On two segmentation problems

Noga Alon * Benny Sudakov †

Abstract

The hypercube segmentation problem is the following: Given a set S of m vertices of the discrete d-dimensional cube $\{0,1\}^d$, find k vertices $P_1, \ldots, P_k, P_i \in \{0,1\}^d$ and a partitions of S into k segments $S_1, \ldots, S_k$ so as to maximize the sum

$$\sum_{i=1}^{k} \sum_{c \in S_i} P_i \odot c,$$

where $\odot$ is the overlap operator between two vertices of the d-dimensional cube, defined to be the number of positions they have in common.

This problem (among other ones) is considered by Kleinberg, Papadimitriou and Raghavan in [9], where the authors design a deterministic approximation algorithm that runs in polynomial time for every fixed k, and produces a solution whose value is within 0.828 of the optimum, as well as a randomized algorithm that runs in linear time for every fixed k, and produces a solution whose expected value is within 0.7 of the optimum.

Here we design an improved approximation algorithm; for every fixed $\epsilon > 0$ and every fixed k, our algorithm produces in linear time a solution whose value is within $(1 - \epsilon)$ of the optimum. Therefore, for every fixed k, this is a polynomial approximation scheme for the problem. The algorithm is deterministic, but it is convenient to first describe it as a randomized algorithm and then to derandomize it using some properties of expander graphs.

We also consider a segmented version of the minimum spanning tree problem, where we show that no approximation can be achieved in polynomial time, unless P=NP.

1 Introduction

Motivated by the study of various decision-making procedures arising in data mining, Kleinberg, Papadimitriou and Raghavan introduced in [9] a new class of optimization problems, which they

*Department of Mathematics, Raymond and Beverly Sackler Faculty of Exact Sciences, Tel Aviv University, Tel Aviv, Israel and Institute for Advanced Study, Princeton, NJ 08540. Email: noga@math.tau.ac.il. Research supported in part by a USA Israeli BSF grant, by the Minerva Center for Geometry at Tel Aviv University, by a Sloan Foundation grant 96-6-2 and by a State of New Jersey grant.

†Department of Mathematics, Raymond and Beverly Sackler Faculty of Exact Sciences, Tel Aviv University, Tel Aviv, Israel. Email: sudakov@math.tau.ac.il.

Running head: Two segmentation problems

called *segmentation problems*. In these problems, a company has some information about a set of customers $\mathcal{C}$, and its objective is to choose (and produce) a prescribed number k of policies. The objective is to optimize, over all possible choices of k policies, the sum, over all customers $c \in \mathcal{C}$, of the value assigned by c to the best policy among the policies produced, according to his individual utility function.

Once the k policies are chosen, the set of customers is partitioned into k segments, where segment number i consists of all customers that pick policy number i. It turns out that in many cases, even when the optimization task is trivial for the non-segmented case ($k = 1$), the corresponding optimization problem is NP-hard already for $k = 2$. In the present paper we study two problems of this type. The first one is the following.

THE HYPERCUBE SEGMENTATION PROBLEM: Given a set S of m customers, each a vertex of the discrete d-dimensional cube $\{0,1\}^d$, find k policies $P_1, \ldots, P_k, P_i \in \{0,1\}^d$ and a partition of S into k segments $S_1, \ldots, S_k$ so as to maximize the sum

$$\sum_{i=1}^{k} \sum_{c \in S_i} P_i \odot c,$$

where $\odot$ is the overlap operator between two vertices of the d-dimensional cube, defined to be the number of positions they have in common.

This problem is considered in [9], where the authors show that its precise solution is NP-hard even for $k = 2$. They design two approximation algorithms for the problem: The first is a deterministic algorithm that runs in polynomial time for every fixed k, and produces a solution whose value is within $2\sqrt{2} - 2$ ($= 0.828...$) of the optimum. It is based on the interesting observation that for every set $S \subset \{0,1\}^d$ there is some $P \in S$ so that for every $x \in \{0,1\}^d$,

$$\sum_{c \in S} P \odot c \geq (2\sqrt{2} - 2) \sum_{c \in S} x \odot c.$$

The second is a randomized algorithm that runs in linear time for every fixed k, and produces a solution whose expected value is within 0.7 of the optimum.

