Minimum sum multicoloring on the edges of trees

Dániel Marx [a,1]

[a] *Department of Computer Science and Information Theory, Budapest University of Technology and Economics, H-1521 Budapest, Hungary.*

Abstract

The edge multicoloring problem is that given a graph G and integer demands $x(e)$ for every edge e, assign a set of $x(e)$ colors to edge e, such that adjacent edges have disjoint sets of colors. In the *minimum sum edge multicoloring problem* the *finish time* of an edge is defined to be the highest color assigned to it. The goal is to minimize the sum of the finish times. The main result of the paper is a polynomial-time approximation scheme for minimum sum multicoloring the edges of trees. We also show that the problem is strongly NP-hard for trees, even if every demand is at most 2.

Key words: multicoloring, approximation scheme, edge coloring, trees

1 Introduction

In this paper we study an edge multicoloring problem that is motivated by applications in scheduling. We are given a graph with an integer demand $x(e)$ for each edge e. A *multicoloring* is an assignment of a set of $x(e)$ integer colors to each edge e such that the sets of colors assigned to adjacent edges are disjoint. In multicoloring problems the usual aim is to minimize the total number of different colors used in the coloring. However, in this paper a different optimization goal is studied. Given a multicoloring, the *finish time* of an edge is defined to be the highest color assigned to it. In the *minimum sum edge multicoloring problem* the goal is to minimize the sum of the finish times.

Email address: dmarx@cs.bme.hu (Dániel Marx).

[1] Research is supported in part by grants OTKA 44733, 42559, and 42706 of the Hungarian National Science Fund.

An application of edge coloring is to model dedicated scheduling of biprocessor tasks. The vertices correspond to the processors and each edge $e = uv$ corresponds to a job that requires $x(e)$ time units of simultaneous work on the two preassigned processors u and v. The colors correspond to the available time slots: by assigning $x(e)$ colors to edge e, we select the $x(e)$ time units when the job corresponding to e is executed. 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 finish time of edge e corresponds to the time slot when job e is finished; therefore, minimizing the sum of the finish times is the same as minimizing the sum of completion times of the jobs. Using the terminology of scheduling theory, we minimize the mean flow time, which is a well-studied optimization goal in the scheduling literature. Such biprocessor tasks arise when we want to schedule file transfers between processors [2] or in the mutual diagnostic testing of processors [7]. Note that it is allowed that a job is interrupted and continued later: the set of colors assigned to an edge does not have to be consecutive, hence our problem models preemptive scheduling (we assume that preemptions can happen only at integer times).

Of particular interest is the case where the graph to be colored is bipartite. A possible application of the bipartite problem is the following. One bipartition class corresponds to a set of clients, the other class corresponds to a set of servers. An edge e between two vertices means that the given client has to access the given server for $x(e)$ units of time. A client can access only one server at a time, and a server cannot accept simultaneous connections from two or more clients. Clearly, bipartite edge multicoloring models this situation.

Minimum sum edge multicoloring is NP-hard on bipartite graphs even if every edge has unit demand [3]. In the unit demand case there is a 1.796-approximation algorithm for bipartite graphs [6], and the problem can be solved in polynomial time if the graph is a tree [3,13]. For general demands and general graphs [1] gives a 2-approximation algorithm.

In this paper we consider the minimum sum edge multicoloring problem restricted to trees. We show that, unlike the unit demand case, minimum sum edge multicoloring is NP-hard for trees if the demands are allowed to be at most 2. On the other hand, we also show that the problem is polynomial-time solvable in trees if every demand is the same. This is a consequence of the following scaling property of minimum sum edge multicoloring in trees: if the demand of every edge is multiplied by the same integer q, then the value of an optimum solution increases by a factor of q. (It is easy to see that the sum changes by at most a factor of q, the interesting thing is that this factor is exactly q.) The main contribution of the paper is a polynomial-time approximation scheme (PTAS) for minimum sum edge multicoloring in trees.

Recently, the vertex coloring version of minimum sum multicoloring was investigated by several papers [1,4,5,11], but the edge coloring problem is only mentioned in [1] and [10]. In [4,5] a PTAS is given for the vertex coloring version of the problem in the case when the graph is a tree, a partial k-tree, or a planar graph. One of the main tools used to derive this PTAS is the decomposition of colors into layers of geometrically increasing sizes. This method will be used in this paper as well. However, most of the other tools in [4,5] cannot be applied to our case, since those tools assume that the graph can be colored with a constant number of colors. In our case this is not necessarily true: if the maximum degree of the tree is arbitrary, then the line graph of the tree can contain arbitrarily large cliques. On the one hand, these large cliques make the tools developed for partial k-trees impossible or very difficult to apply. On the other hand, a large clique helps us in finding an approximate solution since in every coloring of a large clique the sum of finish times must be very large; thus, more errors can be tolerated in an approximate solution, and this gives us more room for constructing a good approximation.

The paper is organized as follows. In Section 2 we introduce notation and give some preliminary results. The complexity of the problem is investigated in Section 3. The scaling property for trees is proved in Section 4. Section 5 gives a PTAS for the special case where the maximum degree of the tree is bounded by a constant. Section 6 gives a PTAS for the general case, using the algorithm of the preceding section as a subroutine.

2 Preliminaries

The problem considered in this paper is the edge coloring version of minimum sum multicoloring, which can be stated formally as follows:

Minimum Sum Edge Multicoloring (SEMC)

Input: A graph $G(V, E)$ and a *demand* function x: $E \to \mathbb{N}$.

Output: A *multicoloring* Ψ: $E \to 2^{\mathbb{N}}$ such that $|\Psi(e)| = x(e)$ for every edge e, and $\Psi(e_i) \cap \Psi(e_j) = \emptyset$ if e_i and e_j are adjacent in G.

Goal: The *finish time* of edge e in coloring Ψ is the highest color assigned to it, $f_\Psi(e) = \max\{c : c \in \Psi(e)\}$. The goal is to minimize the sum of the finish times of the edges. The value $f_\Psi(G) = \sum_{e \in E} f_\Psi(e)$ will be called the *sum* of the coloring Ψ.

For brevity, we will use the word "coloring" instead of "multicoloring." We extend the notion of finish time to a set E' of edges by defining $f_\Psi(E') = \sum_{e \in E'} f_\Psi(e)$. Given a graph G and a demand function $x(e)$ on the edges of G, the minimum sum that can be achieved is denoted by $\mathrm{OPT}(G, x)$, or simply

by OPT(G), if the demand function is clear from the context.

In the *non-preemptive* version of the problem we also require that $\Psi(e)$ is a consecutive interval of colors. This paper addresses only the *preemptive* version, where the sets assigned to the edges can be arbitrary. In general, the preemptive and the non-preemptive variants of the same multicoloring problem can be very different (see e.g., [4,5]).

For a minimization problem, algorithm A is an *α-approximation* algorithm if it always produces a solution with cost at most α times the optimum. A *polynomial-time approximation scheme (PTAS)* is an algorithm that has a parameter ϵ such that for every $\epsilon > 0$ it produces an $(1+\epsilon)$-approximate solution, and the running time is polynomial in n (the size of the input) for every fixed ϵ, e.g., $O(n^{1/\epsilon})$. A linear-time PTAS runs in time $O(f(\epsilon)n)$, where f is an arbitrary function. When designing a PTAS, it can be assumed that ϵ is smaller than some fixed constant ϵ_0. In the following, it is assumed that ϵ is sufficiently small and $1/\epsilon$ is an integer.

Henceforth the graph G is a rooted tree with root r. The root is assumed to be a node of degree one, the *root edge* is the edge incident to r. Every edge has an *upper node* (closer to r) and a *lower node* (farther from r). Edge f is a *child edge* of edge e if the upper node of f is the same as the lower node of e. In this case, edge e is the *parent edge* of edge f. A node is a *leaf node* if it has no children, and an edge is a *leaf edge* if its lower node is a leaf node. The subtree T_e consists of the edge e and the subtree rooted at the lower node of e.

A *top-down traversal* of the edges of G is an ordering of the edges such that every edge appears later than its parent edge. Similarly, in a *bottom-up traversal* of the edges every edge appears earlier than its parent. It is clear that such orderings exist and can be found in linear time.

If the tree has maximum degree Δ, then a color from $\{1, 2, \ldots, \Delta\}$ can be assigned to each edge such that adjacent edges have different colors. Fix such a coloring and let the *type* of an edge be its color in this coloring. In some of the algorithms, the leaf edges will be special, and they are handled differently. Therefore, in these cases we assign a type only to the non-leaf edges. Clearly, if every edge has at most D non-leaf child edges, then the non-leaf edges can be given a type from $\{1, 2, \ldots, D+1\}$ so that adjacent edges have different types.

The following lemma bounds the number of colors required in a minimum sum multicoloring. In the following, the maximum demand in the instance is always denoted by p.

Lemma 2.1 *If G is a graph with maximum degree Δ and maximum demand*

p, then every optimum coloring of the SEMC *problem uses at most* $p(2\Delta-1)$ *colors.*

PROOF. Assume that an optimum coloring Ψ uses a color greater than $p(2\Delta-1)$ on the edge $e=uv$. Remove the colors from e. Since at most $\Delta-1$ edges (other than e) are incident to u, with a demand of at most p each, at most $p(\Delta-1)$ colors are used on edges incident to u. Similarly, there are at most $p(\Delta-1)$ colors used on edges incident to v. Therefore, there are at least p colors not greater than $p(2\Delta-1)$ that are used neither on u nor on v. These p colors can be used to color the edge e. This will decrease the finish time of e, contradicting the optimality of Ψ. $\square$

If both the maximum degree of the tree and the maximum demand are bounded by a constant, then the problem can be solved in linear time. The idea is that there are only a constant number of possible color sets that can appear at each edge, hence using standard dynamic programming techniques, the optimum coloring can be found during a bottom-up traversal of the edges.

Theorem 2.2 *The* SEMC *problem for trees can be solved in* $2^{O(p\Delta)}\cdot n$ *time.*

PROOF. Denote by $\mathcal{U}_k^n$ the set of all k element subsets of $\{1,2,\ldots,n\}$. Let T_e be the subtree of T whose root edge is e. Set $m := p(2\Delta-1)$. For every $e\in E(T)$ and $X\in\mathcal{U}_{x(e)}^m$, let $S(e,X)$ denote the value of the optimum sum in the subtree T_e with the further restriction that e is colored by colors from the set X. Clearly, $\mathrm{OPT}(T,x)=\min_{X\in\mathcal{U}_{x(r)}^m} S(r,X)$, since by Lemma 2.1, the root edge r is colored by a set from $\mathcal{U}_{x(r)}^m$ in every optimum coloring.

