Pre-Reduction Graph Products: Hardnesses of Properly Learning DFAs and Approximating EDP on DAGs

Parinya Chalermsook
Max-Planck-Institut für Informatik
Saarbrücken, Germany

Bundit Laekhanukit
McGill University, Canada.
& Istituto Dalle Molle di Studi
sull'Intelligenza Artificiale (IDSIA)

Danupon Nanongkai
University of Vienna
Faculty of Computer Science
Vienna, Austria

***Abstract*—The study of graph products is a major research topic and typically concerns the term $f(G * H)$, e.g., to show that $f(G * H) = f(G)f(H)$. In this paper, we study graph products in a non-standard form $f(R[G * H]$ where R is a "reduction", a transformation of any graph into an instance of an intended optimization problem. We resolve some open problems as applications.**

The first problem is *minimum consistent deterministic finite automaton (DFA)*. We show a tight $n^{1-\epsilon}$-approximation hardness, improving the $n^{1/14-\epsilon}$ hardness of [Pitt and Warmuth, STOC 1989 and JACM 1993], where n is the sample size. (In fact, we also give improved hardnesses for the case of *acyclic* DFA and NFA.) Due to Board and Pitt [Theoretical Computer Science 1992], this implies the *hardness of properly learning DFAs* assuming $\mathsf{NP} \neq \mathsf{RP}$ (the weakest possible assumption). This affirmatively answers an open problem raised 25 years ago in the paper of Pitt and Warmuth and the survey of Pitt [All 1989]. Prior to our results, this hardness only follows from the stronger hardness of *improperly* learning DFAs, which requires stronger assumptions, i.e., either a cryptographic or an average case complexity assumption [Kearns and Valiant STOC 1989 and J. ACM 1994; Daniely et al. STOC 2014]. The second problem is *edge-disjoint paths* (EDP) on *directed acyclic graphs* (DAGs). This problem admits an $O(\sqrt{n})$-approximation algorithm [Chekuri, Khanna, and Shepherd, Theory of Computing 2006] and a matching $\Omega(\sqrt{n})$ integrality gap, but so far only an $n^{1/26-\epsilon}$ hardness factor is known [Chuzhoy et al., STOC 2007]. (n denotes the number of vertices.) Our techniques give a tight $n^{1/2-\epsilon}$ hardness for EDP on DAGs, thus resolving its approximability status.

As by-products of our techniques: (i) We give a tight hardness of packing vertex-disjoint k-cycles for large k, complimenting [Guruswami and Lee, ECCC 2014] and matching [Krivelevich et al., SODA 2005 and ACM Transactions on Algorithms 2007]. (ii) We give an alternative (and perhaps simpler) proof for the hardness of properly learning DNF, CNF and intersection of halfspaces [Alekhnovich et al., FOCS 2004 and J. Comput.Syst. Sci. 2008]. Our new concept reduces the task of proving hardnesses to merely analyzing graph product inequalities, which are often as simple as textbook exercises. This concept was inspired by, and can be viewed as a generalization of, the *graph product subadditivity* technique we previously introduced in SODA 2013. This more general concept might be useful in proving other hardness results as well.

***Keywords*-hardness of approximation; graph product;**

I. Introduction

A. The Concept of Pre-Reduction Graph Product

Background: Graph Product and Hardness of Approximation. Graph product is a fundamental tool with rich applications in both graph theory and theoretical computer science. It is, roughly speaking, a way to combine two graphs, say G and H, into a new graph denoted by $G * H$. For example, the following *lexicographic product*, denoted by $G \cdot H$, will be particularly useful in this paper.

$$\begin{aligned} V(G \cdot H) &= V(G) \times V(H) \\ &= \{(u, v) : u \in V(G) \ \underline{\text{and}}\ v \in V(H)\}. \\ E(G \cdot H) &= \{(u, a)(v, b) : uv \in E(G)\} \cup \\ &\qquad \{(u, a)(v, b) : u = v \wedge ab \in E(H)\}. \qquad (1) \end{aligned}$$

A common study of graph product aims at understanding how $f(G * H)$ behaves for some function f on graphs denoting a graph property. For example, if we let $\alpha(G)$ be the *independence number* of G (i.e., the cardinality of the maximum independent set), then $\alpha(G \cdot H) = \alpha(G)\alpha(H)$.

Graph products have been extremely useful in *boosting* the hardness of approximation. One textbook example is proving the hardness of n^ϵ for approximating the maximum independent set problem (i.e., approximating $\alpha(G)$ of an input graph G): Berman and Schnitger [1] showed that we can reduce from Max 2SAT to get a constant approximation hardness $c > 1$ for the maximum independent set problem, and then use a graph product to boost the resulting hardness to n^ϵ for some (small) constant ϵ. To illustrate how graph products amplify hardness, suppose we have a (1.001)-gap reduction $R[I]$ that transforms an instance I of SAT into a graph G. Since $\alpha(\cdot)$ is multiplicative, if we take a product $R[I]^k$ for any integer k, the hardness gap immediately

The full version of this paper is available at https://eprints.cs.univie.ac.at/4105/.

Work partially done while P. Chalermsook was at IDSIA, Switzerland and supported by the Swiss National Science Foundation project 200020_144491/1, and while D. Nanongkai was at Nanyang Technological University (NTU), Singapore, and ICERM, Brown University, USA. B. Laekhanukit was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) grant no. 28833, Dr&Mrs M.Leong fellowship, and ERC starting grant 279352 (NEWNET).

becomes $(1.0001)^k = 2^{\Omega(k)}$. Choosing k to be large enough gives $2^{\log^{1-\epsilon} n}$ hardness. Therefore, once we can rule out the PTAS, graph products can be used to boost the hardness to almost polynomial. This idea is also used in many other problems, e.g., in proving the hardness of the longest path problem [2].

Our Concept: Pre-Reduction Graph Product. This paper studies a reversed way to apply graph products: instead of the commonly used form of $(R[I])^k = (R[I] * R[I] * \ldots)$ to boost the hardness of approximation, we will use $R[I^k] = R[I * I * \ldots]$; here, I is a graph which is an instance of a hard graph problem such as maximum independent set or minimum coloring. We refer to this approach as *pre-reduction graph product* to contrast the previous approach in which graph product is performed *after* a reduction (which will be referred to as *post-reduction* graph product). The main conceptual contribution of this paper is the demonstration to the power and versatility of this approach in proving approximation hardnesses. We show our results in Section I-B and will come back to explain this concept in more detail in Section II.

We note one conceptual difference here between the previous post-reduction and our pre-reduction approaches: While the previous approach starts from a reduction R that already gives some hardness result, our approach usually starts from a reduction that does *not immediately* provide any hardness result; in other words, such reduction alone cannot be used to even prove NP-hardness. (See Section II for an illustration.) Moreover, in contrast to the previous use of $(R[I])^k$ which requires $R[I]$ to be a graph, our approach allows us to prove hardnesses of problems whose input instances are not graphs. Also note that our approach gives rise to a study of graph products in a new form: in contrast to the usual study of $f(G*H)$, our hardness results crucially rely on understanding the behavior of $f(R[G*H])$ for some function f, reduction R, and graph product $*$ (which happens to always be the lexicographic product in this paper). Another feature of this approach is that it usually leads to simple proofs that do not require heavy machineries (such as the PCP-based construction) – some of our hardness proofs are arguably simplifications of the previous ones; in fact, most of our hardness results follow from the meta-theorem (see Section III) which shows that a bounds of $f(R[G * H])$ in a certain form will immediately lead to hardness results. We list some bounds of $f(R[G * H])$ in Theorem II.1.

