Minimum Sum Multicoloring on the Edges of Trees⋆ (Extended Abstract)

Dániel Marx

Department of Computer Science and Information Theory,
Budapest University of Technology and Economics
Budapest Pf. 91, H-1521, Hungary
dmarx@cs.bme.hu

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 vertex 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.

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 $x(e)$ colors to each edge e such that the 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 multicoloring problem the goal is to minimize the sum of finish times.

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

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

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 the mutual diagnostic testing of processors [6]. Note the we allow 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.

Of particular interest is the case where the graph G 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 the same time, and a server cannot accept connections from more than one client simultaneously. 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]. For general demands, [1] gives a 2-approximation algorithm. The problem can be solved in polynomial time if every edge has unit demand and the graph is a tree [3,7].

In this paper, we consider the minimum sum edge coloring problem restricted to trees. We show that, unlike in the unit demand case, minimum sum edge coloring is **NP**-hard on trees if the demands are allowed to be at most 2. The main contribution of the paper is a polynomial time approximation scheme (PTAS) for minimum sum edge multicoloring in trees.

In [4,5] PTAS is given for the vertex coloring version of the problem in the case when the graph is a tree, partial k-tree, or a planar graph. One of their main tools is the decomposition of colors into layers of geometrically increasing size. This method will be used in this paper as well. However, most of the other tools in [4,5] cannot be applied in our case, since those tools assume that the graph can be colored with a constant number of colors. If the maximum degree of the tree can be arbitrary, then the line graph of the tree can contain arbitrarily large cliques. On 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 the sum of a large clique has to be very large in every coloring (it grows quadratically in the size of the clique), more errors can be tolerated in an approximate solution, and this gives us more elbow space in constructing a good approximation.

The paper is organized as follows. In Section 2 we introduce some notations and give results on the complexity of the problem. Section 3 gives a PTAS for trees where the maximum degree of the tree is bounded by a constant, while Section 4 gives a linear time PTAS for general trees.

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 $\Psi(e_1) \cap \Psi(e_2) = \emptyset$ if e_1 and e_2 are adjacent in G.
Goal: The *finish time* of edge e in coloring Ψ is the highest color assigned to it, $f_\Psi(e) = \max\{c \in \Psi(e)\}$. The goal is to minimize $\sum_{e \in E} f_\Psi(e)$, the *sum* of the coloring Ψ.

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 obtained is denoted by $\mathrm{OPT}(G, x)$.

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 *bottom up traversal* of the edges is an ordering of the edges in such a way that every edge appears before its parent edge. It is clear that such ordering exists and can be found in linear time.

Since the tree is bipartite, every node has a *parity*, which is either 1 or 2, and neighboring nodes have different parity. Let the parity of an edge be the parity of its upper node. Thus if two edges have the same parity and they have a common node v, then v is the upper node of both edges.

If the tree has maximum degree Δ, then the edges can be colored with Δ colors. This color will be called the *type* of the edge. In some of the algorithms, the leaf edges are special, they are handled differently, therefore we want to 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 colored with $D + 1$ colors, and they can be given a type from $1, 2, \ldots, D + 1$ such that adjacent edges have different type.

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

Lemma 1. *If T is a tree with maximum degree Δ and maximum demand p, then every optimum coloring of the* **semc** *problem uses at most $p(2\Delta - 1)$ colors.*

If both the maximum degree of the tree and the maximum demand is 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. We omit the details.

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

On the other hand, if only the demand is bounded, then the problem becomes **NP**-complete:

Theorem 3. *Minimum sum edge coloring is* **NP***-complete in trees, even if every demand is 1 or 2.*

Using polyhedral techniques, we can show that trees have the following scaling property (proof is omitted):

Theorem 4. *For every tree T, demand function x and integer q, if $x'(e) = q \cdot x(e)$, then $\mathrm{OPT}(T, x') = q \cdot \mathrm{OPT}(T, x)$ holds.*

In general, Theorem 4 does not hold for every graph, not even for every bipartite graph. We will use Theorem 4 to reduce the number of different demand sizes that appear in the graph, with only a small increase of the sum:

