The Hardness of Metric Labeling

Julia Chuzhoy* Joseph (Seffi) Naor†

Abstract

The Metric Labeling problem is an elegant and powerful mathematical model capturing a wide range of classification problems. The input to the problem consists of a set of labels and a weighted graph. Additionally, a metric distance function on the labels is defined, and for each label and each vertex, an assignment cost is given. The goal is to find a minimum-cost assignment of the vertices to the labels. The cost of the solution consists of two parts: the assignment costs of the vertices and the separation costs of the edges (each edge pays its weight times the distance between the two labels to which its endpoints are assigned).

Due to the simple structure and variety of the applications, the problem and its special cases (with various distance functions on the labels) have recently received much attention. Metric Labeling has a known logarithmic approximation, and it has been an open question for several years whether a constant approximation exists. We refute this possibility and show that no constant approximation can be obtained for the problem unless P=NP, and we also show that the problem is $\Omega(\sqrt{\log n})$-hard to approximate, unless NP has quasi-polynomial time algorithms.

1 Introduction

The metric labeling problem, introduced by Kleinberg and Tardos [13], captures a broad range of classification problems that arise in computer vision and related fields. In such classification problems, labels from a given set L are assigned to a set V of n objects on which a pairwise relationship is defined. The pairwise relationships between the objects are represented by a weighted undirected graph $G = (V, E)$, where $w(u, v)$ represents the strength of the relationship between u and v. We assume that a *metric* distance function d is defined on the label set. The objective is to find a labeling, a function $f : V \to L$, that maps objects to labels, where the cost of f, denoted by $Q(f)$, has two components.

- For each $v \in V$, $\ell \in L$, there is a non-negative assignment cost $c(v, \ell)$ for labeling vertex v with label ℓ.
- For each edge $e = (u, v) \in E$, the cost of labeling (u, v) by $(f(u), f(v))$ is $w(u, v) \cdot d(f(u), f(v))$.

Thus,

$$Q(f) = \sum_{u \in V} c(u, f(u)) + \sum_{(u,v) \in E} w(u, v) \cdot d(f(u), f(v))$$

and the goal is to find a labeling f of minimum cost.

Metric labeling has rich connections to some well known problems in combinatorial optimization. A special case of metric labeling is the *0-extension* problem, studied by Karzanov [11, 12]. There are no assignment costs in this problem, however, the graph contains a set of terminals, $t_1, \ldots, t_k$, where the label of terminal t_i is fixed in advance to i, and the non-terminals are free to be assigned to any of the labels. As in the metric labeling problem, a metric is defined on the set of labels. Clearly, the 0-extension problem generalizes the well-studied *multiway cut* problem [7, 4, 10] in which the metric on the label set is the uniform metric.

Kleinberg and Tardos [13] obtained an $O(\log k \log \log k)$-approximation algorithm for the general metric labeling problem, where k denotes the number of labels in L, using the probabilistic tree embedding technique [2, 3]. This bound was recently improved to $O(\log k)$ [8]. Kleinberg and Tardos [13] also gave a 2-approximation for the uniform metric using a linear programming formulation.

Chekuri et al. [6] gave a natural linear programming formulation for the general metric labeling problem. A solution to this linear program is an embedding of the graph in a k-dimensional simplex, where the distance between points in the simplex is defined by a special metric, the *earth mover's metric* (EMD), and not by the (standard) ℓ_1 metric. Chekuri et al. [6] showed that the integrality gap of the formulation for general metrics is at most the distortion of a probabilistic tree embedding of the given metric d, i.e., $O(\log k)$ [8].

*Computer Science Dept., Technion, Haifa 32000, Israel. E-mail: cjulia@cs.technion.ac.il.

†Computer Science Dept., Technion, Haifa 32000, Israel. E-mail: naor@cs.technion.ac.il. Research supported in part by US-Israel BSF Grant 2002276 and by EU contract IST-1999-14084 (APPOL II).

Călinescu et al. [5] considered approximation algorithms for the 0-extension problem via the metric relaxation linear programming formulation, originally studied by Karzanov [11], and obtained an $O(\log k)$-approximation algorithm for general metrics. We note that their formulation does not apply to the metric labeling problem. A lower bound of $\Omega(\sqrt{\log k})$ on the integrality ratio of the metric relaxation was also established by [5]. However, the proof of this lower bound does not seem to carry over in any straight forward way when using the linear programming formulation of [6] specialized to the 0-extension problem.