Here we design an improved approximation algorithm; for every fixed $\epsilon > 0$ and every fixed k, our algorithm produces in linear time a solution whose value is within $(1-\epsilon)$ of the optimum. Therefore, for every fixed k, this is a polynomial time approximation scheme for the problem. Our algorithm is deterministic, but it is convenient to first describe it as a randomized one and then to derandomize it using some properties of expander graphs. This improves the performance ratio as well as the running time of the deterministic algorithm of [9] for all m, d and k. The randomized algorithm of [9] is slightly faster than ours for large k, but the performance ratio of our algorithm is much better.

The second segmentation problem we consider is the following minimization problem, which is only mentioned briefly in [9].

THE MINIMUM SPANNING TREE SEGMENTATION PROBLEM: Given a connected graph $G = (V, E)$ and n non-negative functions $f_i : E \to \mathbf{R}, 1 \leq i \leq n$, find k spanning trees $T_1, \ldots, T_k$ of G, so

as to minimize the sum

$$\sum_{i=1}^{n} \min_{1 \le j \le k} f_i(T_j),$$

where $f_i(T_j) = \sum_{e \in E(T_j)} f_i(e)$.

We show that unless P=NP, there is no polynomial time algorithm that approximates the optimal solution of this problem up to any finite factor, even if $k = 2$.

The rest of this paper is organized as follows. In section 2 we consider the hypercube segmentation problem, describe our randomized approximation algorithm and present its derandomization. In Section 3 we present the (simple) proof that there is no polynomial time approximation algorithm for the minimum spanning tree segmentation problem, unless P=NP. The final section 4 contains some concluding remarks.

2 The hypercube segmentation problem

In this section we present a polynomial approximation scheme for the hypercube segmentation problem. First we describe a randomized algorithm for this problem and then we show how it can be derandomized, using some properties of random walks on expanders.

2.1 Random sampling

Let $S \subset \{0,1\}^d$ be a set of m customers. Denote by $f(P, S') = \sum_{c \in S'} P \odot c$ the total value of policy $P \in \{0,1\}^d$ for the customer set $S' \subset S$, where $P \odot c$ is the number of coordinates in which P and c agree. A family of policies $P_1, \ldots, P_k$ induces a partition of the entire set S into k segments $S_1, \ldots, S_k$ by putting $c \in S$ into the set S_i if $P_i \odot c \ge P_j \odot c$ for all $j \ne i$ and by breaking ties arbitrarily. It is easy to see that this partition maximizes the value of the expression $\sum_i f(P_i, S_i)$ over all possible partitions of S into k parts $S_1, \ldots, S_k$. Therefore the segmentation problem is equivalent to the problem of finding a family of optimal policies. Note that the optimal value of the problem is clearly at least $md/2$. This value can be produced without segmentation by picking the majority bit in each of the d coordinates.

We first describe a simple randomized approximation algorithm for the hypercube segmentation problem, which for any fixed $\epsilon > 0$ produces a solution whose expected value is within $(1 - \epsilon)$ from optimal. For any fixed k and ϵ the running time of this algorithm is linear.

Algorithm A(k,ϵ)

Input: A set S of m customers, each being a vertex of $\{0,1\}^d$.

1. Sample $l = \Theta(\frac{k}{\epsilon^2})$ customers from S with repetitions, randomly and independently, according to a uniform distribution.

2. For all possible partitions of the sample set into at most k segments **do:**

2.1. For each segment in the partition find an optimal policy: a vector from $\{0,1\}^d$ which in

each coordinate agrees with the majority of the elements of the segment. This gives a family of policies.

2.2. Produce the segmentation of the entire set S according to this family of policies.

3. Let $S_1^*, \ldots, S_k^*$ be the optimal segmentation obtained from all possible partitions of the sample set, (note that some of the sets S_i^* may be empty), and let $P_i^*, 1 \leq i \leq k$ be the corresponding family of policies.

Output (S_i^*, P_i^*) for $1 \leq i \leq k$.