We determine the values $S(e,X)$ following a bottom-up traversal of the edges. If T_e consists of only the edge e, then $S(e,X)$ is the highest color in X. Now assume that the child edges of e are $e_1,e_2,\ldots,e_k$, and for each $1\le i\le k$, we have already computed a table containing the value of $S(e_i,Y)$ for every $Y\in\mathcal{U}_{x(e_i)}^m$. We would like to determine the value $S(e,X)$ for some set X. One way to do this is to choose (in every possible way) k sets $X_i\in\mathcal{U}_{x(e_i)}^m$ $(1\le i\le k)$. If the sets $X,X_1,X_2,\ldots,X_k$ are pairwise disjoint, then there is a coloring Ψ with $\Psi(e)=X$, $\Psi(e_i)=X_i$ and $f_\Psi(T_e)=\sum_{i=1}^k S(e_i,X_i)+\max_{c\in X} c$. Set $S(e,X)$ to the minimum of this sum for the best choice of the sets $X_1,\ldots,X_k$. It is easy to see that this is indeed the value given by the definition of $S(e,X)$ (by Lemma 2.1, every optimum coloring uses only the colors $1,\ldots,m$).

The method described above solves at most $|\mathcal{U}_p^m|$ subproblems $S(e,X)$ at each edge e. In each subproblem, $k\le\Delta-1$ subsets X_i are chosen in every possible way, the number of combinations considered is at most $|\mathcal{U}_p^m|^{\Delta-1}=$

$O(m^{p(\Delta-1)}) = 2^{O(p\Delta \log(p\Delta))}$. However, using dynamic programming once again, each subproblem $S(e, X)$ can be solved in $2^{O(p\Delta)}$ time. Denote by $T_{e,i}$ the union of the trees T_{e_1}, T_{e_2}, ..., T_{e_i} (the first i children of edge e). For every $Y \subseteq \{1, 2, \ldots, m\}$ and $1 \leq i \leq k$ denote by $P(i, Y)$ the minimum sum on $T_{e,i}$ with the restriction that exactly the colors in Y are used on the edges e_1, ..., e_i. Clearly, $P(1, Y) = S(e_1, Y)$. To calculate $P(i, Y)$ for some $i > 1$ notice that $P(i, Y)$ is the minimum of $S(e_i, X_i) + P(i-1, Y \setminus X_i)$, where the minimum is taken over all $x(e_i)$ size subsets X_i of Y. Finally, $S(e, X)$ can be easily determined by considering every set $Y \subseteq \{1, 2, \ldots, m\}$ disjoint from X, and selecting the one where $P(k, Y)$ is minimal.

At each edge we solve at most $2^m \cdot (\Delta - 1)$ subproblems $P(i, Y)$. To solve a subproblem $P(i, Y)$, we consider $|\mathcal{U}^m_{x(e_i)}| < 2^m$ different sets X_i. Therefore, the total number of combinations considered per edge is $2^{O(m)}$. The work to be done for each combination is polynomial in m, hence it is dominated by $2^{O(m)}$. The number of edges in the tree is $O(n)$, thus the total running time of the algorithm is $2^{O(m)} \cdot n = 2^{O(p\Delta)} \cdot n$. $\square$

In Section 3 we show that if only the demand is bounded, then the problem becomes NP-hard (Theorem 3.1).

In the minimum sum multicoloring problem our goal is to minimize the sum of finish times, not to minimize the number of different colors used. Nevertheless, in Theorem 2.3 we show that the minimum number of colors required for coloring the edges of a tree can be determined by a simple formula. We also show that there is always an optimum solution where the color sets are relatively simple. This result will be used by the approximation algorithm presented in Section 5.

Theorem 2.3 *Let T be a tree and let $C = \max_{v \in V(T)} \sum_{e \ni v} x(e)$. Every coloring of T uses at least C colors, and one can find in linear time a coloring Ψ using C colors where each $\Psi(e)$ consists of at most two intervals of colors. Moreover, if each $x(e)$ is multiple of some integer q, then we can find one Ψ where the intervals in each $\Psi(e)$ are of the form $[qi_1 + 1, qi_2]$ for some integers i_1 and i_2.*

PROOF. It is clear that at least C colors are required in every coloring: there is a vertex v such that the edges incident to v require C different colors. A coloring Ψ satisfying the requirements can be constructed by a simple greedy algorithm. Call a set of colors $S \subseteq [1, C]$ a *circular interval* if it is either an interval $[a, b]$ or the union of two intervals $[1, a] \cup [b, C]$. The algorithm presented below assigns a circular interval of colors to each edge; therefore, each $\Psi(e)$ consists of one or two intervals.

Consider a top-down traversal of the edges. The edges are colored in a greedy manner following this ordering. After each step of the algorithm, the coloring defined so far satisfies the following invariant condition: for every node v, the set of colors used by the edges incident to v forms a circular interval of $[1, C]$.

At the start of the algorithm, we assign the set $[1, x(r)]$ to the root edge r. When an edge e is visited during the traversal, some of the edges incident to the upper node v of e are already colored, and none of the edges incident to the lower node u of e has a color yet. By assumption, the colors used by the edges incident to v form a circular interval S. Clearly, the size of $[1, C] \setminus S$ is at least $x(e)$, otherwise $\sum_{e \ni v} x(e)$ would be strictly greater than C. We can assign to edge e a circular interval S' of size $x(e)$ such that S and S' are disjoint, and $S \cup S'$ is also a circular interval. Thus the set of colors used at v remains a circular interval. Because of the top-down traversal, the set of colors used at u is exactly S', a circular interval, hence the invariant condition remains valid and the algorithm can proceed with the next edge. Moreover, if every demand is an integer multiple of q, then it can be shown by induction that every edge receives a circular interval $\Psi(e)$ such that the one or two intervals in $\Psi(e)$ are of the form $[qi_1 + 1, qi_2]$. $\square$

The fact that for trees a greedy algorithm can minimize the number of colors used was observed in [10]. However, in our applications it will be important that the color sets have the special form described in Lemma 2.3.

3 Complexity

In this section we prove that minimum sum edge multicoloring is NP-hard for trees, even if every demand is 1 or 2. First we give some definitions that will be useful tools for proving the optimality of certain colorings. Then three families of special trees are introduced, they will be used as gadgets in the NP-hardness proof.

Denote by E_v the set of edges incident to v. Let $\ell(v) = \min_\Psi f_\Psi(E_v)$ be the minimum sum taken on the edges incident to v in any proper coloring. If all the edges incident to v have demand 1, then clearly $\ell(v) = \frac{d(v)(d(v)+1)}{2}$. Furthermore, it is easy to see that if one edge incident to v has demand 2 and all the other edges have demand 1, then $\ell(v) = \frac{d(v)(d(v)+1)}{2} + 1$.

Let $G(A, B; E)$ be a bipartite graph. An obvious lower bound for $\mathrm{OPT}(G)$ is $\ell(A) = \sum_{v \in A} \ell(v)$. We call a coloring Ψ *A-good* if $f_\Psi(E) = \ell(A)$, which is equivalent to $f_\Psi(E_v) = \ell(v)$ for every $v \in A$. Every A-good coloring is clearly an optimum coloring, and if there is an A-good coloring, then every optimum

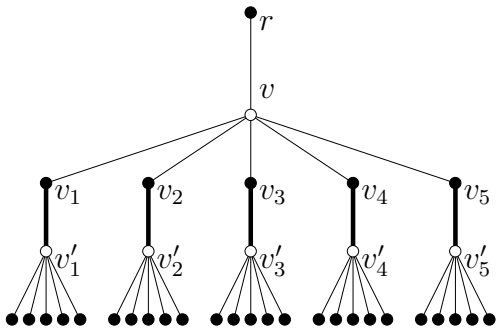

Fig. 1. The tree T_6. The bold edges have demand 2.

coloring is A-good.

We define tree T_i as follows (see Figure 1 for T_6). The root r has a single child v, and node v has $i-1$ children $v_1, v_2, \ldots, v_{i-1}$. Each node v_j has a single child v'_j, and node v'_j has $i-1$ children $v'_{j,1}, \ldots, v'_{j,i-1}$. The edges $v_jv'_j$ have demand 2, the demands of all the other edges are 1. Let the nodes $v, v'_1, \ldots, v'_{i-1}$ be in A (white nodes in the figure), the remaining nodes are in B. Consider the coloring $\Psi(rv) = i$, $\Psi(vv_j) = j$, $\Psi(v_jv'_j) = \{i, i+1\}$, $\Psi(v'_jv'_{j,k}) = k$. 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, then $\Phi(rv) = i$ because $f_\Phi(E_{v'_j}) = \ell(E_{v'_j})$ implies $\Phi(v_jv'_j) = \{i, i+1\}$, and $f_\Phi(E_v) = \ell(v)$ implies that one edge in E_v is colored with color i, which can be only rv. Thus the color of rv is i in every optimum coloring.

A tree $T_{a,b,c}$ (for $a < b < c$) has root r having a single child v; node v has $c-1$ children x, y, v_1, $\ldots$, v_{c-3} (see Figure 2). Every node v_j is the root of a T_a, T_b and T_c tree, as defined in the previous paragraph. We show 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. Color the trees T_a, T_b, T_c at the v_j nodes with an A-good colorin; assign the colors a, b, c to the edges rv, vx, vy in some order, and assign the colors $\{1, \ldots, c\} \setminus \{a, b, c\}$ to the edges $vv_1, \ldots, vv_{c-3}$ in some order. It can be easily verified that this is an A-good (therefore optimum) coloring and the edge rv can have any of the colors a, b, c. To see that in every A-good coloring edge rv can receive only these colors, observe that if the colorings of the T_a, T_b, T_c subtrees rooted at v_j are all A-good, then vv_j cannot be colored with colors a, b, c. In an A-good coloring, the edges incident to v can receive only colors not greater than c, thus rv, vx, vy receive the colors a, b, c. Therefore, rv is colored with either a, b or c.

Finally, we define tree $\widehat{T}_i$ to be a star: the root r has a child v, and node v has $i+1$ children x, v_1, $\ldots$, v_i. The edges rv, vx have demand 2, the other edges have demand 1. The node v is in A. It is easy to see that if Ψ is an A-good coloring, then $\Psi(rv)$ is either $\{i+1, i+2\}$ or $\{i+3, i+4\}$.

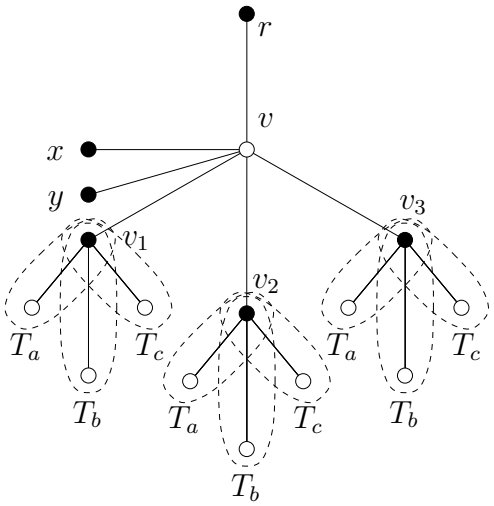

Fig. 2. The tree $T_{a,b,c}$ with $c = 6$

Theorem 3.1 *The* SEMC *problem is* NP*-hard in trees, even if every demand is* 1 *or* 2*.*

PROOF. The reduction is from 3-occurrence 3SAT, which is the restriction of 3SAT where every variable occurs at most three times. This problem is known to be NP-complete even if we assume that every variable occurs at most twice positively and at most twice negatively (cf. [12]). Given a formula with n variables and m clauses, we construct a tree T and a demand function such that the tree has an A-good coloring (i.e., a coloring with sum at most $\ell(A)$) if and only if the formula is satisfiable.