Our Results A question that has intrigued many researchers since the appearance of [13] is whether there exists a constant factor approximation algorithm for the metric labeling problem. We show that there is no constant factor approximation for metric labeling if P $\neq$ NP, and an $\Omega(\sqrt{\log n})$-hardness if NP $\nsubseteq$ DTIME $\left(n^{\mathrm{poly}(\log n)}\right)$. In fact, we show that the result even holds for the special case of metric labeling called $(0,\infty)$-*extension*. In this problem, the assignment costs of the vertices are either 0 or ∞, or equivalently, each vertex $v \in V$ has a list of labels, $L(v)$, to which it is allowed to be assigned. The cost of the solution then only consists of the edge separation cost. We note that Chekuri et al. [6] have shown that the $(0,\infty)$-extension problem is equivalent to the general metric labeling problem.

Organization We start in Section 3 with a simple $(3-\delta)$-hardness proof (for any constant $0 < \delta < 1$) for the $(0,\infty)$-extension problem. This proof provides the intuition and motivation for the new techniques and ideas needed to obtain the stronger hardness bounds shown in Section 4.

2 Preliminaries

We perform our reduction to metric labeling from the gap version of Max 3SAT(5). The input to the problem is a CNF formula φ with n variables and $\frac{5n}{3}$ clauses. Each clause consists of 3 literals and each variable participates in 5 clauses, appearing in each clause at most once.

Let ϵ, $0 < \epsilon < 1$, be a constant and let φ be an instance of Max 3SAT(5). Then φ is called a *Yes-instance* if there is an assignment that satisfies all the clauses, and it is called a *No-instance* (with respect to ϵ) if any assignment satisfies at most a fraction $(1-\epsilon)$ of the clauses. The following well known theorem was proved by [1].

Theorem 2.1 *There is a constant ϵ, $0 < \epsilon < 1$, such that it is NP-hard to distinguish between Yes-instances and No-instances of the Max 3SAT(5) problem.*

For the sake of completeness, we provide a description of the following standard two-prover protocol for the Max 3SAT(5) problem. Given a 3SAT(5) formula φ:

- The verifier randomly chooses a clause C from the formula φ and one of the variables x belonging to C. Variable x is called the *distinguished variable*.
- Prover 1 receives clause C and is expected to return an assignment to all the variables appearing in the clause. Prover 2 receives variable x and is expected to return an assignment to x.
- After receiving the answers of the provers, the verifier checks that the answer of prover 1 defines a satisfying assignment to clause C and that the assignments of prover 1 and prover 2 to variable x are identical.

The following well known theorem follows from Theorem 2.1.

Theorem 2.2 *If φ is a Yes-instance, then there is a strategy of the provers such that the verifier always accepts. If φ is a No-instance, then for any strategy of the provers, the acceptance probability is at most $(1-\frac{\epsilon}{3})$.*

3 A Simple $(3-\delta)$ Hardness

In this section we present a simple $(3-\delta)$-hardness for the $(0,\infty)$-extension problem (for any constant $0 < \delta < 1$) and also provide some intuition as to the new ideas needed to improve this bound.

We start by amplifying the soundness of the 2-prover protocol presented above by means of parallel repetitions of the protocol, a usual practice in PCP reductions. The number of repetitions is a sufficiently large constant ℓ. The new protocol proceeds as follows.

- The verifier chooses, randomly and independently, ℓ clauses $C_1,\ldots,C_\ell$ from the input formula φ. For each i, $1 \leq i \leq \ell$, the verifier chooses, randomly and independently, one variable x_i belonging to C_i.
- Prover 1 receives clauses $C_1,\ldots,C_\ell$ and is expected to return an assignment to all the variables appearing in the clauses, such that all clauses are satisfied. Prover 2 receives variables $x_1,\ldots,x_\ell$ and is expected to return an assignment to these variables.
- After receiving the answers of the provers, the verifier checks that the answer of prover 1 defines satisfying assignments to clauses $C_1,\ldots,C_\ell$ and that the assignments of prover 1 and prover 2 to variables $x_1,\ldots,x_\ell$ are identical.

The following theorem follows from the well known Raz parallel repetition theorem [14], which bounds the error probability of the above protocol.

