of colors $\{a'_1, \ldots, a'_k\}$ different from $\{a_1, \ldots, a_k\}$ in an optimum coloring of $T_{\{a_1,\ldots,a_k\}}$. Since D is locally optimum, then $\sum_{i=1}^{k} a'_i \leq \sum_{i=1}^{k} a_i$, so at least one of the colors not in $\{a_1, \ldots, a_k\}$ is less or equal than $a_k - 1$, and thus at least S trees $T_{[a,b]}$ are colored in a non-optimum way. Therefore the overall sum would be at least $\sum_{i=1}^{k} a'_i + D + S$ and since $\sum_{i=1}^{k} a'_i \geq \binom{k+1}{2}$, this contradicts the optimality of the coloring. Moreover, it is not difficult to see that for each $i = 1, \ldots, k$, there is an optimum coloring of $T_{\{a_1,\ldots,a_k\}}$ where r receives color a_i.

Now, let $\mathcal{I}$ be an instance of 3-SAT, with n variables and m clauses. We will construct $T_{\mathcal{I}}$ as follows: it has a root r with demand $x(r) = n$; the root has $n + m + 1$ children $v_1, \ldots, v_n$, $w_1, \ldots, w_m$, all of them with demand 1, and z with demand $x(z) = n$; each v_i $(1 \leq i \leq n)$ is the root of a copy of $T_{\{2i-1,2i\}}$; each w_i $(1 \leq i \leq m)$ is the root of a copy of $T_{\{a^i_1,a^i_2,a^i_3\}}$, where a^i_1, a^i_2, a^i_3 are the values corresponding to the three literals of the i-th clause of $\mathcal{I}$ in increasing order, assigning to variable k the value $2k-1$ and to its negation the value $2k$. Let $P = \{\{r, z\}, V(T_{\mathcal{I}}) \setminus \{r, z\}\}$ be a partition of the vertices of $T_{\mathcal{I}}$. We

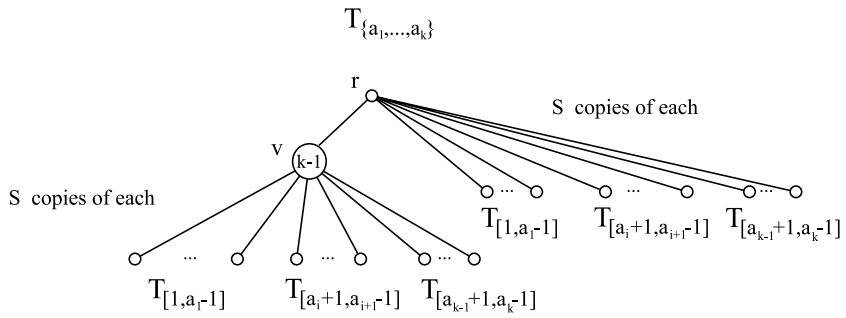

Figure 2: The tree $T_{\{a_1,\ldots,a_k\}}$.

will show that $\mathcal{I}$ is satisfiable if and only if $(T_{\mathcal{I}}, x)$ admits a P-good coloring, that is, a coloring with sum $\binom{2n+1}{2} + \Sigma(T_{\mathcal{I}} \setminus \{r, z\}, x)$ (please note that this value can be computed in polynomial time based on the construction of $T_{\mathcal{I}}$ and the observations above). Suppose first that $\mathcal{I}$ is satisfiable and consider a truth assignment satisfying it. Then assign to z the values corresponding to true literals, to r the values corresponding to false literals, to each w_i the value of a literal satisfying its corresponding clause (and then extend this coloring to an optimum coloring of $T_{\{a_1^i, a_2^i, a_3^i\}}$), and to each v_i the value in $\{2i-1, 2i\}$ not used in r (and then extend this coloring to an optimum coloring of $T_{\{2i-1,2i\}}$). The coloring obtained is P-good. Conversely, suppose that $T_{\mathcal{I}}$ admits a P-good coloring. For $i = 1, \ldots, n$, since $T_{\{2i-1,2i\}}$ is colored in an optimum way, each v_i uses either color $2i-1$ or color $2i$. Moreover, since $\{r, z\}$ use the colors $\{1, \ldots, 2n\}$, r uses exactly one of $\{2i-1, 2i\}$ for each $i = 1, \ldots, n$, and z the other one. Let the variable i be true if $2i-1$ is used in z and false otherwise. For each $i = 1, \ldots, m$, since $T_{\{a_1^i, a_2^i, a_3^i\}}$ is colored in an optimum way, w_i uses one of the colors $\{a_1^i, a_2^i, a_3^i\}$ and then that color should not be used in r, so it should be used in z. If it is an odd color then the corresponding variable is true and appears in the i-th clause, otherwise the corresponding variable is false but its negation appears in the i-th clause. In both cases, the clause is satisfied and so $\mathcal{I}$ is satisfiable. □