Consider the 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 occurence, and $4k+4$ if this is the second negated occurence. Tree T has a node r which is the root of $n+m$ trees. To each variable x_j corresponds a tree $\widehat{T}_{4j}$, and to each clause i a tree $T_{d_{i,1},d_{i,2},d_{i,3}}$. This defines T and its demand function.

Assume that a coloring is A-good, then it is an A-good coloring of all the $n+m$ subtrees (since $r \notin A$). Therefore, the root edge of $\widehat{T}_{4j}$ corresponding to variable x_j uses either the set $\{4j+1, 4j+2\}$ or the set $\{4j+3, 4j+4\}$. Assign to the variable x_j the value "false" in the first case and "true" in the second case. This will be a satisfying assignment: if the root edge of the tree corresponding to clause i uses a color from $\{4j+1, 4j+2, 4j+3, 4j+4\}$, then variable x_j satisfies clause i. More precisely, if it uses $4j+1$ or $4j+2$ (resp., $4j+3$ or $4j+4$), then x_j has the value "true" (resp., "false"), and by construction, x_j appears in clause i positively (resp., negatively).

To prove the other direction, given a satisfying assignment, we construct an A-good coloring of the tree. Take an A-good coloring of the subtree $\widehat{T}_{4j}$ corresponding to variable x_j such that its root edge uses the colors $\{4j+1, 4j+2\}$ (resp., $\{4j+3, 4j+4\}$) if x_j is "false" (resp., "true"). Since every clause is satisfied by some variable, we can choose an A-good coloring for each subtree corresponding to a clause such that it does not conflict with any of the trees corresponding to the variables. Clearly, this will be an A-good coloring of the tree.

We have reduced a known NP-complete problem to the minimum sum edge multicoloring problem. The reduction can be done in polynomial time, thus SEMC is NP-hard. □

We note that in the proof, the optimum solution colors non-preemptively every edge with demand 2. Thus the reduction works even if we impose the additional constraint of non-preemptive coloring.

Corollary 3.2 *The non-preemptive version of* SEMC *is* NP*-hard for trees, even if every demand is* 1 *or* 2. □

4 Scaling and rounding

Consider an instance of SEMC: let G be a graph, and let $x(e)$ be an arbitrary demand function on the edges of G. Multiply the demand of every edge by an integer q, that is, consider the demand function $x'(e) = q \cdot x(e)$. The first observation is that this operation increases the minimum sum by at most a factor q, that is,

$$\mathrm{OPT}(G, x') \leq q \cdot \mathrm{OPT}(G, x). \tag{1}$$

To see this, take an optimum coloring Ψ of (G, x), and replace every color by q consecutive colors: for every $c \in \Psi(e)$, add $\{(c-1)q+1, (c-1)q+2, \ldots, cq\}$ to $\Phi(e)$. Clearly, coloring Φ satisfies the demand x' on every edge, and the finish time of every edge in Φ is exactly q times larger than in Ψ. Therefore, the sum of Φ is exactly q times larger than the optimum sum of (G, x), and (1) follows.

We mention it without proof that one can construct a bipartite graph G, and choose x, q in such a way that (1) holds with strict inequality. The aim of this section is to show that if G is a tree, then there is always equality in (1):

Theorem 4.1 *For every tree $T(V, E)$, demand function x, and integer q, if $x'(e) = q \cdot x(e)$ for every $e \in E$, then* $\mathrm{OPT}(T, x') = q \cdot \mathrm{OPT}(T, x)$.

Before proving Theorem 4.1, we have to make some preparations. The following problem is the weighted version of multicoloring (this problem is studied in [8,9] under the name Generalized Optimum Cost Chromatic Partition). Every vertex has a cost function (thus the same color can have different costs at different vertices), and the goal is to minimize the total cost of the colors used in the coloring. As in the case of other coloring problems, we consider here the edge coloring version:

Generalized Minimum Cost Edge Multicoloring

Input: A graph $G(V, E)$, a *demand* function x: $E \to \mathbb{N}$, a set of available colors $\mathcal{C} = \{1, 2, \ldots, C\}$, and a list of costs $c_{e,i}$ (for every $e \in E$, $i \in \mathcal{C}$).

Output: A *multicoloring* Ψ: $E \to 2^{\mathcal{C}}$ such that $\Psi(e_i) \cap \Psi(e_j) = \emptyset$ if e_i and e_j are adjacent in G and $|\Psi(e)| = x(e)$.

Goal: Minimize the total cost $\sum_{e \in E} \sum_{i \in \Psi(e)} c_{e,i}$.

This problem can be formulated as an integer linear programming problem as follows. Variable $y_{e,i}$ represents the choice of assigning color i to edge e: the value of $y_{e,i}$ is 1 if i is assigned to e, and 0 otherwise. It is easy to verify that the integer solutions of the following linear program correspond to the proper colorings of the graph:

$$\text{minimize} \sum_{e \in E} \sum_{i=1}^{C} c_{e,i} y_{e,i}$$

$$\text{s. t.}$$

$$y_{e,i} \geq 0 \qquad \forall e \in E,\ 1 \leq i \leq C \qquad (2)$$

$$\sum_{e \ni v} y_{e,i} \leq 1 \qquad \forall v \in V,\ 1 \leq i \leq C \qquad (3)$$

$$\sum_{i=1}^{C} y_{e,i} \geq x(e) \qquad \forall e \in E \qquad (4)$$

The inequalities (3) express the requirement that a color i can appear at most once on the edges incident to v, while inequalities (4) ensure that edge e receives at least $x(e)$ colors. The cost of an integer solution equals the cost of the corresponding coloring. In general, this linear program does not necessarily have an integer optimum solution, but if the graph is a tree, then there is always an integer optimum:

Lemma 4.2 *For every tree T with arbitrary demand function x and costs $c_{e,i}$ the linear program has an integer optimum solution.*

PROOF. We show that the coefficient matrix of the linear program is a

network matrix, hence it is totally unimodular. The right-hand side of the linear program is an integer vector, thus the lemma follows from the well-known properties of totally unimodular matrices (cf. [14]).

Recall the definition of network matrices. Let D be a directed graph and T be a spanning tree over the same vertex set V. Denote by n and m the number of edges of T and D, respectively. Direct the edges of T arbitrarily. Consider an $n \times m$ matrix M whose rows correspond to the edges of T and columns correspond to the edges of D. Every directed edge e in D determines a unique path in the tree. If edge f of T lies on this path and its orientation agrees with the directed path, then let the element of M in row f and column e be 1; if its orientation is opposite, then let the element be -1. If f does not lie on the path determined by e, then the element is zero. A matrix M that arises in such a way from some T and D is called a *network matrix*. It is well known that every network matrix is totally unimodular (cf. [14]).

The constraints (2) do not really matter: if an $n \times m$ matrix is totally unimodular, then it remains totally unimodular after appending an $m \times m$ unit matrix to it. To show that the coefficient matrix corresponding to (3) and (4) is a network matrix, we construct a tree T' and a directed graph D such that the edges of T' (resp., D) correspond to the rows (resp., columns) of the matrix. Denote the inequalities in (3) by $d_{v,i}$ ($v \in V$, $1 \leq i \leq C$) and those in (4) by d_e. Let V_1 and V_2 be the two bipartition classes of T. Direct the edges of T from V_1 to V_2; edge $e \in E$ will corresponds to row d_e. Connect C new vertices v_i ($1 \leq i \leq C$) to every vertex v; the C new edges correspond to the rows $d_{v,i}$. Direct these new edges away from v if $v \in V_1$, and to v if $v \in V_2$. Call the resulting tree T'. The directed graph D is defined as follows: if $e = uv$ ($u \in V_1$, $v \in V_2$) is an edge in T, then add the edges $y_{e,i} = \overrightarrow{v_i u_i}$ ($1 \leq i \leq C$) to D. Now it can be verified that the network matrix corresponding to T' and D is the coefficient matrix of the linear program: the unique path corresponding to edge $y_{e,i} = \overrightarrow{v_i u_i}$ contains the edges $d_{v,i}, d_e, d_{u,i}$, and the variable $y_{e,i}$ appears precisely in these inequalities. □

Now we are ready to prove the main result of the section:

PROOF (of Theorem 4.1). Given a coloring Ψ with value $\mathrm{OPT}(T, x')$, we construct a coloring Φ that satisfies the demand function x and has sum at most $\mathrm{OPT}(T, x')/q$. Define the following cost function:

$$c_{e,i} = \begin{cases} 0 & \text{if } i < \lceil f_\Psi(e)/q \rceil, \\ 1 & \text{if } i = \lceil f_\Psi(e)/q \rceil, \\ 2|E| & \text{if } i > \lceil f_\Psi(e)/q \rceil. \end{cases}$$

Consider the generalized minimum cost multicoloring problem on the edges of T, with demand $x(e)$ and color costs $c_{e,i}$. Let C, the number of colors, be an integer larger then the total demand of the tree T. It is clear that the linear program given by inequalities (2)–(4) always has a feasible solution: since C is large enough, the demands can be satisfied even if every color is used at most once. By Lemma 4.2, this program has an integer optimum solution with costs $c_{e,i}$, let $y_{e,i}$ be such a solution. It is easy to see that every variable is 0 or 1. Define coloring Φ with $i \in \Phi(e)$ if and only $y_{e,i} = 1$. Replace every color in the coloring Φ with a sequence of q colors to obtain a coloring Φ', that is, if $i \in \Phi(e)$, then add $\{(i-1)q+1, (i-1)q+2, \ldots, iq\}$ to $\Phi'(e)$. Clearly, $f_{\Phi'}(T) = q \cdot f_\Phi(T)$. Therefore, if it can be shown that $f_{\Phi'}(T) \leq f_\Psi(T) = \mathrm{OPT}(T, x')$, then $\mathrm{OPT}(T, x) \leq f_\Phi(T) = f_{\Phi'}(T)/q \leq \mathrm{OPT}(T, x')/q$ and Theorem 4.1 follows.