Theorem 3.1 *If φ is a Yes-instance, then there is a strategy of the provers such that the verifier always accepts. If φ is a No-instance, then for any strategy of the provers, the acceptance probability is at most $2^{-\alpha\ell}$ for some universal constant α.*

Let Q_1 denote the set of all the possible queries to prover 1 (i.e., each query $q \in Q_1$ is an ℓ-tuple of clauses). Given a query $q_1 \in Q_1$, let $A(q_1)$ denote the set of all the possible answers of prover 1 to query q_1, i.e., $A(q_1)$ is the set of all the possible assignments to the variables that appear in the clauses of q_1 that satisfy these clauses. Similarly, Q_2 denotes the set of all the possible queries to prover 2 (each query is an ℓ-tuple of variables), and given $q_2 \in Q_2$, $A(q_2)$ is the set of all the possible answers of prover 2 to query q_2.

The set of labels is defined as follows. For every possible query to each one of the provers and for every possible answer to this query, there is a label, i.e.,

$$L = \{\ell(q, A) \mid q \in Q_1 \cup Q_2, A \in A(q)\}$$

The metric distance function on the labels is defined by a label graph G_L. The vertices of this graph are the labels, and the metric distance between the labels is the length of the shortest path in this graph. Consider some random string r of the verifier, and let q_1, q_2 be the queries sent to the provers when the verifier chooses r. Let $A_1 \in A(q_1), A_2 \in A(q_2)$ be a pair of consistent answers to these queries. Then there is an edge of length 1 between $\ell(q_1, A_1)$ and $\ell(q_2, A_2)$ in G_L. Note that since each edge connects a label belonging to prover 1 and a label belonging to prover 2, the graph is bipartite. Therefore, for any random string r, if $q_1 \in Q_1$, $q_2 \in Q_2$ are the queries sent to the two provers when the verifier chooses r, and $A_1 \in A(q_1)$, $A_2 \in A(q_2)$ are inconsistent answers to these queries, then the distance between labels $\ell(q_1, A_1)$ and $\ell(q_2, A_2)$ in graph G_L is at least 3.

We now proceed to define the input graph. For every query $q \in Q_1 \cup Q_2$, there is a vertex $v(q)$. This vertex can be assigned only to those labels that correspond to this query, i.e.,

$$V = \{v(q) \mid q \in Q_1 \cup Q_2\}$$

$$L(v(q)) = \{\ell(q, A) \mid A \in A(q)\}$$

The edge set consists of edges connecting every pair of vertices $v(q_1), v(q_2)$, such that $q_1 \in Q_1$, $q_2 \in Q_2$, and for some random string of the verifier, the queries q_1 and q_2 are sent to provers 1 and 2. All edges have unit weight. Note that for each random string of the verifier there is exactly one edge corresponding to it.

Yes-instance If φ is a Yes-instance, then there is a strategy of the provers such that their answers are always accepted by the verifier. This strategy defines the assignments of the vertices to the labels, namely vertex $v(q)$ for $q \in Q_1 \cup Q_2$ is assigned to label $\ell(q, A)$, where $A \in A(q)$ is the answer of the corresponding prover to query q under the above strategy. Consider some random string r of the verifier and the queries $q_1 \in Q_1, q_2 \in Q_2$ that are sent to the provers when the verifier chooses r. Let $A_1 \in A(q_1)$, $A_2 \in A(q_2)$ be the answers of the provers according to the above strategy. Note that vertices $v(q_1)$, $v(q_2)$ are assigned to labels $\ell(q_1, A_1)$, $\ell(q_2, A_2)$ and that the answers A_1 and A_2 of the provers are consistent. Therefore, there is an edge in the label graph between the labels $\ell(q_1, A_1)$ and $\ell(q_2, A_2)$, and thus the distance between the two labels (and the cost incurred by the edge between $v(q_1)$ and $v(q_2)$) is 1. The total cost of the solution is therefore $|R|$, where R is the set of all the random strings of the verifier.

No-instance Consider any solution to the problem. Note that the assignments of the vertices to the labels define a strategy of the provers (the assignment of vertex $v(q)$, $q \in Q_1 \cup Q_2$ to label $\ell(q, A)$, $A \in A(q)$, implies that the answer of the corresponding prover to query q is A). Let $R' \subseteq R$ be the set of random strings of the verifier for which the answers of the two provers are inconsistent. Following Theorem 3.1, $|R'| \geq (1 - 2^{-\alpha\ell})|R|$. Consider such a random string $r \in R'$ and let q_1, q_2 be the queries that are sent to the provers given r. Let $\ell(q_1, A_1)$, $\ell(q_2, A_2)$ be the labels to which the vertices $v(q_1)$, $v(q_2)$ are assigned. As the answers A_1, A_2 of the provers are inconsistent, the distance between the two labels (and hence the cost of the edge between $v(q_1)$ and $v(q_2)$) is at least 3. Therefore, the total cost of the solution is at least $3(1 - 2^{-\alpha\ell})|R|) = 3(1 - \delta)|R|$, where δ is an arbitrarily small constant.