Note that the number of all possible partitions of a set of size l into at most k parts is $O(k^l)$. Therefore, the running time of this algorithm is $O(k^{l+1}md) = e^{O(\frac{k}{\epsilon^2}\ln k)}md$. This is linear in the length of the input md for any fixed k and ϵ and remains polynomial in this length for each k up to $O(\ln(md)/\ln\ln(md))$.

To study the performance of the algorithm, let $P_1, \ldots, P_k$ be the optimal family of policies and let $S_1, \ldots, S_k$ be the corresponding segmentation of S. Denote by X the subset of S obtained by our random sampling. Let $X_1, \ldots, X_k$ be the partition of X defined by $X_i = X \cap S_i$ and let $P_1', \ldots, P_k'$ be the optimal family of policies for this segmentation of X (if X_i is empty, then P_i' can be any vector). Denote by $S_1', \ldots, S_k'$ the partition of S induced by the family $\{P_i'\}$. Since the algorithm $A(k, \epsilon)$ checks all possible partitions of X we conclude that the value of its solution satisfies

$$\sum_i f(P_i^*, S_i^*) \geq \sum_i f(P_i', S_i') \geq \sum_i f(P_i', S_i).$$

It thus suffices to prove that the expected value of $\sum_i f(P_i', S_i)$ is at least $(1-\epsilon)$ of the optimum value $\sum_i f(P_i, S_i)$.

Let S_i be a segment in the optimal partition of S, and let $S_{ij}(r), r = 0, 1$ be the subset of S_i of all vectors, whose j-th coordinate equals to r. Put $\delta_{ij} = \frac{||S_{ij}(0)|-|S_{ij}(1)||}{|S_i|}$. By step 2.1 of the algorithm $A(k, \epsilon)$ the j-th coordinate P_{ij}', in the policy vector P_i' is the majority bit over all the j-th coordinates of the elements from $X \cap S_i$. Thus we obtain that the event $P_{ij}' \neq P_{ij}$ can happen only if the number of elements in X which belong to $S_{ij}(0)$ or to $S_{ij}(1)$ deviates from its expected value by at least $\frac{\delta_{ij}}{2}(\frac{|S_i|l}{m})$. Note that the value $|X \cap S_{ij}(r)|$ is binomially distributed with parameters l and $\frac{|S_{ij}(r)|}{m}$. Therefore, using the standard estimates for Binomial distributions (see e.g., [4], Appendix A) we obtain that

$$Pr(P_{ij}' \neq P_{ij}) \leq e^{-\Omega(\delta_{ij}^2 \frac{|S_i|l}{m})}.$$

Each such event contributes an additive factor of $||S_{ij}(0)|-|S_{ij}(1)|| = \delta_{ij}|S_i|$ to the total difference between the optimal value and the expected result of the algorithm. By the above discussion, this implies that the expected value of this difference is at most

$$\sum_{ij} Pr(P_{ij}' \neq P_{ij})||S_{ij}(0)| - |S_{ij}(1)|| \leq \sum_{ij} \delta_{ij}|S_i| e^{-\Omega(\delta_{ij}^2 \frac{|S_i|l}{m})}.$$

Consider the function $g(t) = te^{-ct^2}$. It is easy to check that $g(t)$ attains its maximum at $t = 1/\sqrt{2c}$ and hence $g(t) \leq c^{-1/2}$ for any real t. By taking c to be $\Theta(\frac{|S_i|l}{m})$ we obtain that $\delta_{ij} e^{-\Omega(\delta_{ij}^2 \frac{|S_i|l}{m})} \leq O(\sqrt{\frac{m}{|S_i|l}})$. Therefore the expected value satisfies

$$E\left(\sum_{i=1}^{k} f(P_i', S_i)\right) \geq \sum_{i=1}^{k} f(P_i, S_i) - \sum_{ij} \delta_{ij}|S_i| e^{-\Omega(\delta_{ij}^2 \frac{|S_i|l}{m})} \geq \sum_{i=1}^{k} f(P_i, S_i) - O\left(\sum_{j=1}^{d} \sqrt{\frac{m}{l}} \sum_{i=1}^{k} \sqrt{|S_i|}\right).$$