B. Problems and Our Results

1) Minimum Consistent DFA and Proper PAC-Learning DFAs: In the *minimum consistent deterministic finite automaton* (DFA) problem, denoted by MINCON(DFA, DFA), we are given two sets $\mathcal{P}$ and $\mathcal{N}$ of *positive* and *negative sample* strings in $\{0,1\}^*$. We let the *sample size*, denoted by n, be the total number of bits in all sample strings. Our goal is to construct a DFA M of *minimum size* that is *consistent* with all strings in $\mathcal{P} \cup \mathcal{N}$. That is, M accepts all positive strings $x \in \mathcal{P}$ and rejects all negative strings $y \in \mathcal{N}$.

This problem can be easily approximated within $O(n)$. Due to its connections to PAC-learning automata and grammars (e.g. [7], [8]), the problem has received a lot of attention from the late 70s to the early 90s. The NP-hardness of this problem was proved by Gold [9] and Angluin [10]. Li and Vazirani [11] later provided the first hardness of approximation result of $(9/8-\epsilon)$. This was greatly improved to $n^{1/14-\epsilon}$ by Pitt and Warmuth [3]. Our first result is a tight $n^{1-\epsilon}$ hardness for this problem, improving [3]. In fact, our hardness result holds even when we allow an algorithm to compare its result to the optimal *acyclic* DFA (ADFA), which is larger than the optimal DFA. This problem is called MINCON($ADFA$, DFA).

Theorem I.1. *Given a pair of positive and negative samples* $(\mathcal{P},\mathcal{N})$ *of size* n *where each sample has length* $O(\log n)$, *for any constant* $\epsilon > 0$, *it is NP-hard to distinguish between the following two cases of MinCon(ADFA,DFA):*

- YES-INSTANCE: *There is an ADFA of size* n^ϵ *consistent with* $(\mathcal{P},\mathcal{N})$.
- NO-INSTANCE: *Any DFA that is consistent with* $(\mathcal{P},\mathcal{N})$ *has size at least* $n^{1-\epsilon}$.

In particular, it is NP-hard to approximate the minimum consistent DFA problem to within a factor of $n^{1-\epsilon}$.

The main motivation of this problem is its connection to the notion of *properly PAC-learning* DFAs. It is one of the most basic problems in the area of proper PAC-learning [7], [8], [3]. Roughly speaking, the problem is to learn an unknown DFA M from given random samples, where a learner is asked to output (based on such random samples) a DFA M' that closely approximates M (see, e.g., [12] for details). The main question is whether DFA is properly PAC-learnable.

This question was the main motivation behind [3]; however, the $n^{1/14-\epsilon}$ hardness in [3] was not strong enough to prove this. Kearns and Valiant [13] showed that a proper PAC-learning of DFAs is not possible if we assume a cryptographic assumption stronger than $P \neq NP$. In fact, their result implies that even *improperly* PAC-learning DFAs (i.e., the output does not have to be a DFA) is impossible. Very recently, Daniely et al. [14] obtained a similar result by assuming a (fairly strong) average-case complexity assumption generalizing Feige's assumption [15].

The question whether the cryptographic assumption could be replaced by the $\mathsf{RP} \neq \mathsf{NP}$ assumption (which would be the weakest assumption possible) was asked 25 years ago in [8], [3]. In particular, the following is the first open problem in [8]: *(i) Can it be shown that DFAs are not properly PAC-learnable based only on the assumption that* $\mathsf{RP} \neq \mathsf{NP}$*? (ii)*

Problems	Upper Bounds	Prev. Hardness	New Hardness
MinCon(DFA, DFA)	$O(n)$	$n^{1/14-\epsilon}$ [3]	$n^{1-\epsilon}$
EDP on DAGs	$\tilde{O}(n^{1/2})$ [4]	$n^{1/26-\epsilon}$ [5]	$n^{1/2-\epsilon}$
k-cycle packing	$O(\min(k, n^{1/2}))$	$\Omega(k)$ [6]	$O(\min(k, n^{1/2-\epsilon}))$
MinCon(CNF, CNF), MinCon(DNF, DNF), MinCon(Halfspace,Halfspace)	$O(n)$	$n^{1-\epsilon}$	$n^{1-\epsilon}$ (Alternative proof)

Table I: Summary of our hardness results.

Stronger still, can the improper learnability result of [13] be strengthened by replacing the cryptographic assumptions with only the assumption that $\mathsf{RP} \neq \mathsf{NP}$*?*

Applebaum, Barak and Xiao [16] showed that proving lower bounds for improper learning using many standard ways of reductions from NP-hard problems will not work unless the polynomial hierarchy collapses, suggesting that an answer to the second question is likely to be negative. For the first question, some hardnesses of proper PAC-learning assuming $\mathsf{RP} \neq \mathsf{NP}$ were already known at the time (e.g. [17]) and there are many more recent results (see, e.g., [12] and references therein). Despite this, the basic problem of learning DFAs (originally asked in the above question) has remained open. Theorem I.1 together with a result of Board and Pitt [18] immediately resolve this problem.

Corollary I.2. *Unless* $\mathsf{NP} = \mathsf{RP}$*, the class of DFAs is not properly PAC-learnable.*

We also note an amusing connection between this type of result and Chomsky's "Poverty of the Stimulus Argument", as noted by Aaronson [19]: "Let's say I give you a list of n-bit strings, and I tell you that there's some nondeterministic finite automaton M, with much fewer than n states, such that each string was produced by following a path in M. Given that information, can you reconstruct M (probably and approximately)? It's been proven that if you can, then you can also break RSA!" Our Corollary I.2 implies that for the case of deterministic finite automaton, being able to reconstruct M will imply not only that one can break RSA but also solve, for instance, traveling salesman problem (TSP) probabilistically.

2) Edge-Disjoint Paths on DAGs: In the edge-disjoint paths problem (EDP) problem, we are given a graph $G = (V, E)$ (which could be directed or undirected) and k source-sink pairs $s_1t_1, s_2t_2, \ldots, s_kt_k$ (a pair can occur multiple times). The objective is to connect as many pairs as possible via edge-disjoint paths. Throughout, we let n and m be the number of vertices and edges in G, respectively. Approximating EDP has been extensively studied. It is one of the major challenges in the field of approximation algorithms. The problem has received significant attention from many groups of researchers, attacking the problem from many angles and considering a few variants and special cases (see, e.g., [20], [21], [22], [23], [24], [25], [26], [27] and references therein).

In directed graphs, EDP can be approximated within a factor of $O(\min(m^{1/2}, n^{2/3}))$ [28], [29], [30]. The $O(m^{1/2})$ factor is *tight* on sparse graphs since directed EDP is NP-hard to approximate within a factor of $n^{1/2-\epsilon}$, for any $\epsilon > 0$ [31]. In contrast to the directed case, undirected EDP is much less understood: The approximation factor for this case is $O(n^{1/2})$ [4] with a matching integrality gap of $\Omega(n^{1/2})$ for its natural LP relaxation, suggesting an $n^{1/2-\epsilon}$ hardness. Despite these facts, we only know a $\log^{1/2-\epsilon} n$ hardness of approximation assuming $\mathsf{NP} \not\subseteq \mathsf{ZPTIME}(n^{\mathrm{polylog}(n)})$. Even in special cases such as planar graphs (or, even simpler, brick-wall graphs, a very structured subclass of planar graphs), it is still open whether undirected EDP admits an $o(n^{1/2})$ approximation algorithm. This obscure state of the art made undirected EDP one of the most important, intriguing open problems in graph routing. (Table II summarizes the current status of EDP.)