Let $z_{e,i}$ be $|\Psi(e) \cap \{(i-1)q+1, (i-1)q+2, \ldots, iq\}|/q$. It can be easily verified that this is a feasible solution of the linear program. Furthermore, the cost of this solution is strictly less than $2|E|$ since, by definition, $z_{e,i}$ is 0 if $i > \lceil f_\Psi(e)/q \rceil$. Therefore, the optimum integral solution $y_{e,i}$ has cost strictly less than $2|E|$, which implies that $y_{e,i} = 0$ for $i > \lceil f_\Psi(e)/q \rceil$, and $f_\Phi(e) \leq \lceil f_\Psi(e)/q \rceil$ follows.

Let $c_\Phi(e) = \sum_{i=1}^{C} c_{e,i} y_{e,i}$ and $c_\Psi(e) = \sum_{i=1}^{C} c_{e,i} z_{e,i}$. We show that for every edge e,

$$f_\Phi(e) \leq f_\Psi(e)/q + c_\Phi(e) - c_\Psi(e),$$

or equivalently

$$f_{\Phi'}(e) \leq f_\Psi(e) + q(c_\Phi(e) - c_\Psi(e)).$$

If the latter inequality holds, then summing for every $e \in E$ gives $f_{\Phi'}(E) \leq f_\Psi(E) + q(\sum_{e \in E} c_\Phi(e) - \sum_{e \in E} c_\Psi(e))$. From the fact that $y_{e,i}$ is an optimum solution of the linear program with costs $c_{e,i}$, it follows that $\sum_{e \in E} c_\Phi(e) \leq \sum_{e \in E} c_\Psi(e)$. This implies $f_{\Phi'}(E) \leq f_\Psi(E)$, proving the theorem.

There are two cases to consider: (a) $f_\Phi(e) = \lceil f_\Psi(e)/q \rceil$ and (b) $f_\Phi(e) < \lceil f_\Psi(e)/q \rceil$ (we have seen that $f_\Phi(e) \leq \lceil f_\Psi(e)/q \rceil$ for every edge e). Since $\Psi(e)$ contains at most $f_\Psi(e) - (\lceil f_\Psi(e)/q \rceil q - q)$ colors greater than $\lceil f_\Psi(e)/q \rceil q - q$, we have that

$$q \cdot c_\Psi(e) \leq f_\Psi(e) - (\lceil f_\Psi(e)/q \rceil q - q)$$

and

$$f_\Psi(e)/q - c_\Psi(e) \geq \lceil f_\Psi(e)/q \rceil - 1.$$

If (a) holds, then $c_\Phi(e) = 1$, thus $f_\Psi(e)/q + c_\Phi(e) - c_\Psi(e) \geq \lceil f_\Psi(e)/q \rceil = f_\Phi(e)$, as required. In case (b), $c_\Phi(e) = 0$, which implies $f_\Psi(e)/q + c_\Phi(e) - c_\Psi(e) \geq \lceil f_\Psi(e)/q \rceil - 1 \geq f_\Phi(e)$, what we had to prove. □

In Section 3, we have shown that the preemptive minimum sum edge coloring problem is NP-hard in trees even if every demand is 1 or 2. However, it becomes

polynomial-time solvable if every demand is 2, or more generally, if every edge has the same demand. By Theorem 4.1, the case where every edge has the same demand can be reduced to the case where every edge has unit demand, which is polynomial-time solvable [3,13].

Corollary 4.3 *The SEMC problem can be solved in polynomial time in trees if every edge has the same demand.* □

The following lemma is another corollary of Theorem 4.1: if the demand of every edge is increased to at most λ times the original demand, then the optimum increases by at most a factor of λ. This is trivial to show if λ is integer (replace every color in the optimum coloring by λ consecutive colors), but the lemma states that in trees this is true even if λ is not an integer. This observation will be used in Section 6.

Lemma 4.4 *Let (T, x) be an instance of* SEMC *where T is a tree, and let λ be a positive rational number. If x' is a demand function with $x'(e) \leq \lambda \cdot x(e)$ for every edge e, then* $\mathrm{OPT}(T, x') \leq \lambda \cdot \mathrm{OPT}(T, x)$.

PROOF. Assume that $\lambda = a/b$ for some integers a and b. Let $x_2(e) = a \cdot x(e)$; by Theorem 4.1, $\mathrm{OPT}(T, x_2) = a \cdot \mathrm{OPT}(T, x)$. Round x_2 down to the nearest integer multiple of b, denote by x_3 the resulting demand function. Let $x_4(e) = x_3(e)/b = \lfloor ax(e)/b \rfloor \geq \lfloor x'(e) \rfloor = x'(e)$. By Theorem 4.1, $\mathrm{OPT}(T, x_4) = \mathrm{OPT}(T, x_3)/b \leq \mathrm{OPT}(T, x_2)/b = (a/b)\mathrm{OPT}(T, x) = \lambda \cdot \mathrm{OPT}(T, x)$. Thus $x_4(e) \geq x'(e)$ implies $\mathrm{OPT}(T, x') \leq \mathrm{OPT}(T, x_4) \leq \lambda \cdot \mathrm{OPT}(T, x)$. □

5 Bounded degree

If a tree T has maximum degree Δ, then the line graph of T is a partial $(\Delta-1)$-tree. Halldórsson and Kortsarz [4] gave a PTAS with running time $n^{O(k^2/\epsilon^5)}$ for minimum sum multicoloring the vertices of partial k-trees; therefore, there is a PTAS for SEMC in bounded degree trees as well. However, the method can be made simpler and more efficient in line graphs of trees. In this section we present a linear-time PTAS for SEMC in bounded degree trees, which makes use of the special structure of trees. Furthermore, our algorithm works even in the more general class of *almost bounded degree trees*: in trees that become of bounded degree after deleting the degree one nodes. Equivalently, we can say that a tree is an almost bounded degree tree if every node has at most a bounded number of non-leaf child edges.

Most of the ideas presented in this section are taken from [4], with appropriate modifications. In Section 6 a PTAS is given for general trees, which uses the

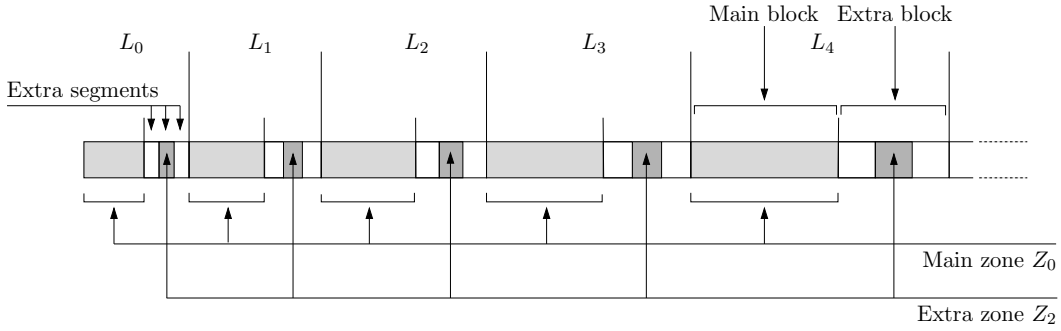

Fig. 3. The decomposition of colors into layers ($\ell = 3$)

result in this section as a subroutine.

5.1 Layers and zones

An important idea of the approximation schemes given in [4,5] is to divide the color spectrum into geometrically increasing layers, and to solve the problem in these layers separately. We use a similar method for the SEMC problem in bounded degree trees (Theorem 5.4) and general trees (Theorem 6.1).

For some $\epsilon > 0$ and integer $\ell \geq 0$, the (ϵ, ℓ)-*decomposition* divides the infinite set of colors into *layers* $L_0, L_1, \ldots$ and *zones* $Z_0, Z_1, \ldots, Z_\ell$. The layers are of geometrically increasing sizes: layer L_i contains the range of colors from q_i to $q_{i+1} - 1$, where $q_i = \lfloor (1+\epsilon)^i \rfloor$. If $q_i = q_{i+1}$, then layer L_i is empty. Denote by $Q_i = |L_i| = q_{i+1} - q_i$ the size of the i-th layer. The total size of layers L_0, L_1, $\ldots$, L_i is $q_{i+1} - 1$. Later we will use that $(1+2\epsilon)q_i \geq q_{i+1} - 1$:

$$\begin{aligned}(1+2\epsilon)q_i &> (1+\epsilon)((1+\epsilon)^i - 1) + \epsilon q_i = (1+\epsilon)^{i+1} - 1 - \epsilon + \epsilon q_i \\ &\geq (1+\epsilon)^{i+1} - 1 \geq q_{i+1} - 1.\end{aligned} \tag{5}$$

That is, if we replace a color from layer L_i with another color from L_i, then the new color is at most $(1+2\epsilon)$ times larger than the original.

Layer L_i is divided into two parts: the first $\frac{1}{1+\epsilon\ell} Q_i$ colors form the *main block* of layer L_i and the remaining $\frac{\epsilon\ell}{1+\epsilon\ell} Q_i$ colors the *extra block* (see Figure 3). Taking the union of the main block of every layer gives the *main zone* Z_0. Divide the extra block of every layer L_i into ℓ equal parts: these are the ℓ *extra segments* of L_i. The union of the j-th extra segment of every layer L_i forms the j-th *extra zone* Z_j. Each extra zone contains $\frac{\epsilon}{1+\epsilon\ell} Q_i$ colors from layer L_i.

Rounding problems will be handled as follows. Layer i is divided such that the first $g_{i,0}$ ($\approx \frac{1}{1+\epsilon\ell} Q_i$) colors are assigned to the main zone Z_0, and extra zone Z_j receives $g_{i,j}$ ($\approx \frac{\epsilon}{1+\epsilon\ell} Q_i$) colors. In Lemma 5.1 we show that the values $g_{i,j}$ can be determined in such a way that the resulting zones approximate reasonably well the "ideal" case where the main zone contains exactly a $\frac{1}{1+\epsilon\ell}$ fraction of

the color spectrum, and each extra zone contains exactly a $\frac{\epsilon}{1+\epsilon\ell}$ fraction of the colors. In particular, we will need the following properties of the defined zones:

Lemma 5.1 *For given ℓ and $\epsilon \leq \frac{1}{2\ell}$, one can calculate values $g_{i,j}$ such that the resulting (ϵ, ℓ)-decomposition of the colors has the following properties:*

(a) For every $c \geq 1$, main zone Z_0 contains at least c colors not greater than $\lfloor (1+\epsilon\ell)c \rfloor$.
(b) For every $c \geq 1$ and $1 \leq j \leq \ell$, extra zone Z_j contains at least c colors not greater than $\frac{2c}{\epsilon}$.

Moreover, each value $g_{i,j}$ can be calculated using a constant number of arithmetic operations.