However, if the maximum value p of the demand function x is bounded by a constant, then there is a polynomial time algorithm for the PMSSC problem on trees as it is shown in the next theorem.

Theorem 2. *Let $T = (V, E)$ be a tree and let $x : V \to \mathbb{Z}^+$ be a demand function for T. Then the PMSSC problem on T can be solved in $O(n(\Delta p)^{2p})$ time. In particular, if p is bounded by a constant, it can be solved in polynomial time.*

Proof. Let n be the number of vertices in T and let Δ be the maximum degree of the vertices in T. Let m, k be positive integers, and let $[m]^k$ denote the set of all k-subsets of $[m]$, where $[m]$ denotes the set $\{1, \ldots, m\}$. Let $q(A) = \sum_{i \in A} i$, where A is a given arbitrary finite set of positive integers. Let C be an upper bound for $C'(T, x)$, thus $C = O(\Delta p)$. The algorithm is based on the idea of dynamic programming. Let r be an arbitrary vertex in T chosen as the root. For each vertex v of T, we denote by T_v the subtree of T rooted

at vertex v. Now, for each vertex v in T, we construct an array S_v of length $\binom{C}{x(v)}$ such that $S_v[X]$ represents the minimum sum for the subtree T_v when vertex v is assigned the subset $X \in [C]^{x(v)}$. The algorithm computes the values of the arrays S_v in a bottom-up way, from the leaves of the tree up to the root as follows.

If v is a leaf then $S_v[X] = q(X)$, for each subset $X \in [C]^{x(v)}$. Let v be an internal vertex in T, and let $v_1, v_2, \ldots, v_t$ be the children vertices of v. Assume that $S_{v_i}[X]$ is computed, for all $1 \leq i \leq t$ and for all $X \in [C]^{x(v_i)}$, and we want to compute the value of $S_v[Y]$ for some fixed subset $Y \in [C]^{x(v)}$. First, for each children vertex v_i we compute the value $f_v(v_i, Y) = \min_{X \in [C]^{x(v_i)}}\{S_{v_i}[X] : X \cap Y = \emptyset\}$. Once the values $f_v(v_i, Y)$ have been computed for all $1 \leq i \leq t$, we can compute the value of $S_v[Y]$ as follows: $S_v[Y] = q(Y) + \sum_{i=1}^{t} f_v(v_i, Y)$. Clearly, the minimum value of $S_r[Z]$ taken over all subsets $Z \in [C]^{x(r)}$ is the value of an optimal sum for the PMSSC problem for (T, x). The complexity of this algorithm can be easily deduced. □

Notice that this algorithm can be easily adapted to the set-coloring version of the OCCP problem, under the same conditions.

4. The NPMSSC problem on line graphs of trees

The MSSC problem on the line graph $L(G)$ of a graph G and demand function x is equivalent to the Minimum Sum Edge Coloring (MSEC) of a multigraph $\widetilde{G}$, obtained from G by multiplying each edge e by $x(e)$. Therefore, in the sequel, we assume that we have as input to the NPMSEC problem a tree $T = (V, E)$ and a demand function x from the edge-set E to the set of positive integers.

In the following, we will show that the NPMSSC problem on line graphs of trees is in general NP-hard, even considering $\sum_{e \in E(T)} x(e)$ as being part of the input size. The reduction we use is based on the results given by Marx in [16] for the PSMC problem on line graphs of trees.

Theorem 3. *The NPMSSC problem on line graphs of trees is strongly NP-hard.*

Proof. First we we introduce three families of special trees that will be used as gadgets in the NP-hardness proof. For any two positive integers i, j with $i < j$, let $[i, j]$ denote the consecutive interval $\{i, i+1, \ldots, j-1, j\}$ of integers.