One problem that may help in understanding undirected EDP is perhaps EDP on *directed acyclic graphs* (DAGs). This case is interesting because (i) its complexity seems to lie somewhere between the directed and undirected cases, (ii) it shares some similar statuses and structures with undirected EDP, and (iii) it has close connections to directed cycle packing [32] (i.e. hard instances for EDP on DAGs are used as a gadget in constructing the hard instance for directed cycle packing). In particular, on the upper bound side, the technique in [4] gives an $O(n^{1/2} \operatorname{poly}\log n)$ upper bound not only to undirected EDP but also to EDP on DAGs. Moreover, the integrality gap of $\Omega(n^{1/2})$ applies to both cases, suggesting a hardness of $n^{1/2-\epsilon}$ for them. However, previous hardness techniques for the case of general directed graphs [31] completely fail to give a lower bound on both DAGs and undirected graphs[1]. On the other hand, subsequent techniques that were invented in [33], [34] to deal with undirected EDP can be strengthened to prove the currently best hardness for DAGs [5][2], which is $n^{1/26-\epsilon}$. These results suggest that the complexity of DAGs lies between undirected and directed graphs. In this paper, we show that our techniques give a hardness of $n^{1/2-\epsilon}$ for this case, thus completely settling its approximability status. Our

[1]The result in [31] crucially relies on the fact that EDP with 2 terminal pairs is hard on directed graphs. This is not true if the graph is a DAG or undirected.

[2]Their result is in fact proved in a more general setting of EDP with congestion c for any $c \geq 1$

Cases	Upper Bounds	Integrality Gap	Prev. Hardness
Undirected	$O(n^{1/2})$ [4]	$\Omega(n^{1/2})$	$\log^{1/2-\epsilon} n$ [34]
DAGs	$\tilde{O}(n^{1/2})$ [4]	$\Omega(n^{1/2})$	$n^{1/26-\epsilon}$ [5]
Directed	$O(\min(m^{1/2}, n^{2/3}))$ [28], [29], [30]	$\Omega(n^{1/2})$	$n^{1/2-\epsilon}$ [31]

Table II: The current status of EDP.

result is formally stated in the following theorem.

Theorem I.3. *Given an instance of* EDP *on DAGs, consisting of a graph $G = (V, E)$ on n vertices and a source-sink pairs $(s_1, t_1), \dots, (s_k, t_k)$, for any $\epsilon > 0$, it is NP-hard to distinguish between the following two cases:*

- YES-INSTANCE: *There is a collection of edge disjoint paths in G that connects $1/n^{\epsilon}$ fraction of the source-sink pairs.*
- NO-INSTANCE: *Any collection of edge disjoint paths in G connects at most $1/n^{1/2-\epsilon}$ fraction of the source-sink pairs.*

In particular, it is NP-hard to approximate EDP *on DAGs to within a factor of $n^{1/2-\epsilon}$.*

3) Other Results: **Minimum Consistent NFA.** Our techniques also allow us to prove a hardness result for the *minimum consistent NFA* problem as stated formally in the following theorem.

Theorem I.4. *Given a pair of positive and negative samples $(\mathcal{P}, \mathcal{N})$ of size n where each sample has length $O(\log n)$, for any constant $\epsilon > 0$, it is NP-hard to distinguish between the following two cases of MinCon(ADFA,NFA):*

- YES-INSTANCE: *There is an NFA of size n^{ϵ} consistent with $(\mathcal{P}, \mathcal{N})$.*
- NO-INSTANCE: *Any NFA that is consistent with $(\mathcal{P}, \mathcal{N})$ has size at least $n^{1/2-\epsilon}$.*

In particular, it is NP-hard to approximate the minimum consistent NFA problem to within a factor of $n^{1/2-\epsilon}$.

This improves upon the $n^{1/14-\epsilon}$ hardness of Pitt and Warmuth [3]. We note that this hardness result is not strong enough to imply a PAC-learning lower bound for NFAs. Such hardness was already known based on some cryptographic or average-case complexity assumptions [13], [14]. We think it is an interesting open problem to remove these assumptions as we did for the case of learning DFAs.

k-Cycle Packing. Our reduction for EDP can be slightly modified to obtain hardness results for k-Cycle Packing, when k is large. In the *k-cycle packing* problem, given an input graph G, one wants to pack as many disjoint cycles as possible into the graph while we are only interested in cycles of length at most k. An $O(\min(k, n^{1/2}))$-approximation algorithm for this problem can be easily obtained by modifying the algorithm of Krivelevich et al. [32]). Very recently, Guruswami and Lee [6] obtained a hardness of $\Omega(k)$, assuming the Unique Game Conjecture, when k is a constant. This matches the upper bound of Krivelevich et al. for small k. In this paper, we compliment the result of Guruswami and Lee by showing a hardness of $n^{1/2-\epsilon}$ for some $k \geq n^{1/2}$, matching the upper bound of Krivelevich et al. for the case of large k.

Theorem I.5. *Given a directed graph G, for any $\epsilon > 0$ and some $k \geq |V(G)|^{1/2}$, it is NP-hard to distinguish between the following cases:*

- *There are at least $|V(G)|^{1/2-\epsilon}$ disjoint cycles of length k in G.*
- *There are at most $|V(G)|^{\epsilon}$ disjoint cycles of length at most $2k - 1$ in G.*

In particular, for some $k \geq n^{1/2}$, the k-cycle packing problem on n-vertex graphs is hard to approximate to within a factor of $n^{1/2-\epsilon}$.

Alternative Hardness Proof for Minimum Consistent CNF, DNF, and Intersections of Halfspaces. Our techniques for proving the DFA hardness result can be used to give an alternative proof for the hardness of the minimum consistent DNF, CNF, and intersections of thresholded halfspaces problems. In the minimum consistent CNF problem, we are given a collection of samples of size n, and our goal is to output a small CNF formula that is consistent with all such samples. Alekhnovich et al. [35] previously showed tight hardnesses for these problems, which imply that the classes of CNFs, DNFs, and the intersections of halfspaces are not properly PAC-learnable. Our techniques give an alternative proof (which might be simpler) for these results. More specifically, we give an alternative proof for the following theorem and corollary (stated in terms of CNF, but the same holds for DNF and intersection of halfspaces[3]).

Theorem I.6. *Let $\epsilon > 0$ be any constant. Given a pair of positive an negative samples $(\mathcal{P}, \mathcal{N})$ of size n where each sample has length at most n^{ϵ}, it is NP-hard to distinguish between the following two cases:*

- YES-INSTANCE*: There is a CNF formula of size n^{ϵ} consistent with $(\mathcal{P}, \mathcal{N})$.*
- NO-INSTANCE*: Any CNF consistent with $(\mathcal{P}, \mathcal{N})$ must have size at least $n^{1-\epsilon}$.*

In particular, it is NP-hard to approximate the minimum consistent CNF problem to within a factor of $n^{1-\epsilon}$.

[3]It is noted in [35] that one only needs to prove the hardness of CNF, since this problem is a special case of the intersection of thresholded halfspaces problem, and the proof for DNF would work similarly.