By Jensen's inequality $\sum_{i=1}^{k} \sqrt{|S_i|} \leq k\sqrt{\frac{\sum_i |S_i|}{k}} = \sqrt{km}$. Therefore,

$$E\left(\sum_{i=1}^{k} f(P_i', S_i)\right) \geq \sum_{i=1}^{k} f(P_i, S_i) - O\left(\sum_{j=1}^{d} \sqrt{\frac{m}{l}}\sqrt{km}\right) = \sum_{i=1}^{k} f(P_i, S_i) - O\left(md\sqrt{\frac{k}{l}}\right).$$

Using the facts that $l = \Theta(\frac{k}{\epsilon^2})$ and that the optimal value of the hypercube segmentation problem is at least $md/2$, we conclude that with the right choice of the constant in the definition of l,

$$E\left(\sum_{i=1}^{k} f(P_i', S_i)\right) \geq \sum_{i=1}^{k} f(P_i, S_i) - \frac{\epsilon md}{2} \geq (1-\epsilon)\sum_i f(P_i, S_i).$$

This completes the proof that the approximation ratio of the algorithm $A(k, \epsilon)$ is at least $(1 - \epsilon)$.

2.2 Derandomization via random walks

Let $G = (V, E)$ be a connected, non-bipartite, d-regular graph on m vertices. A *random walk* on G is equivalent to a time reversible Markov Chain. The states of the Markov Chain are the vertices of the graph, and for any two vertices u and v the transition probability from u to v, $p_{uv} = 1/d$ if (u, v) is an edge and zero otherwise. Note that by definition the transition probability matrix $P = \frac{1}{d}A$, where A is the adjacency matrix of the graph G, and the uniform distribution is the stationary distribution of this walk (see e.g., [13] for some basic results about random walks on graphs). The eigenvalues of P are reals, and the largest eigenvalue (in absolute value) is 1. We denote the second largest (in absolute value) eigenvalue by λ and define the *eigenvalue gap* to be $\delta = 1 - |\lambda|$. This quantity is directly related to the expansion properties of the graph G (see e.g., [2], [3], [14], [15]). Roughly speaking, δ is large if and only if G is a good expander.

Let U be a subset of vertices of G. Consider a random walk on G starting from a vertex chosen uniformly at random. We denote by t_l the number of times the random walk visits a vertex of U during the first l steps. The following useful result about the behavior of t_l was proved by Gillman [7] (following [1], [5], [8]).

Theorem 2.1 ([7]) *Let $G = (V, E)$ be a connected, regular graph on m vertices with eigenvalue gap δ. Consider a random walk on G starting from a vertex chosen uniformly at random. Let U be an arbitrary subset of vertices of G, $|U| = cm$. Then for any l*

$$Pr(|t_l - cl| \geq \gamma) \leq 4e^{-\frac{\gamma^2 \delta}{20l}}.$$

To use the above result for producing efficient deterministic algorithms, we need an explicit construction of regular graphs with constant degree and large eigenvalue gap. The best known such constructions were given by Lubotzky, Phillips and Sarnak [11] and independently by Margulis [12]. For each $d = p+1$, where p is a prime congruent to 1 modulo 4, they constructed an infinite explicit family of d-regular graphs with $|\lambda| \leq 2\sqrt{d-1}$. (We note that these graphs will not have exactly m vertices for any m, but this does not cause any real problem as we can take a graph on n vertices such that $m \leq n \leq (1+o(1))m$. In this case the number of vertices in the subset $U, |U| = cm$ is still $(c+o(1))n)$. Taking, say, $d = 6$ and $\lambda = 2\sqrt{5}$ we get an eigenvalue gap of $\frac{6-2\sqrt{5}}{6} > 0.25$ and thus we can use such a 6-regular expander for our purposes. Using this construction together with the result of Gillman we obtain the following corollary.