Let $G = (A \cup B, E)$ be a bipartite multigraph. Denote by E_v the set of edges incident to vertex v. We will consider the partition of the edges of G defined as $P = \{E_v\}_{v \in A}$. Clearly, the minimum sum taken on E_v in any non-preemptive sum set edge-coloring of $G[N[v]]$ (i.e. the subgraph of G induced by the vertices $N(v) \cup \{v\}$) is $\sum(L(G[N[v]]), x) = \binom{|E_v|+1}{2}$. A non-preemptive edge-coloring Ψ will be called *A-good* if it is P-good for $P = \{E_v\}_{v \in A}$.

We define the tree T_i, for $i \geq 1$, as follows. The tree T_1 is an edge rv, where r is the root vertex and where $x(rv) = 1$. For $i > 1$, the tree T_i is a path on five vertices r, v, v_1, v_2, v_3, being r the root vertex and where $x(rv) = 1$, $x(vv_1) = x(v_2v_3) = i-1$, and $x(v_1v_2) = i$ (see Figure 3). Vertices v and v_2 are in A (black vertices in the figure), the remaining ones are in B. Consider the coloring $\Psi(rv) = i$, $\Psi(vv_1) = \Psi(v_2v_3) = [1, i-1]$, $\Psi(v_1v_2) = [i, 2i-1]$.

This is an A-good coloring, thus it is an optimum coloring and every optimum coloring is A-good. Therefore, if Φ is an optimum coloring for T_i then it must be an A-good coloring with vertices v and v_2 in A, and it is easy to see that this implies edge rv is assigned color i in every optimum coloring.

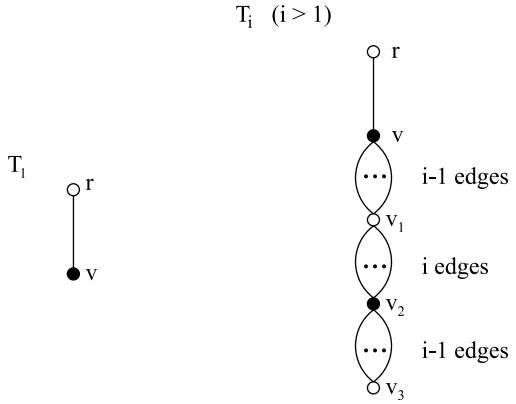

Figure 3: The tree T_i for $i \geq 1$.

A tree $T_{a,b,c}$ (for $a < b < c$) has root r having a single child v with $x(rv) = 1$; vertex v has $c-1$ children $w, y, v_1, \ldots, v_{c-3}$ with $x(vw) = x(vy) = x(vv_1) = \ldots = x(vv_{c-3}) = 1$ (see Figure 4). Every vertex v_j is the root of a T_a, T_b and T_c tree, as defined in the previous paragraph. The black vertices in Figure 4 are in A. Notice that in every A-good (optimum) coloring of $T_{a,b,c}$ the edge rv is colored with color a, b or c, and there are three A-good colorings assigning a, b, and c to edge rv, respectively. The proof of this fact is exactly as it appears in [16].

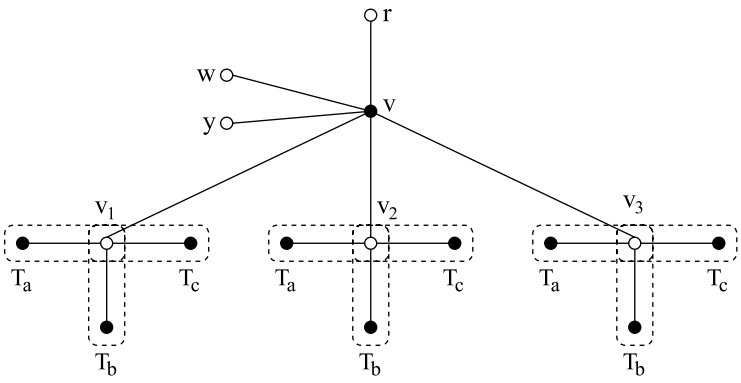

Figure 4: The tree $T_{a,b,c}$ with $c = 6$.