Corollary I.7. *Unless* $\mathsf{NP} = \mathsf{RP}$*, the class of CNF is not properly PAC-learnable.*

II. Overview

A. Example of Reduction R: Vertex-Disjoint Paths

To illustrate the pre-reduction graph product concept, consider the *vertex-disjoint path* (VDP) problem. The objective of VDP is the same as that of EDP except that we want paths to be vertex-disjoint instead of edge-disjoint. The approximability statuses of EDP and VDP on DAGs and undirected graphs are the same, and we choose to present VDP due to its simpler gadget construction. Our hardness of VDP can be easily turned into a hardness of EDP.

Our goal is to show that this problem has an approximation hardness of $n^{1/2-\epsilon}$, where n is the number of vertices. We will use the following reduction[4] R which transforms a graph G (supposedly an input instance of the maximum independent set problem) into an instance $R[G]$ of the vertex-disjoint paths problem with $\Theta(|V(G)|^2)$ vertices. We start with an instance $R[G]$ as in Figure 1a where there are k source-sink pairs (Figure 1a shows an example where $k = 6$) and edges are oriented from left to right and from top to bottom. Let us name vertices in G by 1, 2, …, k. For any pair of vertices i and j, where $i < j$, such that edge ij does *not* present in G, we remove a vertex v_{ij} from $R[G]$, as shown in Figure 1b (this means that two edges that point to v_{ij} will continue on their directions without intersecting each other).

To see an intuition of this reduction, define a *canonical path* be a path that starts at some source s_i, goes all the way right, and then goes all the way down to t_i (e.g., a thick (green) path in Figure 1b). It can be easily seen that any set of vertex-disjoint paths in $R[G]$ that consists only of canonical paths can be converted to a solution for the maximum independent set problem. Conversely any independent set S in G can be converted to a set of $|S|$ vertex-disjoint paths. For example, canonical paths between the pairs (s_1, t_1) and (s_2, t_2) in $R[G]$ in Figure 1b can be converted to an independent set $\{1, 2\}$ in G and vice versa. In other words, if we can *force* the VDP solution to consist only of canonical paths, then we can potentially use the $|V|^{1-\epsilon}$ hardness of maximum independent set to prove a tight $|V|^{1-\epsilon} = |V(R[G])|^{1/2-\epsilon}$ hardness of VDP. This intuition, however, cannot be easily turned into a hardness result since the VDP solution can use non-canonical paths, and it is possible that $\mathsf{VDP}(R[G])$ is much larger than $\alpha(G)$. Thus, the reduction R by itself cannot be used even to prove that VDP is NP-hard!

B. The Use of Pre-Reduction Products

The above situation is very common in attempts to prove hardnesses for various problems. A usual way to obtain hardness results is to modify R into some reduction R'. This modification, however, often blows up the size of the reduction, thus affecting its tightness. For example, VDP and EDP on DAGs are only known to be $n^{1/26-\epsilon}$-hard, as opposed to being potentially $n^{1/2-\epsilon}$-hard, as suggested by the integrality gap. Moreover, the reduction R' is usually much more complicated than R. In this paper, we show that for many problems the above difficulties can be avoided by simply picking an appropriate graph product $*$ and understanding the structure of $R[G * G * \ldots]$. To this end, it is sometimes easier to study $f(R[G * H])$ for any graphs G and H, although we eventually need only the case where $G = H$. This gives rise to the study of the behavior of $f(R[G*H])$ which is a non-standard form of graph product in comparison with the standard study of $f(G * H)$. In fact, most results in this paper follow merely from bounding $f(R[G * H])$ in the form

$$g(G * H) \leq f(R[G * H]) \leq g(G)f(H) + \mathrm{poly}(|V(G)|), \tag{2}$$

where g is an objective function of a problem whose hardness is already known (in this paper, g is either maximum independent set or minimum coloring), and f is an objective function of a problem that we intend to prove hardness. Our bounds for functions f corresponding to problems that we want to solve, e.g. the minimum consistent DFA (function dfa) and maximum edge-disjoint paths (function edp), are listed in the theorem below. (Recall that $G \cdot H$ denotes the lexicographic product as defined in Equation (1).)

Theorem II.1 (Bounds of graph products; informal)**.** *There is a reduction R_1 (respectively R_2) that transforms a graph G into an instance of the minimum consistent DFA problem of size $\tilde{\Theta}(|V(G)|^2)$ (respectively the maximum edge-disjoint paths problem of size $\Theta(|V(G)|^2)$) such that, for any graphs G and H,*

$$\chi(G \cdot H) \leq \mathsf{dfa}(R_1[G \cdot H]) \leq \chi(G)\mathsf{dfa}(R_1[H]) + O(|V(G)|^2) \tag{3}$$

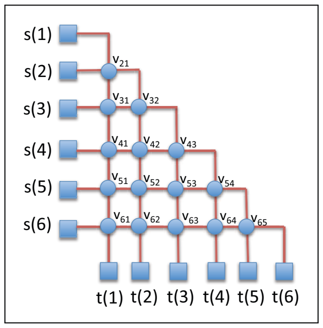
$$\alpha(G \cdot H) \leq \mathsf{edp}(R_2[G \cdot H]) \leq \alpha(G)\mathsf{edp}(R_2[H]) + O(|V(G)|^2) \tag{4}$$

It only requires a systematic, simple calculation to show that these inequalities imply hardnesses of approximation; we formulate this implication as a "meta theorem" (see Section III) which roughly states that for large enough k,

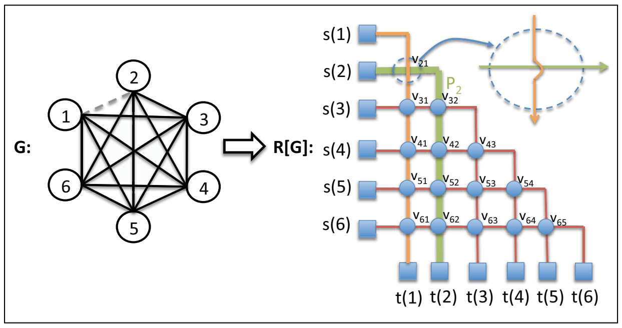
$$f(R[G^k]) \approx g(G^k) \tag{5}$$

where G^k is $G * G * G * \ldots$ (k times). (For an intuition, observe that when k is large enough, the term $\mathrm{poly}(|V(G)|)$ in Equation (2) will be negligible and an inductive argument can be used to show that $g(G^k) \leq f(R[G^k]) \leq g(G)^{O(k)}$ (recall that, in our case, g is multiplicative)). This means that

[4]We thank Julia Chuzhoy who suggested this reduction to us (private communication).


s(1)
s(2)
s(3)
s(4)
s(5)
s(6)
v_{21}
v_{31} v_{32}
v_{41} v_{42} v_{43}
v_{51} v_{52} v_{53} v_{54}
v_{61} v_{62} v_{63} v_{64} v_{65}
t(1) t(2) t(3) t(4) t(5) t(6)

(a)


G:
1
2
3
4
5
6
R[G]:
s(1)
s(2)
s(3)
s(4)
s(5)
s(6)
P_2
v_{21}
v_{31} v_{32}
v_{41} v_{42} v_{43}
v_{51} v_{52} v_{53} v_{54}
v_{61} v_{62} v_{63} v_{64} v_{65}
t(1) t(2) t(3) t(4) t(5) t(6)

(b)

Figure 1: The reduction R for the vertex-disjoint paths problem. The thick (green) path in Figure 1b shows an example of a canonical path.

the hardness of f[5] is at least the same as the hardness of g on graph product instances G^k. For the case of DFA and EDP, $R_1(G)$ and $R_2(G)$ increase the size of input size to $|V(G)|^2$ while α and χ have the hardness of $|V(G)|^{1-\epsilon}$. Thus, we get a hardness of $n^{1/2-\epsilon}$ where n is the input size of DFA and EDP. This immediately implies a tight hardness for EDP and an improved hardness of DFA. How this translates to a hardness of f depends on how much instance blowup the reduction $R[G^k]$ causes. For our problems of DFA and EDP, it is a well known result that the hardness of α and χ stays roughly the same under the lexicographic product, i.e., α and χ on G^k have a hardness of $|V(G^k)|^{1-\epsilon}$. The meta theorem and Theorem II.1 say that this hardness also holds for DFA and EDP. Since $R_1[G^k]$ and $R_2[G^k]$ increase the size of input instances by a quadratic factor — from $|V(G^k)|$ to $n = |V(G^k)|^2$ — we get a hardness of $n^{1/2-\epsilon}$ where n is the input size of DFA and EDP. This immediately implies a tight hardness for EDP and an improved hardness for DFA.