Corollary 2.2 *Given any set S of size m and any natural number l we can construct an explicit family $\mathcal{F}$ of size $6^l m$ of multisets $\{F_i\}, F_i \subset S, |F_i| = l$ with the following property. Let F_i be a multiset, chosen randomly and uniformly from $\mathcal{F}$, then for every subset $U \subset S$ of size cm*

$$Pr(|\ |F_i \cap U| - cl| \geq \gamma) \leq 4e^{-\frac{0.25\gamma^2}{20l}}.$$

Proof. Let G be a 6-regular graph on the vertex set S with eigenvalue gap at least $\frac{6-2\sqrt{5}}{6} > 0.25$. Let $\mathcal{F} = \{F_i\}$ be the family of all possible walks of length l on G. Clearly the size of $\mathcal{F}$ is $6^l m$. A random element F_i of $\mathcal{F}$ corresponds to a random walk of length l on the graph G, which starts in a uniform distribution. Note that by definition $|F_i \cap U| = t_l$. Therefore, Theorem 2.1 completes the proof of the corollary. □

A family of subsets from Corollary 2.2 is the main ingredient of the following deterministic algorithm for the hypercube segmentation problem.

Algorithm B(k,ϵ)

Input: A set S of m customers, each being a vertex of $\{0,1\}^d$.

1. Construct a family $\mathcal{F} = \{F_i\}$ of size $O(6^l m)$ of multisubsets of S, where each $|F_i| = l = \Theta(\frac{k^2}{\epsilon^2})$, satisfying the property in the assertion of Corollary 2.2.

2. For $1 \leq i \leq |\mathcal{F}|$ and **for** all possible partitions of F_i into at most k segments **do:**

2.1. For each segment in the partition find an optimal policy: a vector from $\{0,1\}^d$ which in each coordinate agrees with the majority of the elements of the segment. This gives a family of policies.

2.2. Produce the segmentation of the entire set S according to this family of policies.

3. Let $S_1^*, \ldots, S_k^*$ be the optimal segmentation obtained from all possible partitions of F_i for $1 \leq i \leq |\mathcal{F}|$, (note that some of the sets S_i^* may be empty), and let $P_i^*, 1 \leq i \leq k$ be the corresponding family of the best policies.

Output (S_i^*, P_i^*) for $1 \leq i \leq k$.

The running time of this algorithm is $O(k^{l+1}6^l md) = e^{O(\frac{k^2}{\epsilon^2}\ln k)}md$. Therefore it is linear in md for any fixed k and ϵ and remains polynomial in md for any k up to $O((\ln(md)/\ln\ln(md))^{1/2})$. We claim that the above algorithm produces a solution of value at least $(1-\epsilon)$ of the optimum.

Indeed, consider a multiset $X = F_t$ chosen randomly and independently from the family $\mathcal{F}$. Let $P_1, \ldots, P_k$ be the optimal family of policies and let $S_1, \ldots, S_k$ be the corresponding segmentation of S. Let $X_1, \ldots, X_k$ be the partition of X defined by $X_i = X \cap S_i$ and let $P'_1, \ldots, P'_k$ be the optimal family of policies for this segmentation of X (if X_i is empty, then P'_i can be any vector). Denote by $S'_1, \ldots, S'_k$ the partition of S induced by the family $\{P'_i\}$. As explained in the previous subsection, it suffices to prove that the expected value of $\sum_i f(P'_i, S_i)$ is at least $(1-\epsilon)$ of the optimum value $\sum_i f(P_i, S_i)$.

Let S_i be a segment in the optimal partition of S, and let $S_{ij}(r), r = 0, 1$ be the subset of S_i consisting of all vectors whose j-th coordinate is r. Put $\delta_{ij} = \frac{||S_{ij}(0)|-|S_{ij}(1)||}{|S_i|}$. By step 2.1 of the algorithm $B(k, \epsilon)$ the j-th coordinate P'_{ij}, in the policy vector P'_i is the majority bit over all the j-th coordinates of the elements from $X \cap S_i$. Thus we obtain that the event $P'_{ij} \neq P_{ij}$ can happen only if the number of elements in X which belong to $S_{ij}(0)$ or to $S_{ij}(1)$ deviates from its expected value by at least $\frac{\delta_{ij}}{2}(\frac{|S_i|l}{m})$. Note that by Corollary 2.2 $X = F_t$ satisfies the property that

$$Pr(|\ |X \cap U| - cl| \geq \gamma) \leq 4e^{-\frac{0.25\gamma^2}{20l}},$$

for any subset $U \subset S$ of size cm. Therefore we obtain that

$$Pr(P'_{ij} \neq P_{ij}) \leq e^{-\Omega(\delta_{ij}^2 \frac{|S_i|^2 l}{m^2})} = e^{-\Omega(l(\frac{\delta_{ij}|S_i|}{m})^2)}.$$