As the gap between the costs of Yes and No instances is $3(1-\delta)$, and the size of the construction is polynomial in n, we have that $(0, \infty)$-extension is $3(1 - \delta)$-hard to approximate for any constant δ, unless P=NP.

It is not hard to see that the analysis is tight. Given a pair of labels $\ell(q_1, A_1)$, $\ell(q_2, A_2)$, such that for some random string of the verifier, queries q_1 and q_2 are sent to the two provers and the answers A_1, A_2 to these queries are inconsistent, we show that there is a path of length 3 in graph G_L between these two labels. Let $q_1 = (C_{i_1}, \ldots, C_{i_\ell})$ and $q_2 = (x_{i_1}, \ldots, x_{i_\ell})$. Note that for each $j : 1 \leq j \leq \ell$, x_{i_j} is one of the variables of clause C_{i_j}. Let x'_{i_j} and x''_{i_j} denote the other two variables. The path of length 3 between the two labels is $\langle \ell(q_1, A_1), \ell(q'_2, A'_2), \ell(q_1, A'_1), \ell(q_2, A_2) \rangle$, where $q'_2 = (x'_{i_1}, \ldots, x'_{i_\ell})$ and A'_2 contains assignments to $(x'_{i_1}, \ldots, x'_{i_\ell})$ identical to those in A_1. For each $j : 1 \leq j \leq \ell$, the assignment to clause C_{i_j} that appears in A'_1 is as follows. The assignment to x_{i_j} is the same as in A_2, the assignment to x'_{i_j} is the same as in A'_2, and the assignment to x''_{i_j} is set in such a way that clause C_{i_j} is satisfied.

One can see that even though the answers A_1 and A_2 of the provers might be inconsistent in many coordinates, there is still a short path between the two labels. In order to improve hardness, it would be useful to ensure that if two answers are inconsistent in almost all the coordinates, the length of the shortest path between the two corresponding labels is $\Omega(\ell)$ (so in a way we "correct" one coordinate at a time). This is the intuition behind the construction and the k-prover protocol in the next section.

4 The Main Hardness Result

In this section we show $\Omega(\sqrt{\log n})$ hardness of $(0, \infty)$-extension. We start by defining a new k-prover protocol to 3SAT(5). The protocol is then used in a way similar to section 3 construction to obtain a better hardness result.

4.1 A New k-Prover Protocol

We define a new k-prover protocol which is based on the basic two-prover protocol. We use the new protocol in our construction setting $k = \text{poly}(\log n)$. We denote the provers by $P_1, \ldots, P_k$. The protocol is as follows.

- For each (i, j), $1 \leq i < j \leq k$, the verifier chooses, randomly and independently, a clause C_{ij} and a distinguished variable x_{ij} belonging to the clause. Prover P_i is then sent the clause C_{ij} (and is expected to return an assignment to all the variables appearing in the clause), and prover P_j is sent the variable x_{ij} (and is expected to return an assignment to the variable). Each prover P_a, for $1 < a < k$ and $a \neq i, j$, is sent both clause C_{ij} and variable x_{ij} and is expected to return an assignment to all the variables appearing in C_{ij}. Thus, a query q sent to prover P_i has $\binom{k}{2}$ coordinates. Coordinate (a, b) of the query, $a < b$, is the following:
  - if $i = a$, then the coordinate contains C_{ab}.
  - if $i = b$, then the coordinate contains x_{ab}.
  - if $i \neq a, b$, then the coordinate contains both C_{ab} and x_{ab}.
- After receiving the answers of the provers, the verifier checks, for each coordinate (i, j), $1 \leq i < j \leq k$, that the answers of all the provers are consistent, i.e., all the provers P_a, $a \neq j$, return an identical assignment to the variables of C_{ij}, and the assignment of prover P_j to variable x_{ij} matches the assignments of all the other provers.