Lemma 5. *Let (T, x) be an instance of* **semc** *and let $x'(e)$ be $\lfloor (1+\epsilon)^i \rfloor$ for the smallest i such that $\lfloor (1+\epsilon)^i \rfloor \geq x(e)$. Then $\mathrm{OPT}(T, x') \leq (1+\epsilon) \cdot \mathrm{OPT}(T, x)$.*

As explained in the introduction, our goal is to minimize the sum of finish times, not to minimize the number of different colors used. Nevertheless, in Theorem 6 we show that the minimum number of colors required for the coloring the edges of a tree can be determined in polynomial time. The approximation algorithm presented in Section 3 uses this result to solve certain subproblems.

Theorem 6. *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 multicoloring Ψ using C colors where each $\Psi(e)$ consists of at most two intervals of colors. Moreover, if each $x(e)$ is an integer multiple of some integer q, then we can find such a 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. The coloring Ψ can be constructed by a simple greedy algorithm. Details omitted. □

3 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 $n^{O(k^2/\epsilon^2)}$ 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 if the degree of the tree is not bounded, but we know that every edge has 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 4 a PTAS is given for general trees, which uses the result in this section as a subroutine.

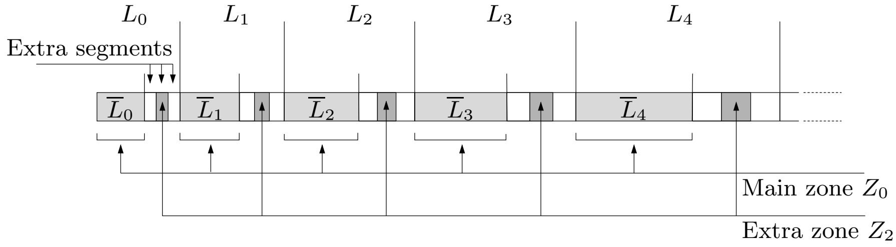

Fig. 1. The decomposition of the colors into layers ($\ell = 3$)

3.1 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 9) and general trees (Theorem 10).

For some $\epsilon > 0$ and integer $\ell \geq 0$, the (ϵ, ℓ)-*decomposition* divides the 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 size: 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 $Q_i^\epsilon = |L_i| = q_{i+1} - q_i$ the size of the ith layer. Notice that $q_{i+1} \leq (1+\epsilon)q_i$. The total size of layers L_0, L_1, $\ldots$, L_i is $q_{i+1} - 1$.

Each layer is divided into a main block and an extra block. The extra block is further divided into extra segments (see Figure 1). Layer L_i is divided into two parts: the first $\frac{1}{1+\epsilon\ell} Q_i^\epsilon$ colors form the *main block* of layer L_i and the remaining $\frac{\epsilon\ell}{1+\epsilon\ell} Q_i^\epsilon$ colors the *extra block*. The main block of layer L_i is denoted by $\overline{L}_i$. The union of the main block of every layer L_i is 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 jth extra segment of every layer L_i forms the jth extra zone Z_j. Each extra zone contains $\frac{\epsilon}{1+\epsilon\ell} Q_i^\epsilon$ colors from layer L_i. Rounding problems are not discussed in this extended abstract, but they can be handled rigorously.

We will need the following properties of the defined zones:

Lemma 7. *For given ϵ and ℓ, the (ϵ, ℓ)-decomposition of the colors has the following properties:*

(a) For every $c \geq 1$, there are at least c colors in Z_0 not greater than $\lfloor (1+\epsilon\ell)c \rfloor$.
(b) For every $c \geq 1$ and $1 \leq j \leq \ell$, there are at least c colors in Z_j not greater than $\lceil (1+\epsilon)(1+\epsilon\ell)/\epsilon \cdot c \rceil$.

Given a multicoloring Ψ, the operation (ϵ, ℓ)-*augmentation* creates a multicoloring Φ in 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 cth color from the main zone Z_0. By Lemma 7a, $f_\Phi(e) \leq \lfloor (1+\epsilon\ell) f_\Psi(e) \rfloor$, 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.

