Electronic Colloquium on Computational Complexity, Report No. 106 (2010)

Optimal Constant-Time Approximation Algorithms and (Unconditional) Inapproximability Results for Every Bounded-Degree CSP

Yuichi Yoshida*

School of Informatics, Kyoto University, and
Preferred Infrastructure, Inc.
yyoshida@lab2.kuis.kyoto-u.ac.jp

Abstract

Raghavendra (STOC 2008) gave an elegant and surprising result: if Khot's Unique Games Conjecture (STOC 2002) is true, then for every constraint satisfaction problem (CSP), the best approximation ratio is attained by a certain simple semidefinite programming and a rounding scheme for it.

In this paper, we show that a similar result holds for constant-time approximation algorithms in the bounded-degree model. Specifically, we present the followings: (i) For every CSP, we construct an oracle that serves an access, in constant time, to a nearly optimal solution of a basic LP relaxation of the CSP. (ii) Using the oracle, we present a constant-time rounding scheme that achieves an approximation ratio coincident with the integrality gap of the basic LP. (iii) We give a generic conversion from integrality gaps of basic LPs to hardness results. All of those results are "unconditional." Therefore, for every bounded-degree CSP, we give the best constant-time approximation algorithm among all.

Key words: Constant-time approximation, constraint satisfaction problems, linear programmings, rounding schemes.

*This work was conducted while the author was visiting Rutgers University.

ISSN 1433-8092

1 Introduction

In a *constraint satisfaction problem* (CSP), the objective is to find an assignment to a set of variables that satisfies the maximum number of a given set of constraints on them. Formally, a CSP Λ is specified by a set of predicates over alphabets $[q] = \{1, \ldots, q\}$. Every instance of Λ consists of a set of variables V, and a set of constraints $\mathcal{P}$ on them. Each constraint consists of a predicate from Λ applied to a subset of variables. The objective is to find an assignment to the variables that satisfies the maximum number of constraints. A large number of fundamental optimization problems, such as Max Cut and Max k-Sat, are examples of CSPs.

Approximation algorithms for CSPs have been intensively studied. Goemans and Williamson [10] first exploited semidefinite programmings (SDP) to approximate Max Cut and Max 2SAT achieving the approximation ratio ≈ 0.878. After this breakthrough, plethora of approximation algorithms using SDPs have been developed for many optimization problems [15, 21]. For inapproximability side, tight hardness results have been successfully obtained for some important optimization problems such as Max 3SAT [14]. However, the approximability of many interesting CSPs such as Max Cut and Max 2SAT remains open. Towards tightening this gap, Khot [16] introduced the Unique Games Conjecture (UGC). Assuming the UGC, tight hardness have been shown for Max Cut [17], Max 2SAT [5], and Max k-CSP [6, 27]. Finally, Raghavendra [25] succeeded to unify and generalize those approximation and inapproximability results for every CSP. Specifically, Raghavendra showed that, assuming the UGC, for every CSP, a certain SDP combined with a certain rounding scheme attains the best approximation ratio among all polynomial-time approximation algorithms. The ingenious technique in the proof is giving a generic conversion from integrality gaps of SDPs to hardness results via the UGC.

In this paper, we are concerned with constant-time approximation algorithms for bounded-degree CSPs. That is, algorithms are supposed to run in time irrespective of sizes of instances. We use the *bounded-degree model* for CSPs, which was originally introduced for graphs [12]. In this model, the number of alphabets, the maximum arity (the number of inputs to a predicate), the maximum degree (the number of constraints where a variable appears), and the maximum weight of a constraint is bounded by constants. Let $\mathcal{I}$ be a Λ-CSP instance. Since a constant-time algorithm cannot read the whole $\mathcal{I}$, we assume the existence of an oracle $\mathcal{O}_{\mathcal{I}}$ with which we can get information of $\mathcal{I}$. Let V and $\mathcal{P}$ be the variable set and the constraint set of $\mathcal{I}$, respectively. Also, let t be the maximum degree of $\mathcal{I}$. Then, by specifying a variable $v \in V$ and an index $i \in [t]$, $\mathcal{O}_{\mathcal{I}}$ returns $P \in \mathcal{P}$ where P is the ith constraint where v appears. If there is no such constraint, $\mathcal{O}_{\mathcal{I}}$ returns a special symbol. The efficiency of an algorithm is measured by the number of accesses to $\mathcal{O}_{\mathcal{I}}$, which is called *query complexity*.

In this paper, we show an analogous result to Raghavendra's result: for every CSP, a certain linear programming (LP) combined with a certain rounding scheme attains the best approximation ratio among all constant-time approximation algorithms. Furthermore, our result is *unconditional*.

To give the statement precisely, we need to define several notions. For a Λ-CSP instance $\mathcal{I}$ with the variable set V and the constraint set $\mathcal{P}$, there is a natural generic LP relaxation shown in Fig. 1, which we call BasicLP. Let $\mathbf{lp}(\mathcal{I})$ denote the objective value of an optimal solution of BasicLP for $\mathcal{I}$ and $\mathbf{opt}(\mathcal{I})$ denote the value of an optimal solution of $\mathcal{I}$. We define $W_{\mathcal{I}}$ as the sum of weights of constraints in $\mathcal{I}$. The *integrality gap* of BasicLP for $\mathcal{I}$ is defined as $\mathbf{opt}(\mathcal{I})/\mathbf{lp}(\mathcal{I})$. Also, the *integrality gap curve* $\alpha_\Lambda(c)$ is the minimum ratio of $\mathbf{opt}(\mathcal{I})/\mathbf{lp}(\mathcal{I})$, given $\mathbf{lp}(\mathcal{I}) = cW_{\mathcal{I}}$, where the minimum is taken over all instances $\mathcal{I}$ of a CSP Λ. Formally,

$$\alpha_\Lambda(c) = \inf_{\substack{\mathcal{I} \in \Lambda, \\ \mathbf{lp}(\mathcal{I}) = cW_{\mathcal{I}}}} \frac{\mathbf{opt}(\mathcal{I})}{\mathbf{lp}(\mathcal{I})}.$$

A value x is called an (α, β)-*approximation* to value x^* if it satisfies $\alpha x^* - \beta \leq x \leq x^*$. Let $\alpha(c) : [0,1] \to [0,1]$ be a function. Then, an algorithm is called an $(\alpha(c), \epsilon n)$-*approximation algorithm* for a CSP Λ if, given an oracle access $\mathcal{O}_{\mathcal{I}}$ to a Λ-CSP instance $\mathcal{I}$ of n variables with $\mathbf{lp}(\mathcal{I}) = cW_{\mathcal{I}}$, it computes an $(\alpha(c), \epsilon n)$-approximation to $\mathbf{opt}(\mathcal{I})$ with probability of at least 2/3.

Theorem 1.1. *For every CSP Λ and $\epsilon > 0$, there is a constant-time $(\alpha_\Lambda(c-\epsilon), \epsilon n)$-approximation algorithm for the CSP Λ, where n is the number variables in an input instance.*

Note that we must allow the additive error ϵn. To see this, suppose that the input instance consists of n variables and constant number of constraints. Then, we have to see at least one constraint if we do not allow the additive error. However, this is obviously impossible in constant time.

The proof of Theorem 1.1 consists of two parts. The first part (Section 3) states that we can compute a (nearly) optimal solution of BasicLP in constant-time. Since the size of the solution itself is non-constant, we do not explicitly create the whole solution. Instead, we construct an oracle that returns the value of a variable when we specify it. The second part (Section 4) states that, given an oracle access to a (nearly) optimal solution of BasicLP, we can efficiently compute an $(\alpha_\Lambda(c-\epsilon), \epsilon n)$-approximation to $\mathbf{opt}(\mathcal{I})$. Note that, for an instance $\mathcal{I}$ satisfying $\mathbf{lp}(\mathcal{I}) = cW_{\mathcal{I}}$, $\alpha_\Lambda(c)$ is the best possible approximation ratio we can hope for. Thus, the second part gives the *optimal rounding scheme* for BasicLP.

For hardness side, we show the following.

Theorem 1.2. *For every CSP $\Lambda, c \in [0,1]$ and $\epsilon > 0$, there exists $\delta > 0$ such that any $(\alpha_\Lambda(c) + \epsilon, \delta n)$-approximation algorithm for the CSP Λ requires $\Omega(\sqrt{n})$ queries, where n is the number of variables in an input instance.*

Combining with Theorem 1.1, we conclude that the algorithm given in Theorem 1.1 is not just the best among constant-time approximation algorithm using BasicLP, but the best among all constant-time approximation algorithms.

Theorem 1.2 has much implication in the setting of property testing. A Λ-CSP instance $\mathcal{I}$ is called *satisfiable* if there is an assignment to variables that satisfies all the constraints. Also, $\mathcal{I}$ is called ϵ-*far from* satisfiability if we must remove at least ϵtn constraints to make it satisfiable, where t is the maximum degree and n is the number of variables. If $\mathcal{I}$ is a unweighted instance, $\mathcal{I}$ is ϵ-far from satisfiability if the optimal value is at most $W_{\mathcal{I}} - \epsilon tn$. An algorithm is called a *testing algorithm* for satisfiability of a CSP Λ if, given an oracle access to a Λ-CSP instance, it accepts with probability of at least 2/3 if the instance is satisfiable, and rejects with probability of at least 2/3 if the instance is ϵ-far from satisfiability. Unlike the hardness result given in [25], Theorem 1.2 holds also for $c = 1$, i.e., satisfiable instances. Thus, if $\alpha_\Lambda(1) < 1$, then it indicates that there is no constant-time testing algorithm for satisfiability of the CSP Λ. Max Cut is an optimization problem for graphs in which we partition vertices into two sets so as to maximize the number of edges between them. It is now hard to show that $\alpha_{\text{Max Cut}}(1) < 1$, and Theorem 1.2 indicates that testing the satisfiability of Max Cut, or equivalently Bipartiteness, requires $\Omega(\sqrt{n})$ queries. However, it is know that Bipartiteness is testable with $\tilde{O}(\sqrt{n})$ queries [11]. Thus, the lower bound in Theorem 1.2 is almost tight.

1.1 Related Work

Subsequent to Raghavendra's work [25], under the UGC, certain SDPs and LPs are shown to be the best approximation algorithms for several classes of problems, such as graph labeling problems (including k-Way Cut, 0-Extension, and Metric Labeling) [22], kernel clustering problems [18],

ordering CSPs (including Maximum Acyclic Subgraph) [13], and strict monotone CSPs (including Minimum Vertex Cover) [20].

There have been many studies on constant-time approximation algorithms in the bounded-degree model. For algorithmic side, mainly graph problems have been studied, e.g., Minimum Spanning Tree [8], Minimum Vertex Cover [24, 23, 30], Maximum Matching [23, 30], Maximum Independent Set [1], and Minimum Dominating Set [23, 30]. For inapproximability results of graph problems, Minimum Dominating Set [1] and Maximum Independent Set [1, 29] have been considered. For CSPs, it is known that, for every $\epsilon > 0$, there exists $\delta > 0$ such that any $(1/2 + \epsilon, \delta n)$-approximation algorithm for Max E2LIN2 and $(7/8 + \epsilon, \delta n)$-approximation algorithm for Max E3SAT require linear number of queries [7]. In [29], it was shown that, for every CSP using only symmetric predicates, for every $\epsilon > 0$ there exists $\delta > 0$ such that any approximation algorithm better than the random assignment by ϵ requires $\Omega(n^{1/2+\delta})$ queries.

We can compute the optimal solution of CSPs within an additive error $O(\epsilon n^s)$ by sampling $\mathrm{poly}(1/\epsilon)$ variables and by solving the induced problem, where s is the maximum arity and n is the number of variables [2]. Thus, dense instances are easy to approximate with constant queries [2, 3, 4]. This is the reason why we are concerned with sparse instances in this paper.

1.2 Proof Overview

Let $\mathcal{O}_{\mathcal{I}}$ be the oracle access to a Λ-CSP instance $\mathcal{I}$. First, we construct an oracle access $\mathcal{O}_{\mathbf{lp}}$ to LP solution $(\boldsymbol{x}, \boldsymbol{\mu})$ of BasicLP for $\mathcal{I}$, where $(\boldsymbol{x}, \boldsymbol{\mu})$ is a nearly optimal solution of BasicLP. Namely, if we specify a variable $\boldsymbol{x}_{v,a} (v \in V, a \in [q])$ or $\boldsymbol{\mu}_{P,\beta} (P \in \mathcal{P}, \beta \in [q]^{V(P)})$, $\mathcal{O}_{\mathbf{lp}}$ outputs its value by accessing $\mathcal{O}_{\mathcal{I}}$ constant times. To this end, we use a distributed algorithm for packing/covering LP given in [19]. In the distributed setting, a linear programing is bound to a graph $G = (V, E)$. Each primal variable $\boldsymbol{x}_i$ and each dual variable $\boldsymbol{y}_j$ is associated with a vertex $v_i^p \in V$ and $v_j^d \in V$, respectively. There are edges between primal and dual vertices wherever the respective variables occur in the corresponding inequality. Thus, $(v_i^p, v_j^d) \in E$ if and only if $\boldsymbol{x}_i$ occurs in the jth inequality of the primal. Let $G_{v,k}$ denote the graph induced by vertices whose distance from v is at most k. Then, a *distributed algorithm in k rounds* works in such a way that each vertex outputs a value of the corresponding variable based on $G_{v,k}$. In [19], it is shown that if the matrix in the LP is "sparse," then there is a distributed algorithm that computes a nearly optimal solution of the LP in k rounds, where k is an integer determined by the sparsity of the LP. Suppose that the degree of the graph is bounded by Δ. Then, given a variable, we can compute the value for it with Δ^k queries in the bounded-degree model by simulating the process of the distributed algorithm, and we achieve $\mathcal{O}_{\mathbf{lp}}$. Though BasicLP is not a packing/covering LP, after applying several number of transformations, we get a packing LP that has essentially the same behavior under approximation. Technically, we need to show that BasicLP is robust in the sense that even if we violate each constraint by small amount, the optimal value does not significantly increase.

Next, we exhibit a solution to the original instance by rounding the LP solution given by $\mathcal{O}_{\mathbf{lp}}$. In [25], instead of BasicLP, Raghavendra considered a certain SDP relaxation, which we call BasicSDP. In [26], Raghavendra and Stuerer showed an optimal rounding scheme for BasicSDP, i.e., get an approximation ratio coincident with the integrality gap of BasicSDP. Our proof is based on their work. First, from an instance $\mathcal{I}$ and its LP solution $(\boldsymbol{x}, \boldsymbol{\mu})$, we create another instance $\mathcal{I}'$ by merging variables of $\mathcal{I}$ that have close values with respect to $(\boldsymbol{x}, \boldsymbol{\mu})$ so that the number of variables of $\mathcal{I}'$ become constant. Though we cannot explicitly construct $\mathcal{I}'$ since the number of constraints is not constant, it is possible to enumerate variables without constructing $\mathcal{I}'$. Then, we perform brute force search on $\mathcal{I}'$. That is, using the oracle $\mathcal{O}_{\mathbf{lp}}$, we estimate the value obtained by each assignment to variables in $\mathcal{I}'$, Finally, we take the maximum of them as the output. Since an assignment to

$\mathcal{I}'$ can be unfolded to the original instance $\mathcal{I}$, $\mathbf{opt}(\mathcal{I})/W_{\mathcal{I}}$ is higher than $\mathbf{opt}(\mathcal{I}')/W_{\mathcal{I}'}$. Also, the crucial fact is that the LP optimum does not change significantly by merging variables. Thus, we can establish an $(\alpha_\Lambda(c-\epsilon), \epsilon n)$-approximation to $\mathbf{opt}(\mathcal{I})$ since we can get an $(\alpha_\Lambda(c-\epsilon), \epsilon n)$-approximation to $\mathbf{opt}(\mathcal{I}')$ by brute force search.

Now, we describe a proof sketch of the hardness result. Let $(\boldsymbol{x}^*, \boldsymbol{\mu}^*)$ be the optimal LP solution of BasicLP for $\mathcal{I}$. First, we create a distribution of instances $\mathcal{D}^{\mathbf{opt}}$ by blowing up variables of BasicLP. With high probability, $\mathcal{J} \in \mathcal{D}^{\mathbf{opt}}$ satisfies that $\mathbf{opt}(\mathcal{J})/W_{\mathcal{J}} \leq \mathbf{opt}(\mathcal{I})/W_{\mathcal{I}} + \epsilon$ where ϵ is an arbitrarily small constant. Next, using the LP solution $(\boldsymbol{x}^*, \boldsymbol{\mu}^*)$, we create another distribution of instances $\mathcal{D}^{\mathbf{lp}}$, which has the property that for all $\mathcal{J} \in \mathcal{D}^{\mathbf{lp}}$, $\mathbf{opt}(\mathcal{J})/W_{\mathcal{J}} \geq \mathbf{lp}(\mathcal{I})/W_{\mathcal{I}}$. From Yao's minimax principle, by showing that any deterministic algorithm that distinguishes $\mathcal{D}^{\mathbf{opt}}$ from $\mathcal{D}^{\mathbf{lp}}$ with high probability requires $\Omega(\sqrt{n})$ queries, we conclude that any $(\mathbf{lp}(\mathcal{I})/\mathbf{opt}(\mathcal{I}) - \epsilon, \epsilon n)$-approximation algorithm requires $\Omega(\sqrt{n})$ queries. By choosing an instance with integrality gap $\alpha_\Lambda(c)$ as $\mathcal{I}$, we have the desired result.

1.3 Organization

In Section 2, we give notations and basic technical tools used in this paper. In Section 3, we present an oracle access $\mathcal{O}_{\mathbf{lp}}$ to a (nearly) optimal solution of BasicLP. Section 4 is devoted to describe how to round the LP solution optimally. We give the proof of robustness of BasicLP in Section 5. In Section 6, we show that any $(\alpha_\Lambda(c) + \epsilon, \delta n)$-approximation algorithm requires $\Omega(\sqrt{n})$ queries.

2 Preliminaries

2.1 Definitions

The *arity* of a predicate $P : [q]^k \to \{0,1\}$ is the number of inputs to P, i.e., k here. The *degree* of a variable is the number of constraints where the variable appears. For a constraint P in a CSP instance, $V(P)$ denotes the set of variables in P. Let β be a vector or a set indexed by elements of a set V. For a subset $S \subseteq V$, we define $\beta_{|S} = \{\beta_v\}_{v \in S}$.

Definition 2.1. *A bounded-degree constraint satisfaction problem Λ is specified by $\Lambda = ([q], s, t, w, \mathbb{P})$, where $[q] = \{1, \ldots, q\}$ is a finite domain, s is the maximum arity of predicates, t is the maximum degree of variable,s w is the maximum weight of predicates, and $\mathbb{P} = \{P : [q]^k \to \{0,1\} | k \leq t\}$ is a set of predicates.*

In this paper, a CSP stands for a bounded-degree constraint satisfaction problem. Also, symbols q, s, t and w are used to denote the number of alphabets, the maximum arity, the maximum degree, and the maximum weight throughout this paper unless mentioned otherwise.

Definition 2.2. *An instance $\mathcal{I}$ of a CSP $\Lambda = ([q], s, t, w, \mathbb{P})$ is given by $\mathcal{I} = (V, \mathcal{P}, \boldsymbol{w})$, where*

- *$V = \{v_1, \ldots, v_n\}$ is a set of variables taking values over $[q]$,*
- *$\mathcal{P}$ is a set of constraints, consisting of predicates $P \in \mathbb{P}$ applied to sequences S of variables V of size at most s. More precisely, when a predicate P is applied to a sequence $S = \{i_1, \ldots, i_k\} \subseteq [n]^k$, P takes variables $V_{|S} = \{v_{i_1}, \ldots, v_{i_k}\}$ as the input. Also, the degree of each variable is bounded by t.*
- *$\boldsymbol{w}$ is a set of weights $\{\boldsymbol{w}_P\}_{P \in \mathcal{P}}$ assigned to each constraint $P \in \mathcal{P}$, where $0 \leq \boldsymbol{w}_P \leq w$.*

$$
\begin{array}{lll}
\max & \displaystyle\sum_{P\in\mathcal{P}} \boldsymbol{w}_P \left(\sum_{\beta\in[q]^{V(P)}} P(\beta)\boldsymbol{\mu}_{P,\beta} \right) & \\
\text{s.t.} & \displaystyle\sum_{a\in[q]} \boldsymbol{x}_{v,a} = 1 & \forall v \in V \\
& \displaystyle\sum_{\beta\in[q]^{V(P)},\beta_v=a} \boldsymbol{\mu}_{P,\beta} = \boldsymbol{x}_{v,a} & \forall P \in \mathcal{P}, v \in V(P), a \in [q] \\
& \boldsymbol{x}_{v,a} \geq 0 & \forall v \in V \\
& \boldsymbol{\mu}_{P,\beta} \geq 0 & \forall P \in \mathcal{P}, \beta \in [q]^{V(P)}.
\end{array}
$$

Figure 1: BasicLP for a Λ-CSP instance $\mathcal{I} = (V, \mathcal{P}, \boldsymbol{w})$

The objective is to find an assignment to variables $\beta : V \to [q]$ that maximizes the number of satisfied constraints, i.e., $\sum_{P\in\mathcal{P}} \boldsymbol{w}_P P(\beta)$.

We consider an LP relaxation for a CSP Λ, which we call BasicLP (see Fig. 1). The goal is to find a collection of vectors $\{\boldsymbol{x}_{v,a}\}_{v\in V,a\in[q]}$ and $\{\boldsymbol{\mu}_{P,\beta}\}_{P\in\mathcal{P},\beta\in[q]^{V(P)}}$. Here, $\boldsymbol{x}_v = \{\boldsymbol{x}_{v,a}\}_{a\in[q]}$ (resp., $\boldsymbol{\mu}_P = \{\boldsymbol{\mu}_{P,\beta}\}_{\beta\in[q]^{V(P)}}$) can be seen as a distribution over assignments to a variable $v \in V$ (resp., a constraint $P \in \mathcal{P}$), and we often identify them as distributions. For an LP solution $(\boldsymbol{x}, \boldsymbol{\mu})$, we define $\mathbf{val}(\mathcal{I}, \boldsymbol{x}, \boldsymbol{\mu})$ as the value of the objective function of BasicLP obtained by $(\boldsymbol{x}, \boldsymbol{\mu})$. Let $\beta : V \to [q]$ be an assignment to variables. Then, $\mathbf{val}(\mathcal{I}, \beta)$ denote the value of $\mathcal{I}$ (not BasicLP for $\mathcal{I}$) obtained by this assignment. We call an LP solution $(\boldsymbol{x}, \boldsymbol{\mu})$ *ϵ-infeasible* if it satisfies constraints of the form $\boldsymbol{x}_{v,a} \geq 0$ and $\boldsymbol{\mu}_{P,x} \geq 0$ and violates other constraints by at most ϵ. Also, we call an LP solution $(\boldsymbol{x}, \boldsymbol{\mu})$ *(α, β)-approximate* if $\mathbf{val}(\mathcal{I}, \boldsymbol{x}, \boldsymbol{\mu})$ is an (α, β)-approximation to $\mathbf{lp}(\mathcal{I})$.

2.2 Basic Tools

As a simple application of Hoeffding's inequality, we obtain the following.

Lemma 2.3. *Suppose that we have an oracle access to a function $f : [n] \to [0, w]$. That is, by specifying $x \in [n]$ as a query, we can see the value of $f(x)$. Then, by querying $O(\frac{w^2}{\epsilon^2} \log \frac{1}{\delta})$ times, with probability of at least $1 - \delta$, we can compute a $(1, \epsilon n)$-approximation to $\sum_i f(i)$.*

Let $\mathcal{I}$ be a Λ-CSP instance. Not surprisingly, we cannot compute the optimal solution $(\boldsymbol{x}^*, \boldsymbol{\mu}^*)$ of BasicLP for $\mathcal{I}$ in constant time. Even worse, it is also hard to obtain a feasible solution in constant time. Instead, as we will see, we can compute a feasible (nearly) optimal solution $(\boldsymbol{x}, \boldsymbol{\mu})$ of a resulting LP obtained by relaxing equality constraints. Though this is an infeasible solution for the original LP, The following lemma states that $\mathbf{val}(\mathcal{I}, \boldsymbol{x}, \boldsymbol{\mu})$ is close to $\mathbf{lp}(\mathcal{I})$. The proof is given in Section 5.

Lemma 2.4 (Robustness of BasicLP)**.** *Let $\mathcal{I}$ be a Λ-CSP instance. Suppose that $(\boldsymbol{x}, \boldsymbol{\mu})$ is an ϵ-infeasible LP solution for $\mathcal{I}$ of value $cW_{\mathcal{I}}$. Then, it holds that*

$$\mathbf{lp}(\mathcal{I}) \geq (c - \epsilon \cdot \mathrm{poly}(qs))W_{\mathcal{I}}.$$

3 A $(1 - \epsilon, \epsilon n)$-approximation algorithm for BasicLP

In this section, we show the following theorem.

Theorem 3.1. *Let $\mathcal{I}$ be a Λ-CSP instance of n variables. Given an oracle access $\mathcal{O}_{\mathcal{I}}$ to $\mathcal{I}$, for any $\epsilon > 0$, we can construct an oracle that gives an access to an ϵ-feasible $(1-\epsilon, \epsilon n)$-approximate LP solution $(\boldsymbol{x}, \boldsymbol{\mu})$. For each query, the number of queries performed to $\mathcal{O}_{\mathcal{I}}$ is at most* $\exp(\exp(\mathrm{poly}(qstw/\epsilon)))$.

As described in the introduction, we utilize a distributed algorithm that solves a packing LP given in [19]. A packing LP is expressed as follows.

$$\begin{array}{ll} \max & \boldsymbol{b}^T \boldsymbol{z} \\ \text{s.t.} & A^T \boldsymbol{z} \le \boldsymbol{c} \\ & \boldsymbol{z} \ge 0, \end{array} \tag{1}$$

where $A \in \mathbb{R}_+^{m \times n}$ is a non-negative matrix and $\boldsymbol{b}, \boldsymbol{c} \in \mathbb{R}_+^n$ are non-negative vectors. Then, there is a distributed algorithm in a constant round to compute a nearly optimal solution of the packing LP (see Appendix A for a formal statement).

When a variable $\boldsymbol{x}_{v,a}, v \in V, a \in [q]$ or $\boldsymbol{\mu}_{P,\beta}, P \in \mathcal{P}, \beta \in [q]^{V(P)}$ is specified as a query, we locally simulate the distributed algorithm and output the value for it. The only issue is that BasicLP is not a packing LP. Thus, this section is devoted to transform BasicLP to a packing LP, and we will show that we can restore a good approximation to BasicLP from an approximation to the resulting packing LP.

First, we substitute $\boldsymbol{x}_{v,a}$ by $1 - \boldsymbol{x}_{v,a}$ and relax each equality constraint by ϵ. Then, we obtain the following LP.

$$\begin{array}{lll} \max & \displaystyle\sum_{P \in \mathcal{P}} \boldsymbol{w}_P \left(\sum_{\beta \in [q]^{V(P)}} P(\beta) \boldsymbol{\mu}_{P,\beta} \right) & \\ \text{s.t.} & \displaystyle\sum_{a \in [q]} \boldsymbol{x}_{v,a} \le q - 1 + \epsilon & \forall v \in V \\ & \displaystyle\sum_{a \in [q]} \boldsymbol{x}_{v,a} \ge q - 1 - \epsilon & \forall v \in V \\ & \boldsymbol{x}_{v,a} + \displaystyle\sum_{\beta \in [q]^{V(P)}, \beta_v = a} \boldsymbol{\mu}_{P,\beta} \le 1 + \epsilon & \forall P \in \mathcal{P}, v \in V(P), a \in [q] \\ & \boldsymbol{x}_{v,a} + \displaystyle\sum_{\beta \in [q]^{V(P)}, \beta_v = a} \boldsymbol{\mu}_{P,\beta} \ge 1 - \epsilon & \forall P \in \mathcal{P}, v \in V(P), a \in [q] \\ & \boldsymbol{x}_{v,a} \ge 0 & \forall v \in V \\ & \boldsymbol{\mu}_{P,x} \ge 0 & \forall P \in \mathcal{P}, \beta \in [q]^{V(P)}. \end{array} \tag{2}$$

Lemma 3.2. *Let $\mathcal{I}$ be a Λ-CSP instance and $(\boldsymbol{x}, \boldsymbol{\mu})$ be an ϵ-infeasible solution for LP (2) of value $cW_{\mathcal{I}}$. Then, $\mathbf{lp}(\mathcal{I}) \ge (c - \epsilon \cdot \mathrm{poly}(kq))W_{\mathcal{I}}$ holds.*

Proof. Clearly, $(1 - \boldsymbol{x}, \boldsymbol{\mu})$ is an 2ϵ-infeasible solution for BasicLP of value c. From Lemma 2.4, the lemma holds. □

Next, to make the directions of the inequalities the same, we introduce a complement variable for each variable, i.e., we define $\overline{\boldsymbol{x}}_{v,a} = 1 - \boldsymbol{x}_{v,a}$ and $\overline{\boldsymbol{\mu}}_{P,\beta} = 1 - \boldsymbol{\mu}_{P,\beta}$. However, such equality constraints cannot be used in a packing LP. Thus, we relax those equality constraints again. That is, we introduce constraints of the form $\boldsymbol{x}_{v,a} + \overline{\boldsymbol{x}}_{v,a} \le 1$ and $\boldsymbol{\mu}_{P,\beta} + \overline{\boldsymbol{\mu}}_{P,\beta} \le 1$. Instead, to discourage them to become much smaller than one, we add additional terms to the objective function. We get the following LP.

$$
\begin{array}{lll}
\max & \displaystyle\sum_{P\in\mathcal{P}} \boldsymbol{w}_P \left(\sum_{\beta\in[q]^{V(P)}} P(\beta)\boldsymbol{\mu}_{P,\beta} \right) + \mathbf{1}^T(\boldsymbol{x}+\overline{\boldsymbol{x}}) + \mathbf{1}^T(\boldsymbol{\mu}+\overline{\boldsymbol{\mu}}), & \\
\text{s.t.} & \displaystyle\sum_{a\in[q]} \boldsymbol{x}_{v,a} \le q-1+\epsilon & \forall v\in V \\
& \displaystyle\sum_{a\in[q]} \overline{\boldsymbol{x}}_{v,a} \le 1+\epsilon & \forall v\in V \\
& \boldsymbol{x}_{v,a} + \displaystyle\sum_{\beta\in[q]^{V(P)},\beta_v=a} \boldsymbol{\mu}_{P,x} \le 1+\epsilon & \forall P\in\mathcal{P}, v\in V(P), a\in[q] \\
& \overline{\boldsymbol{x}}_{v,a} + \displaystyle\sum_{x\in[q]^{V(P)},\beta_v=a} \overline{\boldsymbol{\mu}}_{P,\beta} \le q^{|V(P)|-1}+\epsilon & \forall P\in\mathcal{P}, v\in V(P), a\in[q] \\
& \boldsymbol{x}_{v,a}+\overline{\boldsymbol{x}}_{v,a}\le 1, \quad \boldsymbol{x}_{v,a}\ge 0, \quad \overline{\boldsymbol{x}}_{v,a}\ge 0 & \forall v\in V \\
& \boldsymbol{\mu}_{P,\beta}+\overline{\boldsymbol{\mu}}_{P,\beta}\le 1, \quad \boldsymbol{\mu}_{P,\beta}\ge 0, \quad \overline{\boldsymbol{\mu}}_{P,\beta}\ge 0 & \forall P\in\mathcal{P}, \beta\in[q]^{V(P)}.
\end{array}
\tag{3}
$$

Fortunately, the optimal solutions of LP (3) and LP (2) are essentially the same.

Lemma 3.3 ([9])**.** *Let $\mathcal{I}$ be a Λ-CSP instance and $(\boldsymbol{x}^*, \overline{\boldsymbol{x}}^*, \boldsymbol{\mu}^*, \overline{\boldsymbol{\mu}}^*)$ be the optimal solution of LP (3) with value $cW_{\mathcal{I}} + N$ where N is the number of variables in LP (3). Then, $\boldsymbol{x}^* + \overline{\boldsymbol{x}}^* = \mathbf{1}$ and $\boldsymbol{\mu}^* + \overline{\boldsymbol{\mu}}^* = \mathbf{1}$ hold. Also, $(\boldsymbol{x}^*, \boldsymbol{\mu}^*)$ is the optimal solution of LP (2) with value $cW_{\mathcal{I}}$.* □

Now, using the distributed algorithm given by [19], we have the following lemma. The analysis of the query complexity is tedious and the proof is given in Appendix A.

Lemma 3.4. *Let $\mathcal{I}$ be a Λ-CSP instance. Given an oracle access $\mathcal{O}_{\mathcal{I}}$ to $\mathcal{I}$, for any $\epsilon > 0$, we can construct an oracle that serves an access to $(\boldsymbol{x}, \overline{\boldsymbol{x}}, \boldsymbol{\mu}, \overline{\boldsymbol{\mu}})$, which is a feasible $(1-\epsilon, 0)$-approximate solution to LP (3). For each query, the number of queries performed to $\mathcal{O}_{\mathcal{I}}$ is at most* $\exp(\mathrm{poly}(qstw/\epsilon))$.

Proof of Theorem 3.1. Let $(\boldsymbol{x}, \overline{\boldsymbol{x}}, \boldsymbol{\mu}, \overline{\boldsymbol{\mu}})$ be a feasible $(1-\epsilon', 0)$-approximate solution obtained by Lemma 3.4, where ϵ' is a constant determined later. For notational simplicity, we write the objective function as $\boldsymbol{w}^T\boldsymbol{\mu} + \mathbf{1}^T(\boldsymbol{z}+\overline{\boldsymbol{z}})$ where $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{\mu})$. Let $(\boldsymbol{z}^*, \overline{\boldsymbol{z}}^*)$ be the optimal solution of LP (3). From Lemma 3.3, $\boldsymbol{z}^* + \overline{\boldsymbol{z}}^* = \mathbf{1}$. Also, let $N \le qn + q^s \cdot tn = (q + tq^s)n$ be the number of variables, where n is the number of variables in $\mathcal{I}$. Then, we have

$$
\boldsymbol{w}^T\boldsymbol{\mu} + \mathbf{1}^T(\boldsymbol{z}+\overline{\boldsymbol{z}}) \ge (1-\epsilon')(\boldsymbol{w}^T\boldsymbol{\mu}^* + \mathbf{1}^T(\boldsymbol{z}^*+\overline{\boldsymbol{z}}^*)) = (1-\epsilon')(\boldsymbol{w}^T\boldsymbol{\mu}^* + N).
$$

Thus,

$$
\begin{aligned}
\mathbf{1}^T(\boldsymbol{z}+\overline{\boldsymbol{z}}) &\ge (1-\epsilon')N + (1-\epsilon')(\boldsymbol{w}^T\boldsymbol{\mu}^* - \boldsymbol{w}^T\boldsymbol{\mu}) - \epsilon'\boldsymbol{w}^T\boldsymbol{\mu} \ge N - \epsilon'(w+1)N, \\
\boldsymbol{w}^T\boldsymbol{\mu} &\ge (1-\epsilon')\boldsymbol{w}^T\boldsymbol{\mu}^* + (1-\epsilon')(N - \mathbf{1}^T(\boldsymbol{z}+\overline{\boldsymbol{z}})) - \epsilon'\mathbf{1}^T(\boldsymbol{z}+\overline{\boldsymbol{z}}) \ge (1-\epsilon')\boldsymbol{w}^T\boldsymbol{\mu}^* - \epsilon' N.
\end{aligned}
$$

In the former inequality, we used the fact that $\boldsymbol{w}^T\boldsymbol{\mu}^* \ge \boldsymbol{w}^T\boldsymbol{\mu}$ and $\boldsymbol{w}^T\boldsymbol{\mu} \le wN$. Also, in the latter inequality, we used the fact that $N \ge \mathbf{1}^T(\boldsymbol{z}+\overline{\boldsymbol{z}})$.

From the former inequality, we have

$$
\mathbf{1}^T(\mathbf{1} - \boldsymbol{z} - \overline{\boldsymbol{z}}) \le \epsilon'(w+1)N.
$$

Let S be the set of variables $\boldsymbol{z}_i$ ($= \boldsymbol{x}_{v,a}$ or $\boldsymbol{\mu}_{P,\beta}$) such that $(1 - \boldsymbol{z}_i - \overline{\boldsymbol{z}_i}) \ge \epsilon''$ where ϵ'' is a constant determined later. From Markov's inequality, we have $|S| \le \epsilon'/\epsilon'' \cdot (w+1)N$. Let $S_{\boldsymbol{x}} = S \cap \{\boldsymbol{x}_{v,a}\}_{v\in V, a\in[q]}$ and $S_{\boldsymbol{\mu}} = S \cap \{\boldsymbol{\mu}_{P,\beta}\}_{P\in\mathcal{P},\beta\in[q]^{V(P)}}$. The variables in $S_{\boldsymbol{x}}$ and $S_{\boldsymbol{\mu}}$ are problematic

since constraints in LP (3) involving them are far from being satisfied. Thus, in what follows, we modify these variables and obtain nearly feasible solution of LP (3).

First, we construct variables $\{\boldsymbol{x}'_{v,a}\}_{v\in V,a\in[q]}$ in such a way that $\boldsymbol{x}'_{v,a} = \boldsymbol{x}_{v,a}$ if none of $\{\boldsymbol{x}_{v,a'}\}_{a'\in[q]}$ is in $S_{\boldsymbol{x}}$, and $\boldsymbol{x}'_{v,a} = 1/q$ if otherwise. Then, we construct variables $\{\boldsymbol{\mu}'_{P,\beta}\}_{P\in V(P),\beta\in[q]^{V(P)}}$ as follows. If none of $\{\boldsymbol{x}_{v,a}\}_{v\in V(P),a\in[q]}$ was modified in the previous step, we set $\boldsymbol{\mu}'_{P,\beta} = \boldsymbol{\mu}_{P,\beta}$. If otherwise, we set the values of $\{\boldsymbol{\mu}'_{P,\beta}\}_{\beta\in[q]^{V(P)}}$ in such a way that the distribution $\boldsymbol{\mu}'_P$ becomes consistent with the marginal distributions determined by $\{\boldsymbol{x}'_v\}_{v\in P}$. Note that each modification to $\boldsymbol{x}$ in the previous step involves at most $2tq^s$ modifications to $\boldsymbol{\mu}$.

We calculate the decrease of the objective function. The decrease caused by the modification to $\boldsymbol{x}_{v,a}$ is at most $\sum_{P\ni v}\boldsymbol{w}_P \le tw$, and the decrease caused by the modification to $\boldsymbol{\mu}_{P,\beta}$ is at most $\boldsymbol{w}_P \le w$. Thus, the total decrease is at most $tw|S| + 2twq^s|S| \le \epsilon'/\epsilon''\cdot(1+2q^s)tw(w+1)N$.

Note that for each unmodified variable $\boldsymbol{z}_i$ ($= \boldsymbol{x}_{v,a}$ or $\boldsymbol{\mu}_{P,\beta}$), $\boldsymbol{z}'_i + \overline{\boldsymbol{z}}'_i \ge 1-\epsilon''$ holds. Thus, $(\boldsymbol{x}', \overline{\boldsymbol{x}}', \boldsymbol{\mu}', \overline{\boldsymbol{\mu}}')$ is an ϵ''-infeasible solution with value of at least

$$\begin{aligned} & (1-\epsilon')\boldsymbol{w}^T\boldsymbol{\mu}^* - \epsilon' N - \epsilon'/\epsilon''\cdot(1+2q^s)tw(w+1)N \\ \ge\ & (1-\epsilon')\boldsymbol{w}^T\boldsymbol{\mu}^* - \epsilon'(1+(1+2q^s)tw(w+1)/\epsilon'')(q+tq^s)n. \end{aligned}$$

Thus, $(\boldsymbol{x}', \boldsymbol{\mu}')$ is an ϵ''-infeasible $(1-\epsilon', \epsilon'(1+(1+2q^s)tw(w+1)/\epsilon'')(q+tq^s)n)$-approximate solution. By choosing $\epsilon' = \epsilon^2/(q^{O(s)}\mathrm{poly}(tw))$ and $\epsilon'' = \epsilon$, we have an ϵ-feasible $(1-\epsilon, \epsilon n)$-approximate solution.

We need to look at q variables $\{\boldsymbol{x}_{v,a}\}_{a\in[q]}$ to decide the value of $\boldsymbol{x}'_{v,a}$, and we need to look at at most qs variables $\{\boldsymbol{x}_{v,a}\}_{v\in V(P),a\in[q]}$ to decide the value of $\boldsymbol{\mu}'_{P,\beta}$. Thus, the query complexity is

$$\max(q, qs)\exp(\mathrm{poly}(qstw/\epsilon')) = \exp(\exp(\mathrm{poly}(qstw/\epsilon))).$$

□

4 Optimal Rounding of BasicLP

In this section, we describe how we can optimally round LP solutions, and we give the proof of Theorem 1.1.

Variable folding: Let $\mathcal{I} = (V, \mathcal{P}, \boldsymbol{w})$ be a Λ-CSP instance. For a mapping $\phi : V \to V'$, we define a new Λ-CSP instance $\mathcal{I}/\phi = (V', \mathcal{P}', \boldsymbol{w}')$ on the variable set V' by identifying variables of $\mathcal{I}$ that get mapped to the same variable in V'. That is, for each constraint $P \in \mathcal{P}$ on the variable set $\{v_1, \dots, v_k\}$ with the weight $\boldsymbol{w}_P$, we have a constraint $P' \in \mathcal{P}'$ on the variable set $\{\phi(v_1), \dots, \phi(v_k)\}$ with the weight $\boldsymbol{w}_P$.

For $x \in [0,1]$, we define $x^\epsilon = (k+1)\epsilon$ where k is the positive integer such that $k\epsilon < x \le (k+1)\epsilon$. If $x = 0$, then we define $x^\epsilon = 0$. Let $(\boldsymbol{x}, \boldsymbol{\mu})$ be an LP solution for $\mathcal{I}$. We identify variables of $\mathcal{I}$ that have the same values $\{\boldsymbol{x}^\epsilon_{v,a}\}_{a\in[q]}$. Formally, we output another Λ-CSP instance $\mathcal{I}/\phi_{\boldsymbol{x}}$ where $\phi_{\boldsymbol{x}} : V \to \{0, \dots, 1/\epsilon\}^q$ is defined as

$$\phi_{\boldsymbol{x}}(v) = (\boldsymbol{x}^\epsilon_{v,1}, \dots, \boldsymbol{x}^\epsilon_{v,q}).$$

In what follows, we assume that $1/\epsilon$ is an integer. This is achieved by slightly decreasing ϵ until $1/\epsilon$ become an integer.

Lemma 4.1. *Let $\mathcal{I}$ be a Λ-CSP instance and $(\boldsymbol{x}, \boldsymbol{\mu})$ be an ϵ-infeasible LP solution for $\mathcal{I}$. Then, $(\boldsymbol{x}^\epsilon, \boldsymbol{\mu})$ is a $(q+1)\epsilon$-infeasible LP solution for $\mathcal{I}$.*

Proof. Since we move each $\boldsymbol{x}_{v,a}$ by at most ϵ, each constraint $\sum_{a\in[q]} \boldsymbol{x}^\epsilon_{v,a} = 1$ can be at most $(q+1)\epsilon$-infeasible. Also, each constraint $\sum_{\beta\in[q]^{V(P)},\beta_v=a} \boldsymbol{\mu}_{P,\beta} = \boldsymbol{x}^\epsilon_{v,a}$ can be at most 2ϵ-infeasible. □

Lemma 4.2. *Let $\mathcal{I}$ be a Λ-CSP instance of n variables and $(\boldsymbol{x}, \boldsymbol{\mu})$ be an ϵ-infeasible $(1-\epsilon, \epsilon)$-approximate LP solution for $\mathcal{I}$, where $\epsilon > 0$ is a small constant. Then, the variable folding $\mathcal{I}/\phi_{\boldsymbol{x}}$ satisfies that*

- $\mathbf{lp}(\mathcal{I}/\phi_{\boldsymbol{x}}) \geq \mathbf{lp}(\mathcal{I}) - \epsilon \cdot \mathrm{poly}(qstw)n$,
- *The variable set of $\mathcal{I}/\phi_{\boldsymbol{x}}$ has a cardinality* $\exp(\mathrm{poly}(q/\epsilon))$.

Proof. Since the range of $\phi_{\boldsymbol{x}}$ is $(1/\epsilon)^{O(q)}$, the second claim is obvious.

Suppose that $(\boldsymbol{x}, \boldsymbol{\mu})$ has an LP value $cW_{\mathcal{I}}$. From the fact that $(\boldsymbol{x}, \boldsymbol{\mu})$ is a $(1-\epsilon, \epsilon)$-approximate solution, we have $cW_{\mathcal{I}} \geq (1-\epsilon)\mathbf{lp}(\mathcal{I}) - \epsilon n$. Also, by Lemma 4.1, $(\boldsymbol{x}^\epsilon, \boldsymbol{\mu})$ is a $(q+1)\epsilon$-infeasible LP solution. Since only $\boldsymbol{\mu}$ affects the value of the objective function, the LP value of $(\boldsymbol{x}^\epsilon, \boldsymbol{\mu})$ equals $cW_{\mathcal{I}}$. A key observation is that $(\boldsymbol{x}^\epsilon, \boldsymbol{\mu})$ is also an LP solution for the folded instance $\mathcal{I}/\phi_{\boldsymbol{x}}$. Thus, we see that $\mathcal{I}/\phi_{\boldsymbol{x}}$ has a $(q+1)\epsilon$-infeasible solution of value at least $cW_{\mathcal{I}}$. Lemma 2.4 asserts that

$$\begin{aligned}
\mathbf{lp}(\mathcal{I}/\phi_{\boldsymbol{x}}) &\geq (c - (q+1)\epsilon \cdot \mathrm{poly}(qs))W_{\mathcal{I}} \\
&\geq (1-\epsilon)\mathbf{lp}(\mathcal{I}) - \epsilon n - \epsilon \cdot \mathrm{poly}(qs)W_{\mathcal{I}} \\
&\geq \mathbf{lp}(\mathcal{I}) - \epsilon \cdot \mathrm{poly}(qstw)n.
\end{aligned}$$

In the last inequality, we use the fact that $\mathbf{lp}(\mathcal{I}) \leq W_{\mathcal{I}}$ and $W_{\mathcal{I}} \leq twn$. □

Theorem 4.3 (Theorem 1.1 restated)**.** *Let Λ be a CSP and $\epsilon > 0$ be a small constant. Then, there is an algorithm such that, given an oracle access $\mathcal{O}_{\mathcal{I}}$ to a Λ-CSP instance $\mathcal{I}$ with $\mathbf{lp}(\mathcal{I}) = cW_{\mathcal{I}}$, it outputs an $(\alpha_\Lambda(c-\epsilon), \epsilon n)$-approximation to $\mathbf{opt}(\mathcal{I})$. The number of queries performed to $\mathcal{O}_{\mathcal{I}}$ is at most* $\exp(\exp(\mathrm{poly}(qstw/\epsilon)))$.

Proof. Let ϵ' be a constant chosen later and $(\boldsymbol{x}, \boldsymbol{\mu})$ be an ϵ'-infeasible $(1-\epsilon', \epsilon')$-approximate solution for $\mathcal{I}$. Suppose the folded instance $\mathcal{I}' = \mathcal{I}/\phi_{\boldsymbol{x}}$ on the variable set V'. Since there are at most $\exp(\mathrm{poly}(q/\epsilon'))$ variables in V', there are at most $N := \exp(\exp(\mathrm{poly}(q/\epsilon')))$ assignments to V'. For each assignment $\beta' : V' \to [q]$, we estimate the value $\mathbf{val}(\mathcal{I}', \beta')$ as follows. First, we note that β' can be unfolded to an assignment $\beta : V \to [q]$ to $\mathcal{I}$ with the same value. For each variable $v \in V$, we assign the value $f(v) = \sum_{P\ni v} P(\beta)/|P|$. It is clear that $0 \leq f(v) \leq tw$ and $\sum_{v\in V} f(v) = \mathbf{val}(\mathcal{I}, \beta) = \mathbf{val}(\mathcal{I}', \beta')$. Also, we can calculate the value $f(v)$ by querying $\mathcal{O}_{\mathbf{lp}}$ at most st times. Thus, using the algorithm given in Lemma 2.3, we get a $(1, \epsilon n/2)$-approximation to $\mathbf{val}(\mathcal{I}', \beta')$ with probability of at least $1 - 1/3N$ by querying $\mathcal{O}_{\mathbf{lp}}$ at most $O(\mathrm{poly}(stw/\epsilon)\log N)$ times.

By the union bound, with probability of at least $2/3$, we obtain a $(1, \epsilon n/2)$-approximation to $\mathbf{val}(\mathcal{I}', \beta')$ for every assignment β'. Taking the maximum of these values, we obtain $(1, \epsilon n/2)$-approximation to $\mathbf{opt}(\mathcal{I}')$. The number of queries to $\mathcal{O}_{\mathcal{I}}$ and $\mathcal{O}_{\mathbf{lp}}$ is at most $O(\mathrm{poly}(stw/\epsilon)N\log N)$.

Since we are concerned with $(\cdot, \epsilon n)$-approximation, we can safely assume that $W_{\mathcal{I}} \geq \epsilon n$. If not, $\mathbf{val}(\mathcal{I}, \beta)$ is indeed a $(1, \epsilon n)$-approximation to $\mathbf{opt}(\mathcal{I})$.

When $W_{\mathcal{I}} \geq \epsilon n$, it holds that

$$\begin{aligned}
\mathbf{val}(\mathcal{I},\beta) &\geq \mathbf{opt}(\mathcal{I}') - \frac{\epsilon n}{2} \geq \alpha_\Lambda\left(\frac{\mathbf{lp}(\mathcal{I}')}{W_{\mathcal{I}'}}\right)\mathbf{lp}(\mathcal{I}') - \frac{\epsilon n}{2} \\
&\geq \alpha_\Lambda\left(\frac{\mathbf{lp}(\mathcal{I}) - \epsilon'\cdot\mathrm{poly}(qstw)n}{W_{\mathcal{I}'}}\right)(\mathbf{lp}(\mathcal{I}) - \epsilon'\cdot\mathrm{poly}(qstw)n) - \frac{\epsilon n}{2} \quad \text{(using Lemma 4.2)} \\
&= \alpha_\Lambda\left(\frac{\mathbf{lp}(\mathcal{I})}{W_{\mathcal{I}}} - \frac{\epsilon'\cdot\mathrm{poly}(qstw)n}{W_{\mathcal{I}}}\right)(\mathbf{lp}(\mathcal{I}) - \epsilon'\cdot\mathrm{poly}(qstw)n) - \frac{\epsilon n}{2} \quad \text{(using } W_{\mathcal{I}} = W_{\mathcal{I}'}\text{)} \\
&\geq \alpha_\Lambda\left(\frac{\mathbf{lp}(\mathcal{I})}{W_{\mathcal{I}}} - \frac{\epsilon'\cdot\mathrm{poly}(qstw)}{\epsilon}\right)(\mathbf{lp}(\mathcal{I}) - \epsilon'\cdot\mathrm{poly}(qstw)n) - \frac{\epsilon n}{2} \quad \text{(using } W_{\mathcal{I}} \geq \epsilon n\text{)} \\
&\geq \alpha_\Lambda\left(\frac{\mathbf{lp}(\mathcal{I})}{W_{\mathcal{I}}} - \frac{\epsilon'\cdot\mathrm{poly}(qstw)}{\epsilon}\right)(\mathbf{opt}(\mathcal{I}) - \epsilon'\cdot\mathrm{poly}(qstw)n) - \frac{\epsilon n}{2}. \quad \text{(using } \mathbf{lp}(\mathcal{I}) \geq \mathbf{opt}(\mathcal{I})\text{)}
\end{aligned}$$

By setting $\epsilon' = \epsilon/\mathrm{poly}(qstw)$, we have the desired result. The number of queries performed to $\mathcal{O}_{\mathcal{I}}$ is at most

$$\mathrm{poly}(stw/\epsilon)N\log N\exp(\exp(\mathrm{poly}(qstw/\epsilon'))) = \exp(\exp(\mathrm{poly}(qstw/\epsilon))).$$

□

5 Robustness of BasicLP

In this section, we give a proof of Lemma 2.4. Our strategy is transforming $(\boldsymbol{x},\boldsymbol{\mu})$ to a feasible solution without decreasing the LP value much. In the first step, we construct $\boldsymbol{x}'$ from $\boldsymbol{x}$ that satisfies $\sum_{a\in[q]}\boldsymbol{x}'_{v,a} = 1$ for every $v \in V$.

Lemma 5.1. *Let $(\boldsymbol{x},\boldsymbol{\mu})$ be an ϵ-infeasible LP solution for a Λ-CSP instance $\mathcal{I}$ where $\epsilon > 0$ is a small constant. Then, $\boldsymbol{x}$ can be transformed to $\boldsymbol{x}'$ in such a way that*

$$\sum_{a\in[q]}\boldsymbol{x}'_{v,a} = 1 \quad \forall v \in V, \tag{4}$$

$$|\boldsymbol{x}'_{v,a} - \boldsymbol{x}_{v,a}| = 2\epsilon \quad \forall v \in V, a \in [q]. \tag{5}$$

In particular, $(\boldsymbol{x}',\boldsymbol{\mu})$ is a 3ϵ-infeasible LP solution that satisfies $\sum_{a\in[q]}\boldsymbol{x}'_{v,a} = 1$ for every $v \in V$.

Proof. We define $\boldsymbol{x}'_{v,a} = \boldsymbol{x}_{v,a}/\sum_{a\in[q]}\boldsymbol{x}_{v,a}$. The condition (4) clearly holds. From the ϵ-infeasibility of $\boldsymbol{x}$, $|\sum_{a\in[q]}\boldsymbol{x}_{v,a} - 1| \leq \epsilon$ holds. It follows that $|\boldsymbol{x}'_{v,a} - \boldsymbol{x}_{v,a}| \leq \epsilon/(1-\epsilon) \leq 2\epsilon$ (when ϵ is small). □

In the second step, we construct $\boldsymbol{\mu}'$ that satisfies $\sum_{\beta\in[q]^{V(P)},\beta_v=a}\boldsymbol{\mu}'_{P,\beta} = \boldsymbol{x}'_{v,a}$ for all $P \in \mathcal{P}, v \in V(P)$.

Lemma 5.2. *Let $(\boldsymbol{x},\boldsymbol{\mu})$ be an ϵ-infeasible solution for a Λ-CSP instance $\mathcal{I}$ satisfying $\sum_{a\in[q]}\boldsymbol{