We note that our k-prover system departs from standard protocols in several ways. First, we do not use Parallel Repetitions theorem here, and there is no need to amplify the soundness of the protocol. Observe also that for each prover P_a, for each coordinate $(i, j) : i, j \neq a$, the prover receives both the clause C_{ij} and the distinguished variable x_{ij}. Clearly, some of the information the prover receives is redundant. Indeed, in k-prover systems (e.g., [9]), the provers usually receive either the clause or the distinguished variable, but not both. However, this sending of redundant information to the provers is essential for our reduction. Intuitively, it will ensure that if, for some random string r, the answers of the k provers are inconsistent in many coordinates, then the distances between the corresponding labels are long.

We denote the set of all the random strings of the verifier by R. Given a random string $r \in R$, for each i, $1 \leq i \leq k$, let $q_i(r)$ be the query sent to prover P_i when the verifier chooses the random string r, and let Q_i be the set of all the possible queries of prover i. For each $i : 1 \leq i \leq k$, for each $q_i \in Q_i$, let $A(q_i)$ denote the set of all the possible answers of prover P_i to query q_i, which satisfy all the clauses appearing in the query.

Definition 4.1 *Consider a pair of provers P_i and P_j, $1 \leq i < j \leq k$, and let $q_i \in Q_i$, $q_j \in Q_j$ be a pair of queries, such that for some random string $r \in R$, $q_i = q_i(r), q_j = q_j(r)$. Let A_i and A_j denote the answers of the provers to the queries. We say that the answers are* weakly consistent *if the assignments to C_{ij} and x_{ij} in A_i and A_j respectively are consistent. The answers are called* strongly consistent *if they are also consistent in every other coordinate, i.e., for each (a, b), $1 \leq a < b \leq k$, where $(a, b) \neq (i, j)$:*

- *If both coordinates $q_i(a, b)$ and $q_j(a, b)$ contain clause C_{ab} and variable x_{ab}, then the assignments to clause C_{ab} in A_i and A_j are identical.*
- *If one of the coordinates $q_i(a, b)$ and $q_j(a, b)$ contains clause C_{ab} and the other contains clause C_{ab} and variable x_{ab}, then the assignments to clause C_{ab} in A_i and A_j are identical.*
- *If one of the coordinates $q_i(a, b)$ and $q_j(a, b)$ contains variable x_{ab} and the other contains clause C_{ab} and variable x_{ab}, then the assignments to clause C_{ab} and variable $x_{a,b}$ in A_i and A_j are consistent.*

Theorem 4.2 *If φ is a Yes-instance, then there is a strategy of the k provers such that the verifier always accepts. If φ is a No-instance, then for any strategy of the provers, for every pair of provers P_i and P_j, $i < j$, the probability that their answers are weakly consistent is at most $(1 - \frac{\epsilon}{3})$.*

Proof. Assume otherwise. Let P_i and P_j be a pair of provers such that the probability that their answers are weakly consistent is more than $(1-\frac{\epsilon}{3})$. We partition the set of random strings R into classes, such that within each class the random strings are identical except for the clause C_{ij} and the distinguished variable x_{ij}. Each such class, (together with the corresponding queries and answers to the queries), can be viewed as a two-prover protocol (while we ignore all the coordinates of the queries and the answers except for (i,j)). As the probability of obtaining a pair of weakly consistent answers is more than $(1-\frac{\epsilon}{3})$, at least for one of the classes, the probability that the verifier accepts is greater than $(1-\frac{\epsilon}{3})$. This defines a strategy for the two-prover protocol, where the acceptance probability by the verifier is greater than $(1-\frac{\epsilon}{3})$, contradicting Theorem 2.2. □

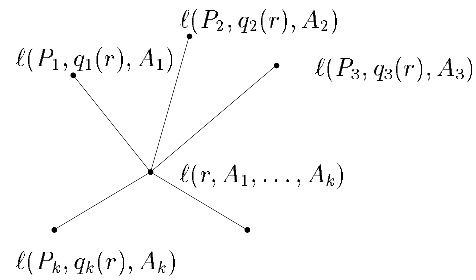

Figure 1. Edges in the graph of labels incident to $\ell(r, A_1, \ldots, A_k)$

4.2 The Graph and the Label Set

In this section we construct from a 3SAT(5) formula φ an instance of the $(0,\infty)$-extension problem. Our construction is based on the k-prover system described above.

The set of labels L consists of two subsets:

Query Labels: for each prover P_i, $1 \le i \le k$, for each query $q \in Q_i$, and for each answer $A \in A(q)$ to the query q, there is a label $\ell(P_i, q, A)$.