3.2 PTAS for Bounded Degree Trees

The polynomial time algorithm of Theorem 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 is bounded. In general, however, the number of different color sets that can be assigned to an edge is exponentional in the demand. The following lemma shows that we can find a good approximate coloring by considering only a restricted type of color sets.

Lemma 8. *For $\epsilon > 0$, define $b_{i,j} = q_i + j\lceil \epsilon^2 Q_i^\epsilon/4\rceil$. If each vertex of the tree T has at most D non-leaf child edges, then it has an $(1+\epsilon)(1+(D+1)\epsilon)$-approximate coloring Ψ with the following properties:*

1. *In the (ϵ, D)-decomposition of the colors, if e is a non-leaf edge, then $\Psi(e)$ contains colors from the main zone only between $\epsilon x(e)/4$ and $2x(e)/\epsilon$.*
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, and the intervals in layer L_i are of the form $[b_{i,j_1}, b_{i,j_2} - 1]$ for some j_1 and j_2.*

Proof. Let Φ be an optimum solution, and let Ψ be the result of an (ϵ, D)-augmentation on Φ. By Lemma 7a, $f_\Psi(e) \le (1 + (D + 1))f_\Phi(e)$ for every e.

If $f_\Psi(e) > 2x(e)/\epsilon$ for a non-leaf edge e of type j, then modify $\Psi(e)$ to be the the first $x(e)$ colors of zone Z_j. By Lemma 7b, Z_j contains at least $x(e)$ colors not greater than $\lceil(1+\epsilon)(1+(D+1)\epsilon)/\epsilon \cdot x(e)\rceil \le 2(1+(D+1)\epsilon)x(e)/\epsilon$. Therefore the $x(e)$ colors assigned to e are not greater than $2(1 + (D + 1)\epsilon)x(e)/\epsilon$, hence $f_\Psi(e) \le (1 + (D + 1))f_\Phi(e)$.

If $\Psi(e)$ contains colors in the main zone below $\epsilon x(e)/4$, then delete these colors and let $\Psi(e)$ contain instead the first $\epsilon x(e)/4$ colors from zone Z_j. There are at least $\epsilon x(e)/4$ colors in Z_i below $\lceil(1 + \epsilon)(1 + (D + 1)\epsilon)/\epsilon \cdot \epsilon x(e)/4\rceil \le (1 + (D + 1))x(e) \le (1 + (D + 1))f_\Phi(e)$. Therefore $f_\Psi(e) \le (1 + (D + 1))f_\Phi(e)$ for each edge e, and Ψ is an $(1 + (D + 1)\epsilon)$-approximate solution satisfying the first three properties of the lemma.

Finally, we make Ψ satisfy the fourth requirement as well. For each non-leaf edge e, let $x_i(e)$ be $|\Psi(e) \cap \overline{L}_i|$ rounded down to the next integer multiple of $\lceil \epsilon^2 Q_i^\epsilon/4\rceil$. If we use x_i as a demand function on the non-leaf edges of the tree, then there is multicoloring satisfying x_i that uses at most $|\overline{L}_i|$ colors: $\Psi(e) \cap \overline{L}_i$ is such a coloring. Therefore by Theorem 6, there is a multicoloring Ψ_i that uses at most $|\overline{L}_i|$ colors, satisfies x_i, and where each $\Psi(e)$ consists of at most two intervals of the form $[1 + j_1\lceil \epsilon^2 Q_i^\epsilon/4\rceil, j_2\lceil \epsilon^2 Q_i^\epsilon/4\rceil]$ for some j_1, j_2. Modify coloring Ψ: let Ψ_i determine how the colors are assigned in the main zone of layer i. Now the third requirement is satisfied, but it is possible that Ψ assigns less than $x(e)$ colors to an edge. We can lose at most $\lceil \epsilon^2 Q_i^\epsilon/4\rceil - 1 \le \epsilon^2 Q_i^\epsilon/4$ colors in layer i, hence we lose at most the $\epsilon^2/4$ part of each layer. Since $\Psi(e)$ contains colors only up to $2x(e)/\epsilon$, thus we lose only at most $\epsilon^2/4 \cdot 2x(e)/\epsilon = \epsilon x(e)/2$ colors. If non-leaf edge e is of type j, then we use extra zone Z_j to replace the lost colors. So far, edge e uses at most $\epsilon x(e)/4$ colors from Z_j (previous paragraph), hence there are still place for more than $3\epsilon x(e)/4$ colors in Z_j 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, hence the finish time can increase by at most a factor of $1+\epsilon$. 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 layer i does not increase, there are enough colors in layer i for the leaf edges. The largest color of each leaf edge will be in the same layer, hence its finish time increases by at most a factor of $1+\epsilon$, and $f_\Psi(e) \leq (1+\epsilon)(1+(D+1)\epsilon)f_\Phi(e)$ follows for every edge e. □