C. Toward A Tight Hardness of the Minimum Consistent DFA Problem

To get the tight $n^{1-\epsilon}$ hardness for DFA, we have to adjust R_1 in Theorem II.1 to avoid the quadratic blowup. We will exploit the fact that, to get a result similar to Equation (5), we only need a reduction R defined on the k-fold graph product G^k instead of on an arbitrary graph G as in the case of R_1. We modify reduction R_1 to $R_{1,k}$ that works only on an input graph in the form G^k and produces an instance $R_{1,k}[G^k]$ of size almost linear in $|V(G^k)|$ while inequalities as in Theorem II.1 still hold, and obtain the following.

Lemma II.2. *For any k, there is a reduction $R_{1,k}$ that reduces a graph $G^k = G{\cdot}G{\cdot}\ldots$ into an instance of the minimum consistent DFA problem of size $O(k{\cdot}|V(G^k)|{\cdot}|V(G)|^2)$ such that*

$$\chi(G)^k \leq \mathsf{dfa}(R_{1,k}[G^k]) \leq \chi(G)^{2k}|V(G)|^4 \qquad (6)$$

Observe that the size $O(k \cdot |V(G^k)| \cdot |V(G)|^2)$ of $R_{1,k}(G^k)$ is almost linear (almost $O(|V(G^k)|)$) as the extra $O(k|V(G)|^2)$ is negligible when k is sufficiently large. Similarly, the term $|V(G)|^4$ in Equation (6) is negligible and thus the value of $\mathsf{dfa}(R_{1,k}[G^k])$ is sandwiched by $\chi(G)^k$ and $\chi(G)^{2k}$. This means that if $\chi(G)$ is small (i.e., $\chi(G) \leq |V(G)|^{\epsilon}$), then $\mathsf{dfa}(R_{1,k}[G^k])$ will be small (i.e., $\mathsf{dfa}(R_{1,k}[G^k]) \leq |V(G^k)|^{2\epsilon}$), and if $\chi(G)$ is large (i.e., $\chi(G) \geq |V(G)|^{1-\epsilon}$), then $\mathsf{dfa}(R_{1,k}[G^k])$ will be also large (i.e., $\mathsf{dfa}(R_{1,k}[G^k]) \geq |V(G^k)|^{1-\epsilon}$). The hardness of $n^{1-\epsilon}$ for DFA thus follows.

We note that in Theorem II.1, we can replace DFA by NFA, a function corresponds to the minimum consistent NFA problem, thus getting a hardness of $n^{1/2-\epsilon}$ for this problem as well. This is, however, not yet tight. We would get a tight hardness if we can replace DFA by NFA in Lemma II.2, which is not the case. We also note that the proof for the tight hardness for the minimum consistent CNF problem follows from the same type of inequalities: We show that there exists a near-linear-size reduction $R_{3,k}$ from the minimum coloring problem to the minimum consistent CNF problem (with function cnf) such that

$$\chi(G)^k \leq \mathsf{cnf}(R_{3,k}[G^k]) \leq \chi(G)^k|V(G)|^{O(1)}. \qquad (7)$$

The proofs of the bounds of graph products (Equations (3), (4), (6) and (7)) are fairly short and elementary; in fact, we believe that they can be given as textbook exercises. These proofs can be found in the full version.

D. Related Concept

Our pre-reduction graph product concept was inspired by the *graph product subadditivity* concept we previously introduced in [36] (some of these ideas were later used in [37], [38]). There, we prove a hardness of approximation using the following framework. As before, let f be an

[5]For conciseness, we will use g and f to refer to problems and their objective functions interchangeably.

objective function of a problem that we intend to prove hardness and g be an objective function of a problem whose hardness is already known. We show that there are graph products $\oplus$, $*_e$, and $*$ such that

- We can "decompose" $f(G *_e J)$: $g(G) \leq f(G *_e J) \leq g(G) + f(G * J)$, and
- $f((G \oplus H) * J)$ is "subadditive": $f((G \oplus H) * J) \leq f(G * J) + f(H * J)$.

We then use the above inequalities to show that if we let $G^k = G \oplus G \oplus \ldots$ (k times), then

$$g(G^k) \leq f(G^k * J) \leq g(G^k) + kf(G * J).$$

For large enough k, the term $kf(G * J)$ is negligible and thus $f(G^k *_e J) \approx g(G^k)$. We use this fact to show that the approximation harness of f is roughly the same as the hardness of g. Observe that if we let $R[G] = G *_e J$, the above inequalities can then be used to show that

$$g(G\oplus H) \leq f(R[G\oplus H]) \leq g(G\oplus H)+f(R[G])+f(R[H]).$$

In the problems considered in [36], one can easily bound $f(R[G])$ and $f(R[H])$ by $|V(G)|$ and $|V(H)|$, respectively. So, our meta theorem will imply that $f(G^k * J) \approx g(G^k)$, which leads to the approximation hardness of f. This means that the previous concept in [36] can be viewed as a special case of our new concept where we restrict the reduction R to be a graph product $R[G] = G *_e J$. The way we use the reduction R in this paper goes beyond this. For example, our reduction R_2 for EDP as illustrated in Figure 1 cannot be viewed as a natural graph product. Moreover, our reduction R_1 reduces a graph G to an instance of DFA which has *nothing* to do with graphs. (This is possible only when we abandon viewing reduction R as a graph product.) Our meta theorem also shows that bounds of graph products in a much more general form can imply hardness results. Finally, the way we exploit graph products using the reduction $R_{1,k}$ has never appeared in [36].

E. Organization

We prove our meta theorems in Section III. These theorems show that bounding $f(R[G * H])$ in a certain way will immediately imply a hardness result. They allow us to focus on proving all results that have been developed in this framework. We show here how to prove the graph product bound for automata problems.

III. Meta Theorems

In this section, we prove general theorems that will be used in proving most hardness results in this paper. These theorems give *abstractions* of the (graph product) properties one needs to prove in order to obtain hardness of approximation results. Our techniques can be used to derive hardnesses for both minimization and maximization problems. For the former, the reduction is from minimum coloring, while the latter is obtained via a reduction from maximum independent set.