Constraint Labels: consider a random string r of the verifier. Let $A_1, \ldots, A_k$, be any collection of possible answers of the provers to the queries $q_1(r), \ldots, q_k(r)$, i.e., for each $1 \le i \le k$, $A_i \in A(q_i(r))$. Moreover, assume that these answers are accepted by the verifier, (i.e., $A_1, \ldots, A_k$ are strongly consistent). Then, there is a label $\ell(r, A_1, A_2, \ldots, A_k)$.

We now define a graph $G_L(L, E')$ on the label set. The metric on the label set is implied by the shortest path distance function in the graph. The vertices of G_L are the labels and the edges are defined as follows. Consider a constraint label $\ell = \ell(r, A_1, A_2, \ldots, A_k)$, Then, for each i, $1 \le i \le k$, there is an edge of length $\frac{1}{2}$ between ℓ and $\ell(P_i, q_i(r), A_i)$.

Thus, the graph is a collection of stars, while some stars share some of their leaves (see Figure 1).

We now proceed to define a graph $G(V, E)$. The vertex set V is the union of two vertex sets: a set of *query vertices*, denoted by V_1, and a set of *constraint vertices*, denoted by V_2.

Query Vertices: for each prover P_i, $1 \le i \le k$, and for each query $q \in Q_i$, there is a vertex $v(P_i, q)$. Thus,

$$V_1 = \{v(P_i, q) \mid 1 \le i \le k \text{ and } q \in Q_i\}$$

Vertex $v(P_i, q)$ can only be assigned to the labels corresponding to (P_i, q_i), i.e.,

$$L(v(P_i, q)) = \{\ell(P_i, q, A) \mid A \in A(q)\}$$

Note that assigning $v(P_i, q)$ to a label in $L(v(P_i, q))$ defines an answer of prover P_i to query q.

Constraint Vertices: for each random string r, there is a vertex $v(r)$, i.e.,

$$V_2 = \{v(r) \mid r \in R\}$$

Vertex $v(r)$ can be assigned only to labels corresponding to r, i.e., $L(v(r))$ consists of labels $\ell(r, A_i, \ldots, A_k)$, such that $\forall i,\ A_i \in A(q_i(r))$ and $(A_1, \ldots A_k)$ are strongly consistent.

The edges of the graph are as follows. Every constraint vertex $v(r)$ is connected to every assignment vertex $v(P_i, q_i(r))$ by a unit-weight edge (see Figure 2).

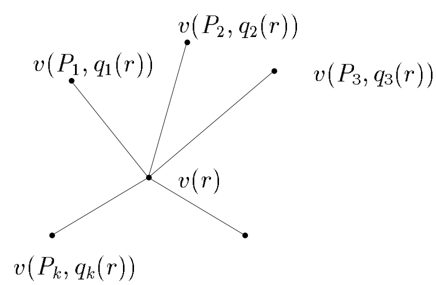

Figure 2. Edges incident to $v(r)$

The graph is therefore a collection of stars that can have common leaves.

4.3 Hardness of Approximation Proof

4.3.1 Yes-Instances

We assume that formula φ is a yes-instance. Consider a strategy of the provers for which the acceptance probability of the verifier is 1. For every prover P_i, $1 \le i \le k$, for every query $q \in Q_i$, let $f(q) \in A(q)$ be the answer of prover P_i to query q under this strategy. (Clearly, f is derived from the satisfying assignment to φ.) Note that for each random string r, $f(q_1(r)), \ldots, f(q_k(r))$ are strongly consistent. We define the following labeling of the graph G (see Figure 3).

- For each random string $r \in R$, vertex $v(r)$ is assigned to label $\ell(r, f(q_1(r)), \ldots, f(q_k(r)))$.
- For each $i : 1 \le i \le k$, $q \in Q_i$, vertex $v(P_i, q)$ is assigned to label $\ell(P_i, q, f(q))$.

Consider an edge in the graph G between $v(r)$ and $v(P_i, q_i(r))$, $r \in R$, $1 \le i \le k$. Vertex $v(r)$ is assigned to label $\ell(r, f(q_1(r)), \ldots, f(q_k(r)))$ and vertex $v(P_i, q_i(r))$ is assigned to label $\ell((P_i, q_i(r), f(q_i(r)))$. Thus, the separation cost of the edge is $\frac{1}{2}$, since the distance between the two labels is $\frac{1}{2}$. Hence, the total cost of the solution is $\frac{1}{2} \cdot k \cdot |R|$.