Finally, we define tree $\hat{T}_i$ as follows. The vertex-set of $\hat{T}_i$ is composed by the vertices $r, w, v, v_1, v_2, v_3, v_4$ and v_5 where r is the root vertex, and the edge-set of $\hat{T}_i$ is the set $\{rv, vw, vv_1, v_1v_2, v_2v_3, v_3v_4, v_4v_5\}$, where $x(rv) = x(vw) = 2$, $x(vv_1) = x(v_2v_3) = x(v_4v_5) = i$, $x(v_1v_2) = 4$, and $x(v_3v_4) = i + 4$ (see Figure 5). We show that in every A-good

(optimum) coloring of $\hat{T}_i$ the set of colors assigned to edges between vertices r and v is either $\{i+1, i+2\}$ or $\{i+3, i+4\}$. Let Ψ be an A-good coloring of $\hat{T}_i$. Notice that vertices v, v_2 and v_4 are in A. Thus, $\Psi(v_3v_4)$ can be equal to $[1, i+4]$ or equal to $[i+1, 2i+4]$. However, if $\Psi(v_3v_4) = [1, i+4]$ then $\Psi(v_2v_3)$ must be $[i+5, 2i+4]$, but as v_2 is in A, $i > 0$, and $x(v_1v_2) = 4$ then it contradicts that Ψ is an A-good coloring. Therefore, $\Psi(v_3, v_4) = [i+1, 2i+4]$ which implies that $\Psi(v_4v_5) = \Psi(v_2v_3) = \Psi(vv_1) = [1, i]$, $\Psi(v_1v_2) = [i+1, i+4]$. Moreover, as vertex v is in A and $x(rv) = x(vw) = 2$ then in an A-good non-preemptive coloring for $\hat{T}_i$, we have only two possibilities: $\Psi(rv) = \{i+1, i+2\}$ and $\Psi(vw) = \{i+3, i+4\}$, or $\Psi(rv) = \{i+3, i+4\}$ and $\Psi(vw) = \{i+1, i+2\}$.

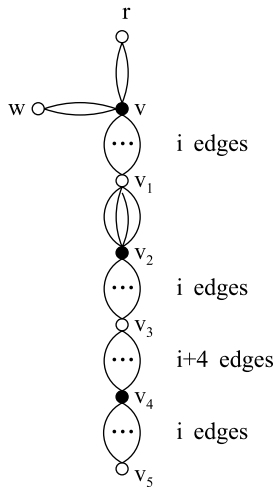

Figure 5: The tree $\hat{T}_i$ for $i \geq 1$.

Now, based in the three families of trees defined previously (i.e., the trees T_i, $T_{a,b,c}$ with $a < b < c$, and $\hat{T}_i$, resp.), we can prove the NP-hardness of the non-preemptive MSSC problem on line graphs of trees. The reduction is from 3-occurrence 3SAT, which is the restriction of 3SAT where every variable occurs at most three times. This problem is NP-complete even if every variable occurs at most twice positively and at most twice negatively (cf. [19]). Given a formula with n variables and m clauses, we construct a tree $T = (V, E)$ and a demand function $x : E \to \mathbb{Z}^+$ such that T has an non-preemptive A-good edge-coloring if and only if the formula is satisfiable. Consider a variable x_k $(0 \leq k < n)$, which is the h-th literal of the i-th clause. Let $d_{i,h}$ be $4k+1$ if this is the first positive occurrence of x_k, $4k+2$ if this is the second positive occurrence, $4k+3$ if this is the first negated occurrence, and $4k+4$ if this is the second negated occurrence. The tree T has a vertex r which is the root of $n+m$ trees (assume that $r \notin A$). To each variable x_j corresponds a tree $\hat{T_{4j}}$, and to each clause i a tree $T_{d_{i,1},d_{i,2},d_{i,3}}$. This defines T and the demand function x. The proof follows exactly as the one given by Marx (see Theorem 3.1 in [16]) for the PSMC problem on line graphs of trees. □

In the following, we will show that under some constraints, there is a pseudo-polynomial

time algorithm to solve the NPMSSC problem on line graphs of trees. Before, we need some preliminaries.