Let us start with maximization problems. Suppose we have an optimization problem Π such that any instance $I \in \Pi$ is associated with an optimal function $\mathsf{OPT}_\Pi(I)$. We consider a transformation R that maps any graph G into an instance $R[G]$ of the problem Π. We say that a transformation R satisfies a *low α-projection property* with respect to a *maximization problem* Π if and only if the following two conditions hold:

- (I) For any graph $G = (V, E)$, $\mathsf{OPT}_\Pi(R[G]) \geq \alpha(G)$.
- (II) There are universal constants $c_1, c_2 > 0$ (independent of the choices of graphs) such that, for any two graphs G and H,
$$\mathsf{OPT}_\Pi(R[G \cdot H]) \leq |V(G)|^{c_1} + \alpha(G)^{c_2}\mathsf{OPT}_\Pi(R[H]).$$
- (III) There is a universal constant $c_0 > 0$ such that
$$\mathsf{OPT}_\Pi(R[G]) \leq c_0|R[G]|.$$

Intuitively, the transformation R with the low α-projection property tells us that there are relationships between the optimal solution of the problem Π on $R[G]$ and the independence number of G. Instead of looking for a sophisticated construction of R, we focus on a "simple" transformation R that establishes a connection on one side, i.e., $\mathsf{OPT}_\Pi(R[G]) \geq \alpha(G)$, and the "growth" of OPT_Π is "slow" with respect to graph products. Property (III) of the low α-projection property says that the optimal is at most linear in the size of the instance, which is the case for almost every natural combinatorial optimization problem.

Next, we turn our focus to a minimization problem. In this case, we relate the optimal solution to the chromatic number of an input graph. Specifically, one can define the *low χ-projection property* with respect to a *minimization problem* Π as follows.

- (I) For any graph $G = (V, E)$, $\mathsf{OPT}_\Pi(R[G]) \geq \chi(G)$.
- (II) There are universal constants $c_1, c_2 > 0$ (independent of the choices of graphs) such that, for any two graphs G and H, we have
$$\mathsf{OPT}_\Pi(R[G \cdot H]) \leq |V(G)|^{c_1} + \chi(G)^{c_2}\mathsf{OPT}_\Pi(R[H]).$$
- (III) There is a universal constant $c_0 > 0$ such that
$$\mathsf{OPT}_\Pi(R[G]) \leq c_0|R[G]|.$$

We observe that the existence of such reductions is sufficient for establishing hardness of approximation results, and the hardness factors achievable from the theorems depend on the size of the reduction.

Theorem III.1 (Meta-Theorem for Maximization Problems)**.** *Let Π be a maximization problem for which there is a reduction R for Π that satisfies low α-projection property with $|R[G]| = O(|V(G)|^d)$. Then for any $\epsilon > 0$, given an* *instance I of Π, it is NP-hard to distinguish between the following two cases:*

- *(*YES-INSTANCE*:)* $\mathsf{OPT}_\Pi(I) \geq |I|^{1/d-\epsilon}$
- *(*NO-INSTANCE*:)* $\mathsf{OPT}_\Pi(I) \leq |I|^{\epsilon}$

Theorem III.2 (Meta-Theorem for Minimization Problems)**.** *Let Π be a minimization problem for which there is a reduction R for Π that satisfies low χ-projection property with $|R[G]| = O(|V(G)|^d)$, for some constant $d \geq 0$. Then for any $\epsilon > 0$, given an instance I of Π, it is NP-hard to distinguish between the following two cases:*

- *(*YES-INSTANCE*:)* $\mathsf{OPT}_\Pi(I) \leq |I|^{\epsilon}$
- *(*NO-INSTANCE*:)* $\mathsf{OPT}_\Pi(I) \geq |I|^{1/d-\epsilon}$

IV. A Graph Product Bound for Automata

We further illustrate the flavor of our approaches by proving bounds for automata problems (remark that, in contrast to EDP, this is not a "graph problem"). Note that the result here is not tight. The full proof (which implies PAC learning lower bound) is deferred to the full version.

A. Definitions

Given two graph G and H, the *lexicographic product* of G and H, denoted by $G \cdot H$, is defined as $V(G \cdot H) = V(G) \times V(H) = \{(u,v) : u \in V(G) \ \underline{\text{and}}\ v \in V(H)\}$, and $E(G \cdot H) = \{(u,a)(v,b) : uv \in E(G) \ \underline{\text{or}}\ (u = v \ \underline{\text{and}}\ ab \in E(H)\ )\}$.

A *deterministic finite automaton* (DFA) is defined as a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where Q is the set of states, Σ is the set of alphabets, $\delta : Q \times \Sigma \to Q$ is a transition function, q_0 is initial state, and $F \subseteq Q$ is the set of accepting states. One can naturally extend the transition function δ into $\delta^* : Q \times \Sigma^* \to Q$ by inductively defining $\delta^*(q, x_1, \ldots, x_\ell)$ as $\delta^*(\delta(q, x_1), x_2, \ldots, x_\ell)$ and $\delta^*(q, null) = q$. We say that M *accepts* x if and only if $\delta^*(q_0, x) \in F$.

In the MINIMUM CONSISTENCY problem, denoted by MINCON($\mathcal{H}$, $\mathcal{F}$), we are given collections $\mathcal{P}$ and $\mathcal{N}$ of positive and negative sample strings in $\{0,1\}^*$, for which we are guaranteed that there is a hypothesis $h \in \mathcal{H}$ that is consistent with all samples in $\mathcal{P} \cup \mathcal{N}$, i.e., $h(x) = 1$ for all $x \in \mathcal{P}$ and $h(x) = 0$ for all $x \in \mathcal{N}$. Our goal is to output a function $f \in \mathcal{F}$ that is consistent with all these samples, while minimizing $|f|$. In other words, $\mathcal{H}$ and $\mathcal{F}$ are the classes of the real hypothesis that we want to learn and those that our algorithm outputs respectively. This notion of learning allows our algorithm to output the hypothesis that is outside of the hypothesis class we want to learn.

Now we need a slightly modified notion of approximation factor. For any instance $(\mathcal{P}, \mathcal{N})$, we denote by $\mathsf{OPT}_\mathcal{H}(\mathcal{P}, \mathcal{N})$ the size of the smallest hypothesis $h \in \mathcal{H}$ consistent with $(\mathcal{P}, \mathcal{N})$. Let $\mathcal{A}$ be any algorithm for MINCON($\mathcal{H}$, $\mathcal{F}$), i.e., $\mathcal{A}$ always outputs the hypothesis in $\mathcal{F}$. The approximation gauranteed provided by $\mathcal{A}$ is:

$$\sup_{\mathcal{P},\mathcal{N}} \frac{|\mathcal{A}(\mathcal{P}, \mathcal{N})|}{\mathsf{OPT}_\mathcal{H}(\mathcal{P}, \mathcal{N})}$$

With this terminology, the problem of learning DFA can be abbreviated as MINCON(DFA, DFA).

B. The Reduction

We design a reduction $R[G]$ with χ-projection property and $|R[G]| = O(|V(G)|^2)$. By our meta theorem, this implies $N^{1/2-\epsilon}$ hardness of MINCON(DFA, NFA).

We will be working with binary strings, i.e., the alphabet set $\Sigma = \{0, 1\}$. Given a graph $G = (V, E)$, we construct two sets $\mathcal{P}, \mathcal{N}$ of positive and negative samples, which encode vertices and edges of the graph. We assume w.l.o.g. that $|V(G)| = 2^k$ for some integer k. Therefore, each vertex $u \in V(G)$ can be associated with a k-bit string $\langle u \rangle \in \{0,1\}^k$.