4.3.2 No-Instances

We assume that formula φ is a no-instance. We prove that the cost of any solution to the metric labeling instance is at least $\binom{k}{2} \cdot \frac{\epsilon}{3} \cdot |R|$. Observe that the assignment of the query vertices to query labels defines a strategy of the provers. We concentrate on this strategy and define the set $T \subseteq R \times [k] \times [k]$.

Definition 4.3 *For $r \in R$, $1 \le i < j \le k$, $(r, i, j) \in T \subseteq R \times [k] \times [k]$ if and only if the answers of provers P_i and P_j to queries $q_i(r)$ and $q_j(r)$, respectively, (under the above strategy) are not weakly consistent.*

The following proposition is a direct consequence of Theorem 4.2.

Proposition 4.4 $|T| \ge \binom{k}{2} \cdot \frac{\epsilon}{3} \cdot |R|$.

Consider an edge $e \in E$ and assume that the endpoints of the edge are assigned to labels ℓ_1 and ℓ_2. We denote by $\mathcal{P}_e$ the shortest path between the labels ℓ_1 and ℓ_2 in the graph of labels G_L. Note that the length of $\mathcal{P}_e$ is exactly the cost paid by edge e, and the solution cost is $\sum_{e \in E} |\mathcal{P}_e|$. We define the set $T' \subseteq R \times [k] \times [k]$ as follows. Consider a random string $r \in R$ and a pair P_i and P_j, $1 \le i, j \le k$, $i \ne j$ of provers. Let e be the edge between $v(r)$ and $v(P_i, q_i(r))$. Then, $(r, i, j) \in T'$ if and only if the path $\mathcal{P}_e$ contains a label belonging to prover P_j (i.e., a label of the form $\ell(P_j, q, A)$, for some $q \in Q_j, A \in A(q)$). Observe that the cost of the solution is at least $|T'|$.

Lemma 4.5 *For $r \in R$, suppose $(r, i, j) \in T$, where $1 \le i < j \le k$. Then, either $(r, i, j) \in T'$, or $(r, j, i) \in T'$.*

Proof. Suppose that vertex $v(r)$ is assigned to label $\ell(r, A_1, \ldots, A_k)$, and suppose vertices $v(P_i, q_i(r))$ and $v(P_j, q_j(r))$ are assigned to labels $\ell(P_i, q_i(r), A'_i)$ and $\ell(P_j, q_j(r), A'_j)$, respectively. As $(r, i, j) \in T$, the answers A'_i and A'_j of provers P_i and P_j cannot be weakly consistent. However, the answers A_i and A_j are strongly consistent. Therefore, either the (i, j) coordinates in A_i and A'_i differ (recall that this coordinate contains an assignment to a clause C_{ij}), or the (i, j) coordinates in A_j and A'_j differ (this coordinate contains an assignment to a distinguished variable x_{ij}). Assume the former is true (the other case is handled similarly).

Let e be the edge between $v(r)$ and $v(P_i, q_i(r))$. It is enough to show that the path $\mathcal{P}_e$ contains a label corresponding to prover P_j. Suppose this is not the case. Let $\ell(P_a, q_a, A)$ and $\ell(P_b, q_b, A')$ be two consecutive query labels on the path. As the two labels are at distance 1, there must be an $r' \in R$, such that $q_a = q_a(r')$ and $q_b = q_b(r')$, and the answers A and A' are strongly consistent. As $a, b \ne j$, the (i, j) coordinate in q_a and in q_b must contain some clause, and the two clauses are identical. Moreover, coordinate (i, j) of A and A' must contain an identical assignment to the variables of this clause. Therefore, if path $\mathcal{P}_e$ starts at $\ell(P_i, q_i(r), A'_i)$, and does not pass through any label belonging to prover P_j, then for every query label $\ell(P_s, q_s, A)$ appearing on the path, coordinate (i, j) of q_s contains the same clause as that of $q_i(r)$, and coordinates (i, j) in A and A'_i are identical. This is also true for the last query label on the path, denoted by $\ell(P_d, q_d, A_d)$. But this label is connected by an edge to label $\ell(r, A_1, \ldots, A_k)$, and therefore coordinates (i, j) of A_d and A_i must be identical, which is impossible. $\square$

It follows from the lemma that $|T'| \ge |T|$, yielding that the solution cost is at least $\binom{k}{2} \cdot \frac{\epsilon}{3} \cdot |R|$.