Each such event contributes an additive factor of $||S_{ij}(0)|-|S_{ij}(1)|| = \delta_{ij}|S_i|$ to the total difference between the optimal value and the expected result of the algorithm. By the above discussion, this implies that the expected value of this difference is at most

$$\sum_{ij} Pr(P'_{ij} \neq P_{ij})||S_{ij}(0)| - |S_{ij}(1)|| \leq m \sum_{ij} \frac{\delta_{ij}|S_i|}{m} e^{-\Omega(l(\frac{\delta_{ij}|S_i|}{m})^2)}.$$

As mentioned in Subsection 2.1 the function $g(t) = te^{-ct^2} \leq c^{-1/2}$ for any real t. By taking c to be $\Theta(l)$ we obtain that $\frac{\delta_{ij}|S_i|}{m} e^{-\Omega(l(\frac{\delta_{ij}|S_i|}{m})^2)} \leq O(1/\sqrt{l})$. Therefore the expected value satisfies

$$E\left(\sum_{i=1}^{k} f(P'_i, S_i)\right) \geq \sum_{i=1}^{k} f(P_i, S_i) - m \sum_{ij} \frac{\delta_{ij}|S_i|}{m} e^{-\Omega(l(\frac{\delta_{ij}|S_i|}{m})^2)} \geq \sum_{i=1}^{k} f(P_i, S_i) - O(\frac{mkd}{\sqrt{l}}).$$

Using the facts that $l = \Theta(\frac{k^2}{\epsilon^2})$ and that the optimal value of the hypercube segmentation problem is at least $md/2$, we conclude that with the right choice of the constant in the definition of l, the expected value satisfies

$$E\left(\sum_{i=1}^{k} f(P'_i, S_i)\right) \geq \sum_{i=1}^{k} f(P_i, S_i) - \frac{\epsilon md}{2} \geq (1-\epsilon)\sum_i f(P_i, S_i).$$

Thus, there exists a particular t and a partition of $X = F_t$ which produces a segmentation of S whose value is within $(1-\epsilon)$ from optimum. But then the algorithm $B(k,\epsilon)$ will find it in stage 3. This completes the proof of the correctness of the algorithm.

3 The Minimum Spanning Tree segmentation problem

A *hypergraph* H is an ordered pair $H = (V, E)$, where V is a finite set (the *vertex set*), and E is a family of distinct subsets of V (*the edge set*). A hypergraph $H = (V, E)$ is 3-uniform if all edges of H are of size 3. The *chromatic number* of H is the minimum number of colors required to color all its vertices so that no edge is monochromatic. Lovász [10] (see also [6]) showed that it is NP-hard to determine whether a 3-uniform hypergraph is 2-colorable.

In this section we present a construction for reducing the 2-colorability problem for 3-uniform hypergraphs to the segmented version of the minimum spanning tree. Using this construction we deduce that unless $P = NP$ the minimum spanning tree segmentation problem does not have any polynomial time approximation even for an extremely simple graph - a path with three parallel edges between each pair of consecutive nodes.

Suppose we are given a 3-uniform hypergraph $H = (V(H), E(H))$ with $|E(H)| = m$ edges. Let $G = (V(G), E(G))$ be a path of length m with three parallel edges between each pair of consecutive nodes. Each triple e_u, e_v, e_w of parallel edges in G corresponds to an edge (u, v, w) of the hypergraph H and each edge in the triple is labeled by a vertex of the edge (u, v, w). For every vertex $u \in V(H)$ define a weight function f_u on the edges of G with $f_u(e), e \in E(G)$ being one if and only if the edge e is labeled by the vertex u and $f_u(e) = 0$ otherwise. We claim that the chromatic number of the hypergraph H is equal to two if and only if there exists a pair T_1, T_2 of spanning trees of G such that