Intuitively, these statements are clear: if the main zone contains a $\frac{1}{1+\epsilon\ell}$ fraction of the color spectrum, then there are $\frac{1}{1+\epsilon\ell} \cdot (1+\epsilon\ell)c = c$ colors below $(1+\epsilon\ell)c$. Furthermore, each extra zone contains a $\frac{\epsilon}{1+\epsilon\ell}$ fraction of the colors, hence there are at least c colors below $\frac{1+\epsilon\ell}{\epsilon}c \leq \frac{2c}{\epsilon}$. However, the formal proof of Lemma 5.1 requires a tedious calculation to properly handle rounding problems (see below).

Given a multicoloring Ψ, the operation (ϵ, ℓ)-*augmentation* creates a multicoloring Φ the following way. Consider the (ϵ, ℓ)-decomposition of the colors, and if $\Psi(e)$ contains color c, then let $\Phi(e)$ contain instead the c-th color from the main zone Z_0. By Lemma 5.1a, $f_\Phi(e) \leq \lfloor (1+\epsilon\ell) f_\Psi(e) \rfloor$, thus this operation increases the sum by at most a factor of $(1+\epsilon\ell)$. After the augmentation, the colors of the extra zones are not used, only the colors of the main zone.

The rest of this section is devoted to the proof of Lemma 5.1. First set

$$g_{i,0} = \lceil \frac{1}{1+\epsilon\ell}(q_{i+1}-1) \rceil - \lceil \frac{1}{1+\epsilon\ell}(q_i - 1) \rceil \leq \lceil \frac{1}{1+\epsilon\ell} Q_i \rceil.$$

The inequality follows from $\lceil a \rceil + \lceil b \rceil \geq \lceil a+b \rceil$. This ensures that $\sum_{k=0}^{i} g_{k,0} = \lceil \frac{1}{1+\epsilon\ell}(q_{i+1}-1) \rceil$. Now there remain $g_i = Q_i - g_{i,0}$ colors for the extra zones in layer L_i. The following lemma shows that these colors can be evenly divided among the ℓ layers:

Lemma 5.2 *If ℓ and g_i $(0 \leq i \leq n)$ are nonnegative integers, then there are nonnegative integers $g_{i,j}$ $(0 \leq i \leq n,\ 1 \leq j \leq \ell)$ such that for every i and j*

$$\sum_{k=0}^{i} g_{k,j} \geq \left\lfloor \frac{1}{\ell} \sum_{k=0}^{i} g_k \right\rfloor \tag{6}$$

and

$$\sum_{j=1}^{\ell} g_{i,j} \leq g_i \tag{7}$$

hold. Moreover, if $\sum_{k=0}^{i} g_k$ can be calculated with a constant number of arithmetic operations, then each $g_{i,j}$ can be also calculated with a constant number of arithmetic operations.

PROOF. We calculate $g_{i,j}$ by determining the values $G_{i,j} = \sum_{k=0}^{i} g_{k,j}$, then $g_{i,j}$ can be obtained as $g_{i,j} = G_{i,j} - G_{i-1,j}$. Let

$$G_{i,j} = \begin{cases} \lceil \frac{1}{\ell} \sum_{k=0}^{i} g_k \rceil & \text{if } j \leq \sum_{k=0}^{i} g_k - \ell \lfloor \frac{1}{\ell} \sum_{k=0}^{i} g_k \rfloor, \\ \lfloor \frac{1}{\ell} \sum_{k=0}^{i} g_k \rfloor & \text{otherwise.} \end{cases}$$

Clearly, $\sum_{j=1}^{\ell} G_{i,j} = \sum_{k=0}^{i} g_k$. It is clear that (6) holds, since $G_{i,j} \geq \lfloor \frac{1}{\ell} \sum_{k=0}^{i} g_k \rfloor$. Furthermore, (7) also holds:

$$\sum_{j=1}^{\ell} g_{i,j} = \sum_{j=1}^{\ell} (G_{i,j} - G_{i-1,j}) = \sum_{k=0}^{i} g_k - \sum_{k=0}^{i-1} g_k = g_i.$$

Each $G_{i,j}$ can be calculated from $\sum_{k=0}^{i} g_k$ by a constant number of arithmetic operations, and this is true also for $g_{i,j} = G_{i,j} - G_{i-1,j}$, hence the claim of the lemma follows. □

PROOF (of Lemma 5.1). Consider the values $g_{i,j}$ given by Lemma 5.2. To verify property (a), notice that for every $d \geq 1$, there are at least $\lceil \frac{1}{1+\epsilon\ell} d \rceil$ colors in the main zone not greater than d. Indeed, if $d = q_{i+1} - 1$ (d is the last color of layer L_i), then this follows from the way $g_{i,0}$ was defined, otherwise it follows from the fact that the main zone uses the first $g_{i,0}$ colors of layer L_i, hence if it is true for $d = q_i - 1$ and $d = q_{i+1} - 1$, then it is true for every value in between. Thus there are at least $\lceil \frac{1}{1+\epsilon\ell} \lfloor (1+\epsilon\ell)c \rfloor \rceil \geq c$ colors below $\lfloor (1+\epsilon\ell)c \rfloor$.

To verify property (b), assume that $q_i - 1 < \frac{1+\epsilon\ell}{\epsilon} \cdot c \leq q_{i+1} - 1$ for some i. Since $\frac{1+\epsilon\ell}{\epsilon} \cdot c$ is greater than $1/\epsilon$, we have

$$(1+2\epsilon) \cdot \frac{1+\epsilon\ell}{\epsilon} \cdot c \geq \frac{1+\epsilon\ell}{\epsilon} \cdot c + 2 \geq q_i + 1 > (1+\epsilon)^i.$$

Multiplying by $1+\epsilon$ we get

$$(1+\epsilon) \cdot (1+2\epsilon) \cdot \frac{1+\epsilon\ell}{\epsilon} \cdot c \geq (1+\epsilon)^{i+1} > q_{i+1} - 1.$$

If $\epsilon \leq \frac{1}{2\ell}$ is sufficiently small, then $q_{i+1} - 1 \leq \frac{2}{\epsilon} \cdot c$ follows. We use Lemma 5.2 to calculate the number of colors in the first i layers (i.e., up to color $q_{i+1} - 1$) that belong to zone Z_j:

$$\begin{aligned}\sum_{k=0}^{i} g_{k,j} &\geq \left\lfloor \frac{1}{\ell} \sum_{k=0}^{i} g_k \right\rfloor = \left\lfloor \frac{1}{\ell} \sum_{k=0}^{i} (Q_k - g_{k,0}) \right\rfloor = \left\lfloor \frac{1}{\ell} \left(\sum_{k=0}^{i} Q_k - \left\lceil \sum_{k=0}^{i} \frac{1}{1+\epsilon\ell} Q_k \right\rceil \right) \right\rfloor \\ &= \left\lfloor \frac{1}{\ell} \left\lfloor \sum_{k=0}^{i} \frac{\epsilon\ell}{1+\epsilon\ell} Q_k \right\rfloor \right\rfloor \geq \left\lfloor \frac{1}{\ell} \cdot \ell \left\lfloor \sum_{k=0}^{i} \frac{\epsilon}{1+\epsilon\ell} Q_k \right\rfloor \right\rfloor = \left\lfloor \sum_{k=0}^{i} \frac{\epsilon}{1+\epsilon\ell} Q_k \right\rfloor \\ &= \left\lfloor \frac{\epsilon}{1+\epsilon\ell} (q_{i+1} - 1) \right\rfloor \geq \left\lfloor \frac{\epsilon}{1+\epsilon\ell} \cdot \frac{1+\epsilon\ell}{\epsilon} \cdot c \right\rfloor = c\end{aligned}$$

Therefore, there are at least c colors in zone Z_j not greater than $q_{i+1} - 1 \leq \frac{2}{\epsilon} \cdot c$, proving property (b). □

5.2 PTAS for bounded degree trees

The polynomial-time algorithm of Theorem 2.2 was based on the observation that we have to consider only a constant number of different colorings at each edge if both the demand and the maximum degree are bounded. In general, however, the number of different color sets that can be assigned to an edge is exponentional in the demand. The main idea of the PTAS in [4] for vertex coloring partial k-trees is that one can select a polynomial number of color sets for each vertex in such a way that there is a good approximate coloring using only these sets. This gives a PTAS, since the best coloring that uses only the selected sets can be found in polynomial time with standard dynamic programming techniques.

Here we also follow this path: Lemma 5.3 shows that one can find a good approximate coloring by considering only a constant number of different color sets at each edge. Combining this with a dynamic programming algorithm similar to that in the proof of Theorem 2.2, results in a linear-time PTAS for the problem.

Recall that if every node has at most D non-leaf child edges, then the non-leaf edges can be divided into $D + 1$ types such that edges of the same type are not adjacent.

Lemma 5.3 *If each vertex of the tree T has at most D non-leaf child edges and $\epsilon \leq \frac{1}{3D}$, then it has a $(1+3D\epsilon)$-approximate coloring Ψ with the following properties:*

(1) In the $(\epsilon, D+1)$-decomposition of the colors, if e is a non-leaf edge, then $\Psi(e)$ contains colors from the main zone only between $\frac{\epsilon}{4} \cdot x(e)$ and $\frac{2}{\epsilon} \cdot x(e)$.

(2) If e is a non-leaf edge of type k, then $\Psi(e)$ contains the first t_e colors from extra zone Z_k (for some t_e), and it does not contain colors from the other extra zones.
(3) If e is a leaf edge, then $\Psi(e)$ contains colors only from the main zone.
(4) If e is a non-leaf edge, then $\Psi(e)$ contains at most two continuous intervals of colors from the main block of each layer.

PROOF. Let Φ be an optimum solution, and let Ψ be the result of an $(\epsilon, D+1)$-augmentation on Φ. By Lemma 5.1a, $f_\Psi(e) \leq (1+(D+1)\epsilon)f_\Phi(e)$ for every e (note that we assumed $\epsilon \leq \frac{1}{3D} < \frac{1}{2(D+1)}$).

If $f_\Psi(e) > \frac{2}{\epsilon} \cdot x(e)$ for a non-leaf edge e of type k, then modify $\Psi(e)$ to be the first $x(e)$ colors of extra zone Z_k. By Lemma 5.1b, Z_k contains at least $x(e)$ colors not greater than $\frac{2}{\epsilon} \cdot x(e)$. Therefore, the $x(e)$ colors assigned to e are not greater than $\frac{2}{\epsilon} \cdot x(e)$, implying that $f_\Psi(e) \leq (1+(D+1)\epsilon)f_\Phi(e)$. In this case requirements 2 and 4 are automatically satisfied for e, thus there is nothing else to do with this edge.

If $\Psi(e)$ contains colors in the main zone below $\frac{\epsilon}{4} \cdot x(e)$, then delete these colors and let $\Psi(e)$ contain instead the first $\frac{\epsilon}{4} \cdot x(e)$ colors from zone Z_k. There are at least $\frac{\epsilon}{4} \cdot x(e)$ colors in Z_i below $\frac{2}{\epsilon} \cdot \frac{\epsilon}{4} \cdot x(e) = x(e)/2$. The finish time of e is at least $x(e)$, hence this modification does not increase the finish time of e. Therefore, Ψ satisfies the first three properties of the lemma.