4.3.3 Construction Size

The size of the construction is dominated by the number of labels. For each i, $1 \le i \le k$, $|Q_i| \le (5n)^{k^2}$, and for each $q \in Q_i$, $|A(q)| \le 7^{k^2}$, and therefore the number of query labels is at most $k(5n)^{k^2} \cdot 7^{k^2}$. The size of R is at most $(5n)^{k^2}$ and for each $r \in R$ the number of k-tuples of consistent answers is at most 7^{k^2}. Hence, the number of constraint labels is bounded by $(5n)^{k^2} \cdot 7^{k^2}$. The construction size is therefore $N = n^{O(k^2)}$. If k is a constant, then it is polynomial in n. Choosing $k = \text{poly}(\log n)$, we get that $k = (\log N)^{\frac{1}{2} - \delta}$ for arbitrarily small constant δ.

Thus, we have proved the following result.

Theorem 4.6 *There is no constant approximation factor for the metric labeling problem, unless P=NP. Moreover, for any constant $\delta > 0$, there is no $\Omega((\log N)^{\frac{1}{2} - \delta})$-approximation for the problem, unless*

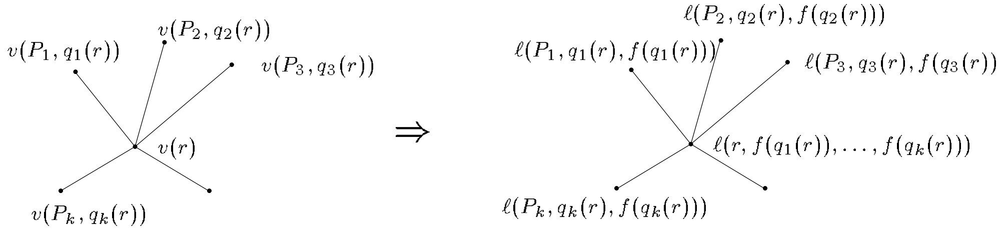

Figure 3. Yes instance: the embedding of edges incident to $v(r)$.

$NP \subseteq DTIME(n^{poly(\log n)})$.

Acknowledgements

The authors would like to thank Sanjeev Khanna for helpful comments on the presentation of the paper.

References

[1] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, 1998.

[2] Y. Bartal. Probabilistic approximation of metric spaces and its algorithmic applications, *Proceedings of the 37th IEEE Symposium on Foundations of Computer Science*, pp. 184–193 (1996).

[3] Y. Bartal. On approximating arbitrary metrics by tree metrics, *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pp. 161–168 (1998).

[4] G. Călinescu, H. Karloff, and Y. Rabani. An improved approximation algorithm for multiway cut. *Journal of Computer and System Sciences*, 60(3):564–574, 2000.

[5] G. Călinescu, H. Karloff, and Y. Rabani. Approximation algorithms for the 0-extension problem. In *Proceedings of the Twelfth Annual Symposium on Discrete Algorithms*, pp. 8–16, 2001.

[6] C. Chekuri, S. Khanna, J. Naor, and L. Zosin. Approximation algorithms for the metric labeling problem via a new linear programming formulation. In *Proceedings of the 12th Annual ACM/SIAM Symposium on Discrete Algorithms*, pp. 109–118, 2001.

[7] E. Dahlhaus, D. S. Johnson, C. H. Papadimitriou, P. D. Seymour, and M. Yannakakis. The complexity of multiterminal cuts. *SIAM Journal on Computing*, 23:864–894, 1994.

[8] J. Fakcharoenphol, S. Rao, and K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing*, pp. 448–455, 2003.

[9] U. Feige. A Threshold of ln n for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.

[10] D. Karger, P. Klein, C. Stein, M. Thorup, and N. Young. Rounding algorithms for a geometric embedding of minimum multiway cut. In *Proceedings of 31st Annual ACM Symposium on the Theory of Computing*, pages 668–678, 1999.

[11] A. Karzanov. Minimum 0-extension of graph metrics. *Europ. J. Combinat.*, 19:71–101, 1998.

[12] A. Karzanov. A combinatorial algorithm for the minimum $(2, r)$-metric problem and some generalizations. *Combinatorica*, 18(4):549–569, 1999

[13] J. Kleinberg and E. Tardos. Approximation algorithms for classification problems with pairwise relationships: metric labeling and markov random fields. *Journal of the ACM*, 49:616–630, 2002.

[14] R. Raz. A parallel repetition theorem. *SIAM Journal on Computing*, 27(3):763–803, 1998.