Call a coloring satisfying the requirements of Lemma 8 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 $\log_{1+\epsilon} \epsilon x(e)/4$ to layer $\log_{1+\epsilon} 2x(e)/\epsilon$, that is, only in $\log_{1+\epsilon}((2x(e)/\epsilon)/(\epsilon x(e)/4)) = \log_{1+\epsilon} 8/\epsilon^2 = O(1/\epsilon \cdot \log 1/\epsilon)$ layers. In each layer, the end points of the intervals can take only at most $4/\epsilon^2$ different values, hence there are $(4/\epsilon^2)^2$ 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 $|\mathcal{C}_e| = ((4/\epsilon^2)^4)^{O((1/\epsilon)\cdot\log 1/\epsilon)} = 2^{O((1/\epsilon)\cdot\log^2 1/\epsilon)}$.

Theorem 9. *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 an $(1+\epsilon_0)$-approximate solution to the* **semc** *problem.*

Proof. Set $\epsilon := \epsilon_0/(2D+3)$. 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. 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 8 at most $(1+\epsilon)(1+(D+1)\epsilon) \leq (1+\epsilon_0)$ 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 $S_1 \in \mathcal{C}_{e_1}, \ldots, S_k \in \mathcal{C}_{e_k}$, 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 $e'_1, \ldots, e'_\ell$ remain to be colored. It is easy to see that the best thing to do is to order these leaf edges by increasing demand size, and color them one after the other, using the colors not already assigned to $e, e_1, \ldots, e_k$. Therefore we can calculate the minimum sum corresponding to a choice of color sets $S_1 \in \mathcal{C}_{e_1}, \ldots, S_k \in \mathcal{C}_{e_k}$, 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 $O(1/\epsilon \cdot \log 1/\epsilon)$ 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 $2^{O(D\cdot 1/\epsilon\cdot\log^2(1/\epsilon))} \cdot n = 2^{O(D^2/\epsilon_0\cdot\log^2(D/\epsilon_0))} \cdot n$. □

4 The general case

In this section, we prove that **semc** admits a PTAS for arbitrary trees:

Theorem 10. *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 an ϵ_0-approximate solution to the* **semc** *problem for every tree T and demand function x.*

Proof. Let $\epsilon := \epsilon_0/72$. The algorithm consists of a series of phases. The last phase produces a proper coloring of (T, x), and has cost at most $(1+\epsilon_0)\mathrm{OPT}(T, x)$. In the following we describe these phases.

Phase 1: Rounding the Demands. Using Lemma 5, we can assume that $x(e)$ is q_i for some i, modifying $x(e)$ this way 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).

Phase 2: Partitioning the Tree. The edges of the tree are partitioned into subtrees in such a way that in a subtree the number of non-leaf child edges of a node is bounded by a constant. Now Theorem 9 can be used to find an 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, since there might be conflicts between edges that were in different subtrees. However, using a series of transformations, these conflicts will be resolved with only a small increase of the sum.