Let m, k be positive integers. Let $J \subseteq [m]$ be a subset of consecutive positive integers. Let $n_1, n_2, \ldots, n_{k+1}$ be positive integers (not necessarily different) such that $\sum_{i=1}^{k+1} n_i \leq m$ and such that $n_{k+1} = |J|$. Let $\mathbb{P}(m, J, n_1, \ldots, n_k)$ be the set where each element is a k-set of intervals of consecutive integers $\{I_1, \ldots, I_k\}$ pairwise disjoint contained in $[m] \setminus J$, such that $|I_i| = n_i$ for $1 \leq i \leq k$. Thus,

Lemma 2. *The cardinality of the set $\mathbb{P}(m, J, n_1, \ldots, n_k)$ is bounded by m^k, and it can be computed in $O(k^2 m^k)$ time.*

Proof. Since the sets $\{I_1, \ldots, I_k\}$ are intervals of consecutive integers, they are univocally defined by their starting point. So there are at most m^k possibilities for choosing the starting points of the k sets within the interval $[m]$ satisfying the constraints above. They can be generated by simple enumeration, and for each of them, the constraints satisfaction can be checked in $O(k^2)$ time. □

Theorem 4. *Let $T = (V, E)$ be a tree with maximum degree equal to Δ and let $x : E \to \mathbb{Z}^+$ be a demand function defined on the edge-set of T. The NPMSSC problem for the line graph $L(T)$ of T can be solved in $O(n\Delta^{\Delta+3}p^{\Delta+1})$ time. In particular, if Δ is bounded by a constant, then it can be solved in pseudo-polynomial time.*

Proof. We propose a dynamic programming algorithm. Let n be the number of vertices of T. We choose as root of T a vertex r with degree equal to 1. For a vertex v in T, we denote T_v the subtree rooted at v. Moreover, for each vertex v in T different from r, we denote by v' the father vertex of v. Finally, for each vertex $v \neq r$, we denote by $T_{v'v}$ the subtree of T formed by T_v to which we join to v its father vertex v'. Given a subtree $T_{v'v}$, we say that v' is its root.

Let C be an upper bound for $C(L(T), x)$. Since $\Delta(L(T)) \leq 2\Delta(T)$, by Lemma 1, we have that $C = O(\Delta p)$. We construct a $n \times C$ table S such that $S[v, j]$ represents the minimum sum for the subtree $T_{v'v}$ when edge $e = v'v$ is assigned the interval $[j, j+x(e)-1]$, for every vertex $v \in T$ and every $1 \leq j \leq C$. First, we define for every vertex v ($v \neq r$) and every $1 \leq j \leq C$ the value $q_v(j)$ which is equal to $j.x(v'v) + \binom{x(v'v)}{2}$ if $j + x(v'v) - 1 \leq C$, otherwise, it is equal to ∞. The algorithm computes the values of table S in a bottom-up way, from the leaves of T up to the root r. If v is a leaf then $S[v, j] = q_v(j)$ for all $1 \leq j \leq C$. Otherwise, let $v \neq r$ be an internal vertex in T and let $v_1, v_2, \ldots, v_k$ be the children of v. Assume that $S[v_i, s]$ is computed, for all $1 \leq i \leq k$ and for all $1 \leq s \leq C$, and we want to compute the value of $S[v, j]$ for a fixed value j such that $j + x(v'v) - 1 \leq C$ (otherwise, $S[v, j] = \infty$). Consider now the set $\mathbb{P}_v(j) = \mathbb{P}(C, [j, j + x(v'v) - 1], x(vv_1), \ldots, x(vv_k))$. If $|\mathbb{P}_v(j)| = 0$ then $S[v, j] = \infty$. Otherwise, let X be an element of the set $\mathbb{P}_v(j)$. By definition, $X = \{I_1, \ldots, I_k\}$, where for each i we have that $|I_i| = x(vv_i)$, I_i is an interval of consecutive integers, $I_i \cap [j, j + x(v'v) - 1] = \emptyset$ and $I_i \cap I_t = \emptyset$ whenever $i \neq t$. For each i, let $\alpha(X, i)$ be the minimum integer in the interval I_i of X. Now, $S[v, j] = q_v(j) + \min_{X \in \mathbb{P}_v(j)}\{\sum_{i=1}^{k} S[v_i, \alpha(X, i)]\}$. Let z be the only children of root vertex r in T.

It is easy to verify that the minimum value of $S[z, j]$, for $1 \leq j \leq C$, is the value of an optimal sum for the NPMSSC of $(L(T), x)$. The overall time complexity of this algorithm can be easily deduced by using Lemma 2. □

5. Open questions

We conclude this paper with the following two open questions.