Finally, we make Ψ satisfy the fourth requirement as well. For each non-leaf edge e, define $x_i(e)$ to be the number of colors in $\Psi(e)$ that belong to the main block of L_i, rounded down to the next integer multiple of $\lceil \epsilon^2 Q_i/8 \rceil$. If we use x_i as a demand function on the non-leaf edges of the tree, then there is a coloring satisfying x_i that uses only the main block of L_i colors: $\Psi(e)$ restricted to the main block of L_i gives such a coloring. Every $x_i(e)$ is an integer multiple of $\lceil \epsilon^2 Q_i/8 \rceil$; therefore, by Theorem 2.3, it can be assumed that each $\Psi_i(e)$ consists of at most two intervals of the form $[1 + j_1\lceil \epsilon^2 Q_i/8 \rceil, j_2\lceil \epsilon^2 Q_i/8 \rceil]$ for some j_1, j_2. Modify coloring Ψ: let Ψ_i determine how the colors are assigned in the main block of layer i. Now the third requirement is satisfied, but it is possible that Ψ assigns fewer than $x(e)$ colors to an edge. We can lose at most $\lceil \epsilon^2 Q_i/8 \rceil - 1 < \epsilon^2 Q_i/8$ colors in layer i, hence we lose at most a $\frac{\epsilon^2}{8}$ fraction of each layer. Assume that the highest color of $\Psi(e)$ is in layer L_i. Since $\Psi(e)$ contains colors only up to $\frac{2}{\epsilon} \cdot x(e)$, the last color of layer L_i is less than $(1+2\epsilon) \cdot \frac{2}{\epsilon} \cdot x(e) \leq \frac{4}{\epsilon} \cdot x(e)$ (Inequality (5)). Thus we lose only at most $\frac{\epsilon^2}{8} \cdot \frac{4}{\epsilon} \cdot x(e) = \frac{\epsilon}{2} \cdot x(e)$ colors. If non-leaf edge e is of type k, then we use extra zone Z_k to replace the lost colors. So far, edge e uses at most $\frac{\epsilon}{2} \cdot x(e)$ colors from Z_k (previous paragraph), hence there is still place for at least $\frac{\epsilon}{2} \cdot x(e)$ colors in Z_k below $(1+(D+1)\epsilon)x(e) \leq (1+(D+1)\epsilon)f_\Phi(e)$.

The modification in the previous paragraph can change the finish times of the non-leaf edges, but the largest color of each edge remains in the same layer. By Inequality (5), $(1+2\epsilon)q_i \geq q_{i+1} - 1$, therefore the finish time of an edge can increase by at most a factor of $(1+2\epsilon)$. Moreover, since we modified only the non-leaf edges, there can be conflicts between the non-leaf and the leaf edges. But that problem is easy to solve: since the number of colors used by the non-leaf edges at vertex v from the main block of layer i was not increased, there are enough colors in layer i for assigning new colors to the leaf edges. After recoloring the leaf edges, the largest color of each edge remains in the same layer, hence the finish time of each leaf edge can increase by at most a factor of $1+2\epsilon$, and $f_\Psi(e) \leq (1+2\epsilon)(1+(D+1)\epsilon)f_\Phi(e) \leq (1+3D\epsilon)f_\Phi(e)$ follows for every edge e. $\square$

Call a coloring satisfying the requirements of Lemma 5.3 a *standard* coloring. Notice that on a non-leaf edge e only a constant number of different color sets can appear in standard colorings: the main zone is not empty only in a constant number of layers, and in each layer the (at most two) intervals can be placed in a constant number of different ways. More precisely, in a standard coloring edge e can use the main zone only from layer $\lfloor \log_{1+\epsilon}(\frac{\epsilon}{4} \cdot x(e)) \rfloor$ to layer $\lceil \log_{1+\epsilon}(\frac{2}{\epsilon} \cdot x(e)) \rceil$, that is, only in at most

$$\log_{1+\epsilon} \frac{\frac{2}{\epsilon} \cdot x(e)}{\frac{\epsilon}{4} \cdot x(e)} + 2 = \log_{1+\epsilon} 8/\epsilon^2 + 2 = O\left(\frac{1}{\epsilon} \cdot \log \frac{1}{\epsilon}\right)$$

layers. In layer L_i, the intervals are of the form $[1 + j_1\lceil \epsilon^2 Q_i/8 \rceil, j_2\lceil \epsilon^2 Q_i/8 \rceil]$ for some j_1, j_2. This means that the end points of the intervals can take only at most $8/\epsilon^2$ different values, hence there are $(8/\epsilon^2)^2$ different possibilities for each of the two intervals. Therefore, if we denote by $\mathcal{C}_e$ the different color sets that can appear in a standard coloring on non-leaf edge e, then $|\mathcal{C}_e| = ((8/\epsilon^2)^4)^{O((1/\epsilon)\cdot \log 1/\epsilon)} = 2^{O((1/\epsilon)\cdot \log^2 1/\epsilon)}$.

Theorem 5.4 *If every edge of $T(V,E)$ has at most D non-leaf child edges, then for every $\epsilon_0 > 0$, there is a $2^{O(D^2/\epsilon_0 \cdot \log^2(D/\epsilon_0))} \cdot n$ time algorithm that gives a $(1+\epsilon_0)$-approximate solution to the* SEMC *problem.*

PROOF. Set $\epsilon := \epsilon_0/3D$. We use dynamic programming to find the best standard coloring: for every non-leaf edge e, and every set $S \in \mathcal{C}_e$, we determine $\mathrm{OPT}(e,S)$, which is defined to be the sum of the best standard coloring of T_e, with the additional requirement that edge e receives color set S (recall that T_e is the subtree with root edge e). Clearly, if all the values $\{\mathrm{OPT}(r,S) : S \in \mathcal{C}_r\}$ are determined for the root edge r of T, then the minimum of these values is the sum of the best standard coloring, which is by Lemma 5.3 at most $(1+3D\epsilon) = (1+\epsilon_0)$ times the minimum sum.

The values $\mathrm{OPT}(e, S)$ are calculated in a bottom-up traversal of the edges. Assume that e has k non-leaf child edges $e_1, e_2, \ldots, e_k$ and ℓ leaf child edges $e'_1, e'_2, \ldots, e'_\ell$. When $\mathrm{OPT}(e, S)$ is determined, the values $\mathrm{OPT}(e_i, S_i)$ are already available for every $1 \leq i \leq k$ and $S_i \in \mathcal{C}_{e_i}$. In a standard coloring of T_e, every edge e_i is assigned a color set from $\mathcal{C}_{e_i}$. We enumerate all the $\prod_{i=1}^{k} |\mathcal{C}_{e_i}|$ possibilities for these color sets. For each combination $S_1 \in \mathcal{C}_{e_1}$, ..., $S_k \in \mathcal{C}_{e_k}$, we check whether these sets are pairwise disjoint. If so, then we determine the minimum sum that a standard coloring can have with these assignments. The minimum sum of subtree T_{e_i} with color set S_i on e_i is given by $\mathrm{OPT}(e_i, S_i)$. The finish time of edge e can be calculated from S. Now only the leaf edges e'_1, ..., e'_ℓ remain to be colored. It is easy to see that the best thing to do is to sort these leaf edges by increasing demand size, and color them one after the other, using the colors not already assigned to e, e_1, ..., e_k. Therefore, we can calculate the minimum sum corresponding to a choice of color sets $S_1 \in \mathcal{C}_{e_1}$, ..., $S_k \in \mathcal{C}_{e_k}$, and we set $\mathrm{OPT}(e, S)$ to the minimum over all the combinations.

The algorithm solves at most $\sum_{e \in E} |\mathcal{C}_e| = n \cdot 2^{O((1/\epsilon) \cdot \log^2 1/\epsilon)}$ subproblems. To solve a subproblem, at most $2^{O(D \cdot (1/\epsilon) \cdot \log^2 1/\epsilon)}$ different combinations of the sets $S_1, \ldots, S_k$ have to be considered. Each color set can be described by $O(\frac{1}{\epsilon} \cdot \log \frac{1}{\epsilon})$ intervals, and the time required to handle each combination is polynomial in D and the number of intervals. Therefore, the total running time of the algorithm is $2^{O(D \cdot 1/\epsilon \cdot \log^2(1/\epsilon))} \cdot n = 2^{O(D^2/\epsilon_0 \cdot \log^2(D/\epsilon_0))} \cdot n$. $\square$

6 The general case

In this section, we prove that SEMC admits a PTAS for arbitrary trees. The edges of the tree are partitioned into subtrees in such a way that each subtree is an almost bounded degree tree (recall that in an almost bounded degree tree each node has a bounded number of non-leaf child edges). Now the algorithm presented in Section 5.2 can be used to obtain a good approximate coloring for each subtree. These colorings can be merged into a coloring of the whole tree, but this coloring will not be necessarily a proper coloring: there might be conflicts between edges that were in different subtrees. However, we show that using a series of transformations, these conflicts can be resolved with only a small increase in the value of the solution.

Theorem 6.1 *For every $\epsilon_0 > 0$, there is a $2^{O(1/\epsilon_0^{11} \cdot \log^2(1/\epsilon_0))} \cdot n$ time algorithm that gives a $(1+\epsilon_0)$-approximate solution to the* SEMC *problem for every tree T and demand function x_0.*

PROOF. Let $\epsilon := \epsilon_0/32$. The algorithm consists of a series of phases. The last phase produces a proper coloring of (T, x_0), and has cost at most $(1 + \epsilon_0)\mathrm{OPT}(T, x_0)$. In the following we describe these phases.

Phase 1: Rounding the demands. Let $x(e)$ be the smallest q_i that is not smaller than $x_0(e)$. Since $q_{i+1} \leq (1+\epsilon)^{i+1} \leq (1+\epsilon)(q_i+1)$, thus $x(e) \leq (1+\epsilon)x_0(e)$. Therefore, by Lemma 4.4, this modification increases the minimum sum by at most a factor of $1+\epsilon$. An edge e with demand q_i will be called a *class* i edge (if $x(e) = q_i$ for more than one i, then take the smallest i). The class of edge e will be denoted by $\mathrm{class}(e)$.

Phase 2: Partitioning the tree. We partition the edges of the tree into connected components such that in every subtree the number of non-leaf child edges of a node is bounded by a constant. To obtain this partition, the edges of the tree are divided into *large edges*, *small edges*, and *frequent edges*. It will be done in such a way that every node has at most $D := 6/\epsilon^5$ large child edges. If a node has fewer than D children, then its child edges are large edges. Let v be a node with at least D children, and denote by $n(v, i)$ the number of class i child edges of v. Let $N(v)$ be the largest i such that $n(v, i) > 0$ and set $F := 6/\epsilon^3$. Let e be a class i child edge of v. If $n(v, i) > F$, then e is a *frequent edge*. If $n(v, i) \leq F$ and $i \leq N(v) - 1/\epsilon^2$, then e is a *small edge*. Otherwise, if $n(v, i) \leq F$ and $i > N(v) - 1/\epsilon^2$, then e is a *large edge*. Clearly, v can have at most $F \cdot 1/\epsilon^2 = 6/\epsilon^5 = D$ large child edges: for each class $N(v)$, $N(v) - 1$, $\ldots$, $N(v) - 1/\epsilon^2 + 1$, there are at most F such edges.