To obtain this partition, the edges of the tree are divided into *large edges* and *split edges*. It will be done in such a way that every node has at most $D := 4/\epsilon^5$ large child edges. If a node has less 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

$F := 4/\epsilon^3$. Let e be a class i child edge of v. If $n(v,i) > F$, then e is a split edge, and it will be called a *frequent edge.* If $n(v,i) \leq F$ and $i \leq N(v) - \lfloor 1/\epsilon^2 \rfloor$, then e is a split edge, and it will be a called a *small edge.* Otherwise, if $n(v,i) \leq F$ and $i > N(v) - \lfloor 1/\epsilon^2 \rfloor$, then e is a large edge. Clearly, v can have at most $F \cdot \lfloor 1/\epsilon^2 \rfloor = 4/\epsilon^3 \cdot \lfloor 1/\epsilon^2 \rfloor \leq 4/\epsilon^5 = D$ large child edges: for each class $N(v)$, $N(v)-1, \ldots, N(v) - \lfloor 1/\epsilon^2 \rfloor + 1$, there are at most F such edges.

The tree is split at the lower node of every split edge, the connected components of the resulting forest form the classes of the partition. Defined in another way: delete every split edge, make the connected components of the remaining graph the classes of the partition, and put every split edge into the class where its upper node belongs. Clearly, every split 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 every node has at most D non-leaf child edges.

Now assume that each subtree is colored with the algorithm of Theorem 9, this step can be done in $2^{O(D^2/\epsilon \cdot \log^2(D/\epsilon))} \cdot n = 2^{O(16/\epsilon^{10} \cdot 1/\epsilon \cdot \log^2(4/\epsilon^6))} \cdot n = 2^{O(1/\epsilon^{11} \cdot \log^2(1/\epsilon))} \cdot n$ time. Each coloring is an $(1+\epsilon)$-approximate coloring of the given subtree, thus merging these colorings yields a (not necessarily proper) coloring Ψ_1 of T such that $f_{\Psi_1}(T) \leq (1+\epsilon) OPT(T,x)$. 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. Consider the (ϵ, ℓ)-augmentation of the coloring Ψ_1 with $\ell := 6$. This results in a coloring Ψ_2 such that $f_{\Psi_2}(G) \leq f_{\Psi_1}(G)(1+\epsilon\ell)$ (see Section 3). 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 (recall that the parity of the edge is the parity of its upper node). 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, roughly by a factor of $1/\epsilon$. However, we show that the total demand of the small edges are 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 $q_{N(v)}$.

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

$$\sum_{i=0}^{k} n(v,i)q_i \le F\sum_{i=0}^{k}(1+\epsilon)^i \le 4(1+\epsilon)/\epsilon^4 \cdot (1+\epsilon)^k$$
$$\le 5/\epsilon^4 \cdot \lfloor (1+\epsilon)^k \rfloor = 5/\epsilon^4 \cdot q_k.$$

That is, the finish time of an edge is at most $5/\epsilon^4$ times its demand. Therefore the total finish time of the small edges is at most $5/\epsilon^4$ times the total demand, which is

$$\frac{5}{\epsilon^4}\sum_{i=0}^{N(v)-\lfloor 1/\epsilon^2 \rfloor} n(v,i)q_i \le \frac{20}{\epsilon^7}\sum_{i=0}^{N(v)-\lfloor 1/\epsilon^2 \rfloor}(1+\epsilon)^i$$
$$\le \frac{21}{\epsilon^8}(1+\epsilon)^{N(v)-\lfloor 1/\epsilon^2 \rfloor} \le \frac{22}{\epsilon^8}(1+\epsilon)^{N(v)-1/\epsilon^2}$$
$$\le \frac{22}{\epsilon^8}\cdot 2^{-1/\epsilon}\cdot(1+\epsilon)^{N(v)} \le \frac{\epsilon^2}{2}\cdot\frac{1}{2}(1+\epsilon)^{N(v)}$$
$$\le \frac{\epsilon^2}{2}\cdot q_{N(v)}.$$

(In the fourth inequality we use $(1+\epsilon)^{1/\epsilon} \ge 2$, in the fifth inequality it is assumed that ϵ is sufficiently small that $2^{1/\epsilon} \ge 44/\epsilon^{10}$ holds.) However, the small edges do not use the full color spectrum, only the colors in zone Z_r. By Lemma 7b, zone Z_r contains at least c colors up to $\lceil (1+\epsilon\ell)(1+\epsilon)/\epsilon \cdot c \rceil \le 2/\epsilon \cdot c$, thus every finish time in the calculation above should be multiplied by at most $2/\epsilon$. Therefore the sum of the small edges is

$$f_{\Psi_2}(R_v) \le 2/\epsilon \cdot \frac{\epsilon^2}{2}\cdot q_{N(v)} \le \epsilon q_{N(v)},$$

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. First we ensure that every frequent edge e uses only colors greater than $2x(e)/\epsilon$. After that, the conflicts are resolved using a set of so far unused colors, the colors in extra zones Z_5 and Z_6.