$$\sum_{u \in V(H)} \min_{1 \le i \le 2} f_u(T_i) = 0.$$

To prove this claim assume, first, that $c : V(H) \to \{1, 2\}$ is a 2-coloring of the vertices of H with no monochromatic edges. Denote by $G_i, i = 1, 2$ the subgraph of G spanned by all edges, corresponding to the vertices of H with color i. Since no edge of the hypergraph H is monochromatic we obtain that from every triple of parallel edges in G at least one belongs to $G_i, i = 1, 2$. Therefore each of the subgraphs G_1 and G_2 contains a spanning tree T_i of G. Thus it is enough to prove that for any $u \in V(H)$ at least one of the values $f_u(G_i), i = 1, 2$ is equal to zero. To do so, consider G_i for $i = 3 - c(u)$. By definition G_i contains no edges corresponding to u and thus $f_u(G_i) = 0$. This implies that

$$0 \le \sum_{u \in V(H)} \min_{1 \le i \le 2} f_u(T_i) \le \sum_{u \in V(H)} \min_{1 \le i \le 2} f_u(G_i) = 0.$$

Now assume that there exists a pair T_1 and T_2 of spanning trees of G such that $\min_{1 \le i \le 2} f_u(T_i) = 0$ for every vertex $u \in V(H)$. Let $V_i, i = 1, 2$ be the subset of vertices of H which correspond to the

labels of the edges in the tree T_i. Consider a vertex coloring of the hypergraph H by two colors such that all vertices in $V_i, i = 1, 2$ are colored by the color i and all the remaining vertices are colored arbitrarily. We need to prove that no edge of H is monochromatic and no vertex gets two colors simultaneously. This will imply that the hypergraph H is 2-colorable. Note first, that the subsets V_1 and V_2, are disjoint. Indeed, assume this is false and let u be a vertex of H which belongs to $V_1 \cap V_2$. Then the spanning trees T_1 and T_2 both contain an edge of G which is labeled by the vertex u. By the definition of the function f_u it follows that $f_u(T_1), f_u(T_2) > 0$, contradiction. Hence each vertex of H gets only one color. Since T_i is a spanning tree of G it has at least one edge from every triple of parallel edges in G. Therefore $V_i, i = 1, 2$ intersects every edge of the hypergraph H in at least one vertex. This implies that there are no monochromatic edges.

Applying now the result of Lovász, we get that if $P \neq NP$, then there is no polynomial algorithm to decide whether the optimal value in the minimum spanning tree segmentation problem is strictly positive, even for the case of two trees. This implies the following.

Theorem 3.1 *Given a connected graph $G = (V, E), |V| = n$ and a family $f_i, 1 \leq i \leq m$ of nonnegative weight functions on $E(G)$, it is impossible to approximate in polynomial time in m and n the optimal value of the minimum spanning tree segmentation problem for G (even for $k = 2$) within any finite factor, unless $P = NP$.*

4 Concluding remarks

The class of segmentation problems contains several interesting algorithmic questions, and our present paper deals with two of them.

It is not difficult to extend the problem and results of Section 2 to hypercubes over a larger fixed alphabet. We omit the simple details.

The hardness result for the Minimum Spanning Tree segmentation problem holds, as mentioned in Section 3, even if the input graph is the path with three parallel edges between every two consecutive vertices. Similarly, it holds for many other graphs, including every n vertex graph that contains a spanning subgraph which is a subdivision of a graph obtained from any path by replacing each of $\Omega(n^\delta)$ edges by three parallel ones. This includes the graphs of the d-cubes, as well as many other ones.