The tree is split at the lower node of every small and frequent edge, the connected components of the resulting forest form the classes of the partition. Another way to describe this partition: delete every small and frequent edge, make the connected components of the remaining graph the classes of the partition, and put every deleted edge into the class where its upper node belongs. Clearly, every small and frequent edge becomes a leaf edge in its subtree, thus if every node has at most D large child edges in the tree, then in every subtree each node has at most D non-leaf child edges.

Color each subtree with the algorithm of Theorem 5.4. This step can be done in $2^{O(D^2/\epsilon \cdot \log^2(D/\epsilon))} \cdot n = 2^{O(36/\epsilon^{10} \cdot 1/\epsilon \cdot \log^2(6/\epsilon^6))} \cdot n = 2^{O(1/\epsilon^{11} \cdot \log^2(1/\epsilon))} \cdot n$ time. Each coloring is a $(1+\epsilon)$-approximate coloring of the given subtree, thus merging these colorings yields a (not necessarily proper) coloring Ψ_1 of T such that $f_{\Psi_1}(T) \leq (1+\epsilon)\mathrm{OPT}(T, x)$. In the rest of the proof, we transform Ψ_1 into a proper coloring in such a way that the sum of the coloring does not increase too much.

Phase 3: Small edges. Since the tree is a bipartite graph, we can assign a *parity* to each node, such that each parity is either 1 or 2, and neighboring nodes have different parities. Let the parity of an edge be the parity of its upper node. Observe that if two edges have the same parity and they have a common node v, then v is the upper node of both edges.

Consider the (ϵ, ℓ)-augmentation of the coloring Ψ_1 with $\ell := 6$. This results in a coloring Ψ_2 such that $f_{\Psi_2}(G) \leq (1+\epsilon\ell) f_{\Psi_1}(G)$ (see Section 5.1). First we modify Ψ_2 in such a way that the small edges use only the extra zones Z_1 and Z_2. More precisely, if a small edge e has parity $r \in \{1, 2\}$, then e is recolored using the colors in Z_r. Since the extra zones contain only a very small fraction of the color spectrum, the recoloring can significantly increase the finish time of the small edges, but not more than by a factor of $\frac{2}{\epsilon}$ (Lemma 5.1b). However, we show that the total demand of the small edges at v is so small compared to the largest demand on the child edges of v, that their total finish time will be negligible, even after this large increase. By definition, the largest child edge of v has demand $q_{N(v)}$.

Let S_v be the set of those small edges whose upper node is v. Let r be the parity of v. Color the edges in S_v one after the other, in the order of increasing demand size, using only the colors in Z_r. Call the resulting coloring Ψ_3. We claim that $f_{\Psi_3}(S_v) \leq \epsilon q_{N(v)}$ for every node v, thus transforming Ψ_2 into Ψ_3 increases the total sum by at most $\sum_{v \in T} f_{\Psi_3}(S_v) \leq \epsilon \sum_{v \in T} q_{N(v)} \leq \epsilon f_{\Psi_2}(T)$ and $f_{\Psi_3}(T) \leq (1+\epsilon) f_{\Psi_2}(T)$ follows. To give an upper bound on $f_{\Psi_3}(S_v)$, we assume the worst case, that is, $n(v,i) = F$ for every $i \leq N(v) - 1/\epsilon^2$. Imagine first that the small edges are colored using the full color spectrum, not only with the colors of zone Z_r. Assume that the small edges are colored in the order of increasing demand size, and consider a class k edge e. In the coloring, only edges of classes not greater than k are colored before e. Hence the finish time of e is at most

$$\begin{aligned}\sum_{i=0}^{k} n(v,i) q_i &\leq F \sum_{i=0}^{k} (1+\epsilon)^i \leq \frac{6(1+\epsilon)}{\epsilon^4} \cdot (1+\epsilon)^k \\ &\leq \frac{14}{\epsilon^4} \cdot \frac{1}{2}(1+\epsilon)^k \leq \frac{14}{\epsilon^4} \cdot \lfloor (1+\epsilon)^k \rfloor = \frac{14}{\epsilon^4} \cdot q_k.\end{aligned}$$

That is, the finish time of an edge is at most $\frac{14}{\epsilon^4}$ times its demand (in the second inequality, we used $\sum_{i=0}^{k} (1+\epsilon)^i = ((1+\epsilon)^{k+1} - 1)/\epsilon < (1+\epsilon)^{k+1}/\epsilon$). Therefore, the total finish time of the small edges is at most $\frac{14}{\epsilon^4}$ times the total

demand, which is

$$\frac{14}{\epsilon^4}\sum_{i=0}^{N(v)-1/\epsilon^2} n(v,i)q_i \le \frac{84}{\epsilon^7}\sum_{i=0}^{N(v)-1/\epsilon^2}(1+\epsilon)^i$$
$$\le \frac{85}{\epsilon^8}(1+\epsilon)^{N(v)-1/\epsilon^2} \le \frac{85}{\epsilon^8}\cdot 2^{-1/\epsilon}\cdot(1+\epsilon)^{N(v)}$$
$$\le \frac{\epsilon^2}{2}\cdot\frac{1}{2}(1+\epsilon)^{N(v)} \le \frac{\epsilon^2}{2}\cdot\lfloor(1+\epsilon)^{N(v)}\rfloor = \frac{\epsilon^2}{2}\cdot q_{N(v)}.$$

(In the third inequality we use $(1+\epsilon)^{1/\epsilon} \ge 2$, in the fourth inequality it is assumed that ϵ is sufficiently small that $2^{1/\epsilon} \ge 4\cdot 85/\epsilon^{10}$ holds.) However, the small edges do not use the full color spectrum, only the colors in zone Z_r. By Lemma 5.1b, zone Z_r contains at least c colors up to $\frac{2c}{\epsilon}$, thus every finish time in the calculation above should be multiplied by at most $\frac{2}{\epsilon}$. Therefore, the sum of the small edges is at most

$$f_{\Psi_3}(S_v) \le \frac{2}{\epsilon}\cdot\frac{\epsilon^2}{2}\cdot q_{N(v)} \le \epsilon q_{N(v)},$$

as claimed.

Phase 4: Shifting the frequent edges. Now we have a coloring Ψ_3 that is still not a proper coloring, but conflicts appear only between some frequent edges and their child edges. In Phases 4 and 5 we ensure that every frequent edge e uses only colors greater than $\frac{2}{\epsilon}\cdot x(e)$ from the main zone. In Phase 6, the conflicts are resolved using a set of so far unused colors, the colors in extra zones Z_5 and Z_6.

Let F_v be the set of frequent child edges of v, and let $\Lambda_v = \bigcup_{e\in F_v}\Psi_3(e)$ be the colors used by the frequent child edges of node v. We recolor the edges in F_v using only the colors in Λ_v and some colors from zones Z_3 and Z_4. Let $e_1, e_2, \ldots, e_{|F_v|}$ be an ordering of the edges in F_v by increasing demand size. Recall that the algorithm in Theorem 5.4 assigned the colors to the leaf edges in increasing order of demand size, thus it can be assumed that frequent edge e_1 uses the first $x(e_1)$ colors in Λ_v, edge e_2 uses the $x(e_2)$ colors after that, etc. Denote by $t(c) = |\{e \in F_v : f_{\Psi_3}(e) \ge c\}|$ the number of edges whose finish time is at least c, and denote by $t(c,i) = |\{e \in F_v : f_{\Psi_3}(e) \ge c,\ \text{class}(e) = i\}|$ the number of class i edges among them. Clearly, $t(c) = \sum_{i=0}^{\infty} t(c,i)$ holds. Moreover, it can be easily verified that the total finish time of the edges in F_v can be expressed as $f_{\Psi_3}(F_v) = \sum_{c=1}^{\infty} t(c)$.

The first step is to produce a coloring Ψ_4 where every frequent edge e has only $(1-\frac{2\epsilon}{5})x(e)$ colors, but these colors are all greater than $\frac{2}{\epsilon}\cdot x(e)$. The demand function is split into two parts: $x(e) = x_1(e)+x_2(e)$, where $x_1(e)$ is $(1-\frac{2\epsilon}{5})x(e)$

and $x_2(e)$ is $\frac{2\epsilon}{5} \cdot x(e)$, but rounding has to be done carefully. What we want to achieve is that

$$\sum_{j=1}^{k} x_2(e_j) \leq \frac{2\epsilon}{5} \sum_{j=1}^{k} x(e_j) \tag{8}$$

holds for every $1 \leq k \leq |F_v|$, and the total demand of the class i edges in x_1 is at most

$$\sum_{e \in F_v,\ \mathrm{class}(e)=i} x_1(e) \leq \left\lceil n(e,i) \left(1 - \frac{2\epsilon}{5}\right) q_i \right\rceil . \tag{9}$$

It can be easily verified that these two requirements hold if x_1 is defined as $x_1(e) = \lceil (1 - \frac{2\epsilon}{5}) q_i \rceil$ for the first m edges of class i, and $x_1(e) = \lfloor (1 - \frac{2\epsilon}{5}) q_i \rfloor$ for the rest of the class i edges, where

$$m = \left\lceil n(v,i) \left(1 - \frac{2\epsilon}{5}\right) q_i \right\rceil - n(v,i) \left\lfloor \left(1 - \frac{2\epsilon}{5}\right) q_i \right\rfloor .$$

This phase of the algorithm produces a coloring Ψ_4 of F_v that assigns only $x_1(e)$ colors to every edge $e \in F_v$, but satisfies the condition that it uses only the colors in Λ_v, and every edge e receives only colors greater than $\frac{2}{\epsilon} \cdot x(e)$. In the next phase we will extend this coloring using the colors in zones Z_3 and Z_4: every edge e will receive $x_2(e)$ additional colors.