Now our reduction $R[G]$ is defined as follows. The positive samples are given by $\mathcal{P} = \left\{\langle u \rangle 1 \langle u \rangle^R : u \in V(G)\right\}$ and the negative samples are $\mathcal{N} = \left\{\langle u \rangle 1 \langle v \rangle^R : uv \in E(G)\right\}$. We denote this instance of the consistency problem by an ordered pair $(\mathcal{P}, \mathcal{N})$. Now we proceed to prove property (I), that any NFA consistent with $(\mathcal{P}, \mathcal{N})$ must have at least $\chi(G)$ states.

Lemma IV.1. *Let $M = (Q, \Sigma, \delta, q_0, F)$ be an NFA that is consistent with $(\mathcal{P}, \mathcal{N})$. Then for any vertex $u \in V(G)$,*

$$\delta^*(q_0, \langle u \rangle) \not\subseteq \bigcup_{v : uv \in E(G)} \delta^*(q_0, \langle v \rangle).$$

Corollary IV.2. *Any NFA M that is consistent with $(\mathcal{P}, \mathcal{N})_G$ must have at least $\chi(G)$ states. Therefore, $\mathsf{OPT}_{DFA}(R[G]) \geq \mathsf{OPT}_{NFA}(R[G]) \geq \chi(G)$ for all G.*

C. χ-Projection Property

We will consider a specific class of DFA $M = (Q, \Sigma, \delta, q_0, F)$, which we call *canonical DFA*. Specifically, we say that a DFA is *canonical* if it has the following properties.

- The state diagram has exactly ℓ layers for some ℓ, and each path from q_0 to any sink has length exactly ℓ.
- All accepting states are in the last layer.

Denote shortly by $\mathsf{OPT}(R[G])$ the number of states in the minimum canonical DFA consistent with $R[G]$. So we have that $\mathsf{OPT}(R[G]) \geq \mathsf{OPT}_{DFA}(R[G]) \geq \mathsf{OPT}_{NFA}(R[G])$. The following lemma gives the χ-projection property for $\mathsf{OPT}(\cdot)$

Lemma IV.3. $\mathsf{OPT}(R[G \cdot H]) \leq \chi(G)(\mathsf{OPT}(R[H]) + O(|V(G)|))$

Now we prove the lemma. Let $M_H = (Q_H, \{0,1\}, \delta_H, q_H, F_H)$ be the minimum DFA for the instance $R[H]$ whose number of states is $s = \mathsf{OPT}(R[H])$ and has $\ell_H = 2h + 1$ layers for $h = \lceil \log |V(H)| \rceil$. Let $C_1, \ldots, C_B$ be the color classes of G defined by the optimal coloring, so $B = \chi(G)$. Let $f : V(G) \to [B]$ be the corresponding coloring function. We will also be using several copies of a directed complete binary tree with 2^k

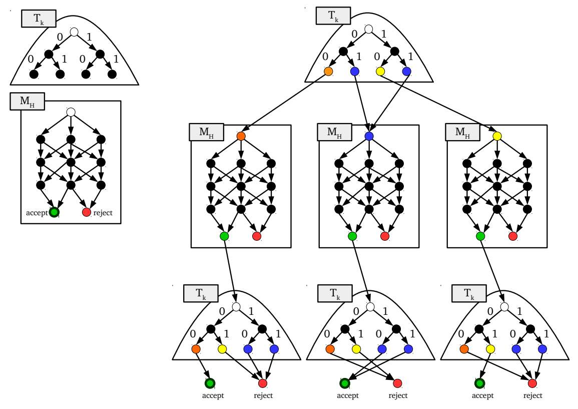


Figure 2: The illustration of the construction in the proof of Lemma IV.3.

leaves, where each leaf corresponds to a string in $\{0,1\}^k$ and is associated with a vertex in $V(G)$. Call this directed binary tree T_k.

We will use M_H and T_k to construct a new acyclic DFA M that have at most $B(s + O(|V(G)|))$ states and exactly $\ell = 2(k+\ell_H)+1$ layers. Now we proceed with the description of machine $M = (Q, \{0,1\}, \delta, q, F)$. We start by taking a copy of directed tree T_k, and call this copy $T_k^{(0)}$. The starting state q is defined to be the root of $T_k^{(0)}$. This is the *first phase* of the construction. Notice that there are k layers in the first phase, so exactly k positions of any input string will be read after this phase. Each state in the last layer is indexed by $state(\langle v\rangle)$ for each $v \in V(G)$.

In the *second phase*, we take B copies of the machines M_H where the j^{th} copy, denoted by $M_H^{(j)} = (Q_H^{(j)}, \{0,1\}, \delta_H^{(j)}, q_H^{(j)}, F_H^{(j)})$, is associated with color class C_j defined earlier. For each vertex $v \in V(G)$, we connect the corresponding state $state(\langle v\rangle)$ in the last layer of Phase 1 to the starting state $q_H^{(f(v))}$. This transition can be thought of as a "null" transition which can be removed afterward, but keeping it this way would make the analysis simpler. Since each copy of M_H has $2\ell_H+1$ layers, now our construction has exactly $2\ell_H + k + 1$ layers.

In the final phase, we first extend all rejecting states in $M_H^{(j)}$ by a unified path until it reaches layer $2(\ell_H + k) + 1$. This is a rejecting state rej_0. Now, for each $j = 1, \ldots, B$, we connect each accepting state in the last layer of $M_H^{(j)}$ to the root in the copy $T_k^{(j)}$ again by a "null" transition, so we reach the desired number of layers now (notice that each root-to-leaf path has $2(k+\ell_H)+1$ states.) The states in the last layer of $T_k^{(j)}$ are indexed by $state(j, \langle v\rangle)$. The accepting states of M are defined as $F = \bigcup_{j=1}^{B} \{state(j, \langle u\rangle^R) : u \in C_j\}$, and the rest of the states are defined as rejecting. This completes our construction. See Figure 2 for illustration.

The size of the construction is $|V(G)| + Bs + O(|V(G)|B) = B(s + O(|V(G)|))$. The next claim shows that the machine M is consistent with samples obtained from the product of G and H, which thus finish the proof.

Claim IV.4. *If M_H that is consistent with $R[H]$, then machine M constructed above is consistent with $R[G \cdot H]$.*

V. Conclusion and Open Problems

There are many open problems on edge-disjoint paths. Most notably can one narrow down the gap of undirected EDP between $O(\sqrt{n})$ upper bound and $\log^{1/2-\epsilon} n$ lower bound? For directed EDP, there is still a gap in the case of routing with congestion, between the upper bound of $n^{1/c}$ [39] and the lower bound of $n^{1/(3c+23)}$ [5] if we allow routing with congestion c. We believe that our techniques are likely to work there. Closing this gap would resolve an open question in Chuzhoy et al. [5].

Another interesting problem is the cycle packing problem. For this problem, the approximability is pretty much settled on undirected graphs with an upper bound of $O(\log^{1/2} n)$ and a lower bound of $\log^{1/2-\epsilon} n$ ([40], [32]). On directed graphs, there is still a gap between $n^{1/2}$ and $\Omega(\frac{\log n}{\log\log n})$.

Acknowledgement: We thank Julia Chuzhoy for suggesting the EDP reduction and for related discussions when the first author was still at the University of Chicago.

References

[1] P. Berman and G. Schnitger, "On the complexity of approximating the independent set problem," *Inf. Comput.*, vol. 96, no. 1, pp. 77–94, 1992, also, in STACS'89. 1

[2] D. R. Karger, R. Motwani, and G. D. S. Ramkumar, "On approximating the longest path in a graph," *Algorithmica*, vol. 18, no. 1, pp. 82–98, 1997. 2

[3] L. Pitt and M. K. Warmuth, "The minimum consistent DFA problem cannot be approximated within any polynomial," *J. ACM*, vol. 40, no. 1, pp. 95–142, 1993, announced at STOC 1989. 2, 3, 4

[4] C. Chekuri, S. Khanna, and F. B. Shepherd, "An $O(\sqrt{n})$ approximation and integrality gap for disjoint paths and unsplittable flow," *Theory of Computing*, vol. 2, no. 1, pp. 137–146, 2006. 3, 4

[5] J. Chuzhoy, V. Guruswami, S. Khanna, and K. Talwar, "Hardness of routing with congestion in directed graphs," in *STOC*, 2007, pp. 165–178. 3, 4, 9

[6] V. Guruswami and E. Lee, "Inapproximability of feedback vertex set for bounded length cycles," *Electronic Colloquium on Computational Complexity (ECCC)*, vol. 21, p. 6, 2014. 3, 4

[7] C. De la Higuera, *Grammatical inference: learning automata and grammars.* Cambridge University Press, 2010. 2

[8] L. Pitt, *Inductive inference, DFAs, and computational complexity.* Springer, 1989. 2

[9] E. M. Gold, "Complexity of automaton identification from given data," *Information and Control*, vol. 37, no. 3, pp. 302–320, 1978. 2

[10] D. Angluin, "On the complexity of minimum inference of regular sets," *Information and Control*, vol. 39, no. 3, pp. 337–350, 1978. 2

[11] M. Li and U. V. Vazirani, "On the learnability of finite automata," in *COLT*, 1988, pp. 359–370. 2

[12] V. Feldman, "Hardness of proper learning," in *Encyclopedia of Algorithms*. Springer, 2008. 2, 3

[13] M. J. Kearns and L. G. Valiant, "Cryptographic limitations on learning boolean formulae and finite automata," *J. ACM*, vol. 41, no. 1, pp. 67–95, 1994, announced at STOC 1989. 2, 3, 4

[14] A. Daniely, N. Linial, and S. Shalev-Shwartz, "From average case complexity to improper learning complexity," in *STOC*, 2014, pp. 441–448. 2, 4

[15] U. Feige, "Relations between average case complexity and approximation complexity," in *STOC*, 2002, pp. 534–543. 2

[16] B. Applebaum, B. Barak, and D. Xiao, "On basing lower-bounds for learning on worst-case assumptions," in *FOCS*, 2008, pp. 211–220. 3

[17] L. Pitt and L. G. Valiant, "Computational limitations on learning from examples," *J. ACM*, vol. 35, no. 4, pp. 965–984, 1988. 3

[18] R. Board and L. Pitt, "On the necessity of occam algorithms," *Theoretical Computer Science*, vol. 100, no. 1, pp. 157 – 184, 1992. [Online]. Available: http://www.sciencedirect.com/science/article/pii/030439759290367O 3

[19] S. Aaronson, "6.080/6.089 Great Ideas in Theoretical Computer Science, Spring 2008, Lecture 21," MIT OpenCourseWare, 2008, available at http://stellar.mit.edu/S/course/6/sp08/6.080/courseMaterial/topics/topic1/lectureNotes/lec21/lec21.pdf. 3

[20] N. Robertson and P. D. Seymour, "Graph minors .xiii. the disjoint paths problem," *J. Comb. Theory, Ser. B*, vol. 63, no. 1, pp. 65–110, 1995. 3

[21] J. Chuzhoy, "Routing in undirected graphs with constant congestion," in *STOC*, 2012, pp. 855–874. 3

[22] J. Chuzhoy and S. Li, "A polylogarithmic approximation algorithm for edge-disjoint paths with congestion 2," in *FOCS*, 2012, pp. 233–242. 3

[23] C. Chekuri, S. Khanna, and F. B. Shepherd, "Edge-disjoint paths in planar graphs with constant congestion," *SIAM J. Comput.*, vol. 39, no. 1, pp. 281–301, 2009. 3

[24] ——, "Multicommodity flow, well-linked terminals, and routing problems," in *STOC*, 2005, pp. 183–192. 3

[25] J. M. Kleinberg, "An approximation algorithm for the disjoint paths problem in even-degree planar graphs," in *FOCS*, 2005, pp. 627–636. 3

[26] J. M. Kleinberg and É. Tardos, "Approximations for the disjoint paths problem in high-diameter planar networks," *J. Comput. Syst. Sci.*, vol. 57, no. 1, pp. 61–73, 1998. 3

[27] K. Kawarabayashi and Y. Kobayashi, "The edge disjoint paths problem in eulerian graphs and 4-edge-connected graphs," in *SODA*, 2010, pp. 345–353. 3

[28] J. M. Kleinberg, "Approximation algorithms for disjoint paths problems," Ph.D. dissertation, Citeseer, 1996. 3, 4

[29] C. Chekuri and S. Khanna, "Edge-disjoint paths revisited," *ACM Transactions on Algorithms*, vol. 3, no. 4, 2007. 3, 4

[30] K. R. Varadarajan and G. Venkataraman, "Graph decomposition and a greedy algorithm for edge-disjoint paths," in *SODA*, 2004, pp. 379–380. 3, 4

[31] V. Guruswami, S. Khanna, R. Rajaraman, F. B. Shepherd, and M. Yannakakis, "Near-optimal hardness results and approximation algorithms for edge-disjoint paths and related problems," *J. Comput. Syst. Sci.*, vol. 67, no. 3, pp. 473–496, 2003, also, in STOC 1999. 3, 4

[32] M. Krivelevich, Z. Nutov, M. R. Salavatipour, J. Yuster, and R. Yuster, "Approximation algorithms and hardness results for cycle packing problems," *ACM Transactions on Algorithms*, vol. 3, no. 4, 2007. 3, 4, 9

[33] M. Andrews and L. Zhang, "Logarithmic hardness of the undirected edge-disjoint paths problem," *J. ACM*, vol. 53, no. 5, pp. 745–761, 2006, also, in STOC'05. 3

[34] M. Andrews, J. Chuzhoy, V. Guruswami, S. Khanna, K. Talwar, and L. Zhang, "Inapproximability of edge-disjoint paths and low congestion routing on undirected graphs," *Combinatorica*, vol. 30, no. 5, pp. 485–520, 2010. 3, 4

[35] M. Alekhnovich, M. Braverman, V. Feldman, A. R. Klivans, and T. Pitassi, "The complexity of properly learning simple concept classes," *J. Comput. Syst. Sci.*, vol. 74, no. 1, pp. 16–34, 2008, announced at FOCS 2004. 4

[36] P. Chalermsook, B. Laekhanukit, and D. Nanongkai, "Graph products revisited: Tight approximation hardness of induced matching, poset dimension and more," in *SODA*, 2013, pp. 1557–1576. 6, 7

[37] ——, "Independent set, induced matching, and pricing: Connections and tight (subexponential time) approximation hardnesses," in *FOCS*, 2013, pp. 370–379. 6

[38] ——, "Coloring graph powers: Graph product bounds and hardness of approximation," in *LATIN*, 2014, pp. 409–420. 6

[39] S. G. Kolliopoulos and C. Stein, "Approximating disjoint-path problems using packing integer programs," *Math. Program.*, vol. 99, no. 1, pp. 63–87, 2004. 9

[40] Z. Friggstad and M. R. Salavatipour, "Approximability of packing disjoint cycles," *Algorithmica*, vol. 60, no. 2, pp. 395–400, 2011. 9