Question 1. *Let $T = (V, E)$ be a tree with maximum degree equal to Δ and let x be an arbitrary demand function defined on the edge-set of T. Is there a polynomial time algorithm for solving the NPMSSC problem for the line graph $L(T)$ of T when Δ is bounded by a constant?*

Question 2. *Let G be a block graph and let x be an arbitrary demand function defined on the vertices of G. Does there exist a Polynomial Time Approximation Scheme for solving on G either the PMSSC or the NPMSSC problem?*

Acknowledgments

We thank the anonymous referees for their helpful comments that improved the presentation of this paper.

References

[1] A. Bar-Noy, M. Bellare, M. M. Halldórsson, H. Shachnai, and T. Tamir. On chromatic sums and distributed resource allocation. *Information and Computation*, 140(2):183–202, 1998.

[2] A. Bar-Noy and G. Kortsarz. Minimum color sum of bipartite graphs. *Journal of Algorithms*, 28(2):339–365, 1998.

[3] A. Bar-Noy, M. M. Halldórsson, G. Kortsarz, R. Salman, and H. Shachnai. Sum multicoloring of graphs. *Journal of Algorithms*, 37(2):422–450, 2000.

[4] J. Cardinal, V. Ravelomanana, and M. Valencia-Pabon. Minimum sum edge coloring of multicycles. *Discrete Applied Mathematics*, 158(12):1216–1223, 2010.

[5] U. Feige, L. Lovász, and P. Tetali. Approximating min sum set cover. *Algorithmica*, 40(4):219–234, 2004.

[6] R. Gandhi and J. Mestre. Combinatorial algorithms for data migration to minimize average completion time. *Algorithmica*, 54(1):54–71, 2009.

[7] K. Giaro, R. Janczewski, M. Kubale, and M. Malafiejski. A 27/26-approximation algorithm for the chromatic sum coloring of bipartite graphs. In *Proc. of APPROX'02*, LNCS vol. 2486, pp. 135–145. Springer-Verlag, 2002.

[8] K. Giaro and M. Kubale. Edge-chromatic sum of trees and bounded cyclicity graphs. *Information Processing Letters*, 75(1–2):65–69, 2000.

[9] M. M. Halldórsson, G. Kortsarz, A. Proskurowski, R. Salman, H. Shachnai, and J. A. Telle. Multi-coloring trees. *Information and Computation*, 180(2):113–129, 2002.

[10] M. M. Halldórsson, G. Kortsarz, and H. Shachnai. Sum coloring interval and k-claw free graphs with application to scheduling dependent jobs. *Algorithmica*, 37(3):187–209, 2003.

[11] M. M. Halldórsson, G. Kortsarz, and M. Sviridenko. Min sum edge coloring in multigraphs via configuration LP. In *Proc. of IPCO'08*, LNCS vol. 5035, pp. 359–373, 2008.

[12] K. Jansen. Complexity results for the optimum cost chromatic partition problem. In *Proc. of ICALP'97*, LNCS vol. 1256, pp. 727–737, 1997.

[13] E. Kubicka. The Chromatic Sum of a Graph. PhD thesis, Western Michigan University, 1989.

[14] E. Kubicka and A. J. Schwenk. An introduction to chromatic sums. In *Proc. of the 17th ACM Annual Comput. Sci. Conf.*, pp. 39–45, 1989.

[15] D. Marx. Complexity results for minimum sum edge coloring. *Discrete Applied Mathematics*, 157(5):1034–1045, 2009.

[16] D. Marx. Minimum sum multicoloring on the edges of trees. *Theoretical Computer Science*, 361(2-3):133–149, 2006.

[17] D. Marx. The complexity of tree multicolorings. In *Proc. of MFCS'02*, LNCS vol. 2420, pp. 532–542, 2002.

[18] S. Nicoloso, M. Sarrafzadeh, and X. Song. On the sum coloring problem on interval graphs. *Algorithmica*, 23(2):109–126, 1999.

[19] C. H. Papadimitriou. *Computational complexity*. Addison–Wesley, Reading, MA, 1994.

[20] M. Salavatipour. On sum coloring of graphs. *Discrete Applied Mathematics*, 127(3):477–488, 2003.

[21] T. Szkaliczki. Routing with minimum wire length in the dogleg-free manhattan model is NP-complete. *SIAM Journal on Computing*, 29(1):274–287, 1999.

[22] X. Zhou and T. Nishizeki. Algorithm for the cost edge-coloring of trees. *Journal of Combinatorial Optimization*, 8(1):97–108, 2004.