Coloring Ψ_4 is defined as follows. Consider the edges $e_1, \ldots, e_{|F_v|}$ in this order, and assign to e_k the first $x_1(e_k)$ colors in Λ_v greater than $\frac{2}{\epsilon} \cdot x(e_k)$ and not already assigned to an edge e_j $(j < k)$. Notice the following property of Ψ_4: if $j < k$, then every color in $\Psi_4(e_j)$ is less than every color in $\Psi_4(e_k)$. This follows from $\frac{2}{\epsilon} \cdot x(e_j) \leq \frac{2}{\epsilon} \cdot x(e_k)$: every color usable for e_k is also usable for e_j if $j < k$. Define $t'(c) = |\{e \in F_v : f_{\Psi_4}(e) \geq c\}|$ and $t'(c,i) = |\{e \in F_v : f_{\Psi_4}(e) \geq c,\ \mathrm{class}(e) = i\}|$ as before, but now using the coloring Ψ_4. We claim that $t'(c,i) \leq (1+\epsilon) t(c,i)$ holds for every $c \geq 1, i \geq 0$. If this is true, then $t'(c) \leq (1+\epsilon) t(c)$ holds and $f_{\Psi_4}(F_v) \leq (1+\epsilon) f_{\Psi_3}(F_v)$ follows from $f_{\Psi_4}(F_v) = \sum_{c=1}^{\infty} t'(c)$. Summing this for every node v gives $f_{\Psi_4}(T) \leq (1+\epsilon) f_{\Psi_3}(T)$.

First we show that $t'(c,i) \leq t(c,i) + 2/\epsilon$. If every class i edge has finish time at least c in Ψ_3, then $t(c,i) = n(c,i) \geq t'(c,i)$ and we are done. Therefore, there is at least one class i edge that has finish time less than c in Ψ_3. This implies that the frequent edges of class 0, 1, ..., $i-1$ use only colors less than c. Denote by X the total demand of these edges (in the demand function $x(e)$), and denote by Y the number of colors used by the class i edges below c in Ψ_3.

Now recall the way Ψ_4 was defined, and consider the step when every edge with class less than i is already colored. At this point at most X colors of Λ_v are used below c (possibly fewer, since Ψ_4 assigns only $x_1(e)$ colors to every edge e, and only colors greater than $\frac{2}{\epsilon} \cdot x(e)$). Therefore, at least Y colors are still unused in Λ_v below c. From these colors at least $Y - \frac{2}{\epsilon} \cdot q_i$ of them are above $\frac{2}{\epsilon} \cdot q_i$. Thus Ψ_4 can color at least $(Y - \frac{2}{\epsilon} \cdot q_i)/q_i = Y/q_i - 2/\epsilon$ edges

of class i using only colors below c. However, Ψ_3 uses Y colors below c for the class i edges, hence it can color at most Y/q_i such edges below c, and $t'(c,i) \leq t(c,i) + 2/\epsilon$ follows.

We consider two cases. If $t(c,i) \geq 2/\epsilon^2$, then $t'(c,i) \leq t(c,i) + 2/\epsilon \leq (1+\epsilon)t(c,i)$, and we are done. Let us assume therefore that $t(c,i) \leq 2/\epsilon^2$, it will turn out that in this case $t'(c,i) = 0$. There are $n(v,i) - t(c,i) \geq n(v,i) - 2/\epsilon^2$ class i edges that has finish time less than c in Ψ_3. Therefore, as in the previous paragraph, before Ψ_4 starts coloring the class i edges, there are at least $(n(v,i) - 2/\epsilon^2) \cdot q_i$ unused colors less than c in Λ_v. By (9), the total demand of the class i edges in demand function $x_1(e)$ is at most $\lceil n(e,i)(1-\frac{2\epsilon}{5})q_i \rceil$. The following calculation shows that the unused colors below c in Λ_v is sufficient to satisfy all these edges, thus Ψ_4 assigns to these edges only colors less than c. We have to skip the colors not greater than $\frac{2}{\epsilon} \cdot q_i$, these colors cannot be assigned to the edges of class i, which means that the number of usable colors is at least

$$\begin{aligned}(n(v,i) - 2/\epsilon^2) \cdot q_i - 2q_i/\epsilon &\geq \left(n(v,i) - \frac{12\epsilon^2}{5}\right) \cdot q_i + 1 \\ &\geq \left(1 - \frac{2\epsilon}{5}\right) n(v,i) q_i + 1 \geq \left\lceil n(e,i) \left(1 - \frac{2\epsilon}{5}\right) q_i \right\rceil,\end{aligned}$$

since $n(v,i) \geq 6/\epsilon^3$ by the definition of the frequent edges. Therefore, Ψ_4 assigns to the class i edges only colors less than c, and $t(c,i) = 0$ follows.

Phase 5: Full demand for the frequent edges. The next step is to modify Ψ_4 such that every frequent edge receives $x(e)$ colors, not only $x_1(e)$. Coloring Ψ_5 is obtained from Ψ_4 by assigning to every frequent edge e an $x_2(e)$ additional colors from zones Z_3 or Z_4. More precisely, let v be a node with parity r (as defined in Phase 3), and let $e_1, \ldots, e_{|F_v|}$ be its frequent child edges, ordered in increasing demand size, as before. Assign the first $x_2(e_1)$ colors from Z_{2+r} to e_1, the first $x_2(e_2)$ colors from Z_{2+r} not used by e_1 to e_2, etc. It is clear that no conflict arises with the assignment of these colors.

We claim that these additional colors do not increase the finish time of the frequent edges. Let $x_i^* = \sum_{j=1}^{i} x_1(e_j)$ be the total demand in x_1 of the first i frequent edges at v. The finish time of e_i in Ψ_4 is clearly at least x_i^*, since Ψ_4 colors every edge e_j with $j < i$ before e_i. On the other hand, by Lemma 5.1b, zone Z_{2+r} contains at least $\lfloor \frac{\epsilon}{2} \cdot x_i^* \rfloor$ colors not greater than x_i^*. These colors are sufficient to satisfy the additional demand of the first i edges: by (8) the first i edges need a total of at most $\frac{2\epsilon}{5} \sum_{j=1}^{i} x(e) \leq \frac{\epsilon}{2} \cdot x_i^*$ colors.

Phase 6: Resolving the conflicts. Now we have a coloring Ψ_5 such that there are conflicts only between frequent edges and their child edges. Further-

more, if e is a frequent edge, then $\Psi_5(e)$ contains only colors greater than $\frac{2}{\epsilon} \cdot x(e)$ from the main zone. It is clear from the construction of Ψ_5 that only the colors in the main zone can conflict.

Let e be a frequent edge that conflicts with some of its children. Assume that the child edges of e have parity r (as defined in Phase 3). There are at most $x(e)$ colors that are used by both e and a child of e. We resolve this conflict by recoloring the child edges of e in such a way that they use the first at most $x(e)$ colors in zone Z_{4+r} instead of the colors in $\Psi_5(e)$. It is clear that if this operation is applied for every frequent edge e, then the resulting color Ψ_6 is a proper coloring.

Notice that if a child edge e' of e is recolored, then it has finish time at least $\frac{2}{\epsilon} \cdot x(e)$, otherwise it does not conflict with e. On the other hand, by Lemma 5.1b, zone Z_{4+r} contains at least $x(e)$ colors up to $\frac{2}{\epsilon} \cdot x(e)$, thus the recoloring does not add colors greater than that. Therefore, the finish time of e' is not increased.

Analysis. The sum of the coloring Ψ_6 can be bounded as follows (assuming that ϵ is sufficiently small):

$$\begin{aligned} f_{\Psi_6}(T) &= f_{\Psi_5}(T) = f_{\Psi_4}(T) \leq (1+\epsilon) f_{\Psi_3}(T) \\ &\leq (1+\epsilon)^2 f_{\Psi_2}(T) \leq (1+(\ell+1)\epsilon)(1+\epsilon)^2 f_{\Psi_1}(T) \\ &\leq (1+(\ell+1)\epsilon)(1+\epsilon)^3 \mathrm{OPT}(T,x) \leq (1+(\ell+1)\epsilon)(1+\epsilon)^4 \mathrm{OPT}(T,x_0) \\ &\leq (1+7\epsilon)(1+8\epsilon)\mathrm{OPT}(T,x_0) \leq (1+32\epsilon)\mathrm{OPT}(T,x_0) \\ &= (1+\epsilon_0)\mathrm{OPT}(T,x_0) \end{aligned}$$

Therefore, Ψ_6 is a $(1+\epsilon_0)$-approximate solution to the SEMC instance (T, x_0).

The running time of the algorithm is dominated by the coloring of the low-degree components with the algorithm of Theorem 5.4. This phase requires $2^{O(36/\epsilon^{10} \cdot 1/\epsilon \cdot \log^2(6/\epsilon^6))} \cdot n = 2^{O(1/\epsilon_0^{11} \log^2(1/\epsilon_0))} \cdot n$ time. The other parts of the algorithm can be done in time linear in the size of the input. Therefore, the total running time is $2^{O(1/\epsilon_0^{11} \cdot \log^2(1/\epsilon_0))} \cdot n$, which completes the proof of Theorem 6.1. □

Acknowledgments

I'm grateful to Katalin Friedl and to the anonymous referees for their comments that greatly improved the presentation of the paper.

References

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

[2] E. G. Coffman, Jr., M. R. Garey, D. S. Johnson, and A. S. LaPaugh. Scheduling file transfers. *SIAM J. Comput.*, 14(3):744–780, 1985.

[3] K. Giaro and M. Kubale. Edge-chromatic sum of trees and bounded cyclicity graphs. *Inform. Process. Lett.*, 75(1-2):65–69, 2000.

[4] M. M. Halldórsson and G. Kortsarz. Tools for multicoloring with applications to planar graphs and partial k-trees. *J. Algorithms*, 42(2):334–366, 2002.

[5] M. M. Halldórsson, G. Kortsarz, A. Proskurowski, R. Salman, H. Shachnai, and J. A. Telle. Multicoloring trees. *Inform. and Comput.*, 180(2):113–129, 2003.

[6] 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.

[7] J. A. Hoogeveen, S. L. van de Velde, and B. Veltman. Complexity of scheduling multiprocessor tasks with prespecified processor allocations. *Discrete Appl. Math.*, 55(3):259–272, 1994.

[8] K. Jansen. The optimum cost chromatic partition problem. In *Algorithms and complexity (Rome, 1997)*, volume 1203 of *Lecture Notes in Comput. Sci.*, pages 25–36. Springer, Berlin, 1997.

[9] K. Jansen. Approximation results for the optimum cost chromatic partition problem. *J. Algorithms*, 34(1):54–89, 2000.

[10] M. Kubale. Preemptive versus nonpreemptive scheduling of biprocessor tasks on dedicated processors. *European Journal of Operational Research*, 94(2):242–251, 1996.

[11] D. Marx. The complexity of tree multicolorings. In *Mathematical foundations of computer science 2002*, volume 2420 of *Lecture Notes in Comput. Sci.*, pages 532–542. Springer, Berlin, 2002.

[12] C. H. Papadimitriou. *Computational complexity*. Addison-Wesley Publishing Company, Reading, MA, 1994.

[13] M. R. Salavatipour. On sum coloring of graphs. *Discrete Appl. Math.*, 127(3):477–488, 2003.

[14] A. Schrijver. *Combinatorial optimization. Polyhedra and efficiency.*, volume 24 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, 2003.