Some of our techniques here are useful in the study of other segmentation problems. One of these is the *catalog segmentation problem* (in the dense case) considered in [9]. Suppose that we have a set of n customers and a set U of m items. For each customer we are given a subset of items this customer likes. We wish to create a catalog with r items to be sent to the customers. Our objective is to maximize the sum, over all items, of the number of customers that like this item. The simple optimal solution is, of course, to select the r most popular items. However, if instead of one catalog we can create k different ones, each with r items, sending one of them to each customer, we can

sometimes ensure a much bigger value than that given by a single catalog. This leads to the catalog segmentation problem, whose precise formulation is the following.
THE CATALOG SEGMENTATION PROBLEM. Given a set U of size m and a family $S_1, \ldots, S_n$ of n subsets of U, find k subsets $X_1, \ldots, X_k$ of U, each of size r, so as to maximize the sum

$$\sum_{i=1}^{n} \max_{1 \le j \le k} |S_i \cap X_j|.$$

This problem is considered in [9], where the authors prove that it is NP-hard even for $k = 2$. For fixed k, and for the special (dense) case in which each customer likes at least a fraction ϵ of all items (that is, $|S_i| \ge \epsilon |U|$ for all i), they design a randomized polynomial time approximation scheme. Our technique here can be used to provide a *deterministic* polynomial time approximation scheme for this special case. Without the density assumption, the problem appears to be much more difficult, and as mentioned in [9], even the problem of improving the trivial 1/2 approximation for $k = 2$ in polynomial time seems difficult.

References

[1] M. Ajtai, J. Komlós and E. Szemerédi, Deterministic Simulation in Logspace, *Proc. of the* 19^{th} *ACM STOC*, ACM Press (1987), 132-140.

[2] N. Alon, Eigenvalues and expanders, *Combinatorica* 6 (1986), 83-96.

[3] N. Alon and V. D. Milman, Eigenvalues, expanders and superconcentrators, *Proc.* 25^{th} *Annual Symp. on Foundations of Computer Science*, Singer Island, Florida, IEEE (1984), 320-322. (Also: λ_1, isoperimetric inequalities for graphs and superconcentrators, *J. Combinatorial Theory, Ser. B* 38 (1985), 73-88.)

[4] N. Alon and J. Spencer, **The Probabilistic Method**, Wiley, New York, 1992.

[5] A. Cohen and A. Wigderson, Dispersers, deterministic amplification, and weak random sources, *Proc. of the* 30^{th} *IEEE FOCS*, IEEE (1989), 14-19.

[6] M.R. Garey and D.S. Johnson, **Computers and Intractability: a Guide to the Theory of $\mathcal{NP}$-Completeness**, Freeman, New York, 1979.

[7] D. Gillman, A Chernoff bound for random walks on expander graphs, *Proc. of the* 34^{th} *IEEE FOCS*, IEEE (1993), 680-691; (Also: *SIAM J. of Computing* 27 (1998), 1203-1220).

[8] R. Impagliazzo and D. Zuckerman, How to Recycle Random Bits, *Proc. of the* 30^{th} *IEEE FOCS*, IEEE (1989), 248-253.

[9] J. Kleinberg, C. Papadimitriou and P. Raghavan, Segmentation problems, *Proc. of the* 30^{th} *ACM STOC*, ACM Press (1998), 473-482.

[10] L. Lovász, Coverings and colorings of hypergraphs, *Proc.* 4^{th} *S.E. Conf. on Combinatorics, Graph Theory and Computing*, 1973, Utilitas Math., 3-12.

[11] A. Lubotzky, R. Phillips and P. Sarnak, Explicit expanders and the Ramanujan conjectures, *Proc. of the* 18^{th} *ACM Symp. on the Theory of Computing*, (1986), 240-246; (Also: Ramanujan graphs, *Combinatorica* 8 (1988), 261-277).

[12] G. A. Margulis, Explicit group-theoretical constructions of combinatorial schemes and their application to the design of expanders and superconcentrators, *Problemy Peredachi Informatsii*, 24 (1988), 51-60 (in Russian). (English translation in *Problems of Information Transmission*, 24 (1988), 39-46).

[13] R. Motwani and P. Raghavan, **Randomized Algorithms**, Cambridge University Press, 1995.

[14] A. Sinclair and M. R. Jerrum, Approximate counting, uniform generation and rapidly mixing Markov chains, *Information and Computation* 82 (1989), 93-133.

[15] R. M. Tanner, Explicit construction of concentrators from generalized N-gons, *SIAM J. Alg. Disc. Meth.* 5 (1984), 287-293.