Let M_v be the set of frequent child edges of v, and let $\Lambda_v = \bigcup_{e \in M_v} \Psi_3(v)$ be the colors used by the frequent child edges of node v. We recolor the edges in M_v using only the colors in Λ_v. Let $e_1, e_2, \ldots, e_{|M_v|}$ be an ordering of the edges in M_v by increasing demand size. Recall that the algorithm in Theorem 9 assigned the colors to the split edges (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 $t(c) = |\{e \in M_v | f_{\Psi_3}(e) \ge c\}|$ the number of edges whose finish time is at least c, and denote by $t(c,i) = |\{e \in M_v | f_{\Psi_3}(e) \ge c,\ x(e) = q_i\}|$ 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 M_v can be expressed as $f_{\Psi_3}(M_v) = \sum_{c=1}^{\infty} t(c)$.

The first step is to produce a coloring Ψ_4 where every frequent edge e has only $x(e)/(1+\epsilon)$ colors, but these colors are all greater than $2x(e)/\epsilon$. The demand function is split into two parts: $x(e) = x_1(e) + x_2(e)$, where $x_1(e)$ is $x(e)/(1+\epsilon)$ and $x_2(e)$ is $\epsilon x(e)/(1+\epsilon)$, rounding problems are ignored in this extended abstract.

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

Coloring Ψ_4 is defined as follows. Consider the edges $e_1, \dots, e_{|M_v|}$ in this order, and assign to e_k the first $x_1(e_k)$ colors in Λ_v greater than $2x(e_k)/\epsilon$ 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 $2x(e_j)/\epsilon \le 2x(e_k)/\epsilon$: every color usable for e_k is also usable for e_j if $j < k$. Define $t'(c) = |\{e \in M_v | f_{\Psi_4}(e) \ge c\}|$ and $t'(c,i) = |\{e \in M_v | f_{\Psi_4}(e) \ge c,\ x(e) = q_i\}|$ as before, but now using the coloring Ψ_4. We claim that $t'(c,i) \le (1+\epsilon)t(c,i)$ holds for every $c \ge 1, i \ge 0$. If this is true, then $t'(c) \le (1+\epsilon)t(c)$ and $f_{\Psi_4}(M_v) \le (1+\epsilon) f_{\Psi_3}(M_v)$ follow from $f_{\Psi_4}(M_v) = \sum_{c=1}^{\infty} t'(c)$. Summing this for every node v gives $f_{\Psi_4}(T) \le (1+\epsilon) f_{\Psi_3}(T)$.

First we show that $t'(c,i) \le t(c,i) + 3/\epsilon$. Denote by $\lambda_c = |\Lambda_v \cap [1, c-1]|$ the number of colors in Λ_v available below c. If every class i edge has finish time at least c in Ψ_3, then $t(c,i) = n(c,i) \ge t'(c,i)$ and we are ready. Therefore there is at least one class i edge that has finish time less than c. This implies that the frequent edges of class $0, 1, \dots, i-1$ use only colors less than c. Denote by X the total demand of these edges (in the demand function $x(e)$), the class i frequent edges use at most $\lambda_c - X$ colors below c.

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_c are used (possibly less, since Ψ_4 assigns only $x_1(e)$ colors to every edge e, and only colors above $2x(e)/\epsilon$). Therefore at least $\lambda_c - X$ colors are still unused in Λ_v below c. From these colors at least $\lambda_c - X - \lceil 2q_i/\epsilon \rceil$ of them are above $2q_i/\epsilon$. Thus Ψ_4 can color at least $(\lambda_c - X - \lceil 2q_i/\epsilon \rceil)/q_i \ge (\lambda_c - X)/q_i - 3/\epsilon$ edges of class i using only colors below c. However, Ψ_3 uses $\lambda_c - X$ colors below c for the class i edges, hence it can color at most $(\lambda_c - X)/q_i$ such edges below c, and $t'(c,i) \le t(c,i) + 3/\epsilon$ follows.

We consider two cases. If $t(c,i) \ge 3/\epsilon^2$, then $t'(c,i) \le t(c,i) + 3/\epsilon \le (1+\epsilon)t(c,i)$, and we are ready. Let us assume therefore that $t(c,i) \le 3/\epsilon^2$, it will turn out that in this case $t'(c,i) = 0$. There are $n(v,i) - t(c,i) \ge n(v,i) - 3/\epsilon^2$ 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 $(n(v,i) - 3/\epsilon^2) \cdot q_i$ unused colors less than c in Λ_v. The total demand of the class i edges in $x_1(e)$ is at most $n(e,i)q_i/(1+\epsilon)$. 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 $2q_i/\epsilon$, these colors cannot be assigned to the edges of class i, which means that the number of usable colors is at least

$$(n(v,i) - 3/\epsilon^2) \cdot q_i - 2q_i/\epsilon \geq (n(v,i) - \frac{1}{1+\epsilon} \cdot 4/\epsilon^2) \cdot q_i + 1$$
$$\geq (1 - \frac{\epsilon}{1+\epsilon})n(v,i)q_i + 1 \geq n(e,i)q_i/(1+\epsilon),$$

since $n(v,i) \geq 4/\epsilon^3$ by the definition of the frequent edges. Therefore Ψ_4 assigns to the class i edges only colors less than c, hence $t(c,i) = 0$, as claimed.

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 additional $x_2(e)$ colors from zone Z_3 or Z_4. More precisely, let v be a node with parity r, and let $e_1, \ldots, e_{|M_v|}$ be its frequent child edges, ordered in increasing demand size, as before. Assign to e_1 the first $x_2(e_1)$ colors from Z_{2+r}, to e_2 the first $x_2(e_2)$ colors from Z_{2+r} not used by e_1, etc. It is clear that no conflict arises with the assignment of these colors.

We claim that these additional colors increase the total finish time of the frequent edges at v by at most a factor of $(1 + (\ell + 1)\epsilon)$. Let $x_i^* = \sum_{j=1}^{i} x(e_j)$ be the total demand of the first i edges. 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 7b, zone Z_{2+r} contains at least $\frac{\epsilon}{1+\epsilon}x_i^*$ colors not greater than $\lceil (1 + \epsilon\ell)x_i^* \rceil$. These colors are sufficient to satisfy the additional demand of the first i edges.

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. Furthermore, if e is a frequent edge, then $\Psi_5(e)$ contains only colors greater than $2x(e)/\epsilon$ 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. Let the child edges of e have parity r. 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 $\lceil 2x(e)/\epsilon \rceil$, otherwise it does not conflict with e. On the other hand, by Lemma 7b, zone Z_{4+r} contains at least $x(e)$ colors up to $\lceil (1+\epsilon\ell) \cdot (1+\epsilon)x(e)/\epsilon \rceil \leq \lceil 2x(e)/\epsilon \rceil$, thus the recoloring does not add colors above that. Therefore the finish time of e' is not increased, since $f_{\Psi_5}(e') \geq \lceil 2x(e)/\epsilon \rceil$.

Analysis. If we follow how the sum of the coloring changes during the previous steps, it turns out that Ψ_6 is an ϵ_0-approximate solution to the 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 9. This phase requires $2^{O(16/\epsilon^{10}\cdot 1/\epsilon \cdot \log^2(4/\epsilon^6))} \cdot n) = 2^{O(1/\epsilon_0^{11} \log^2(1/\epsilon_0))} \cdot n$ 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 $2^{O(1/\epsilon_0^{11} \log^2(1/\epsilon_0))} \cdot n$, which completes the proof of Theorem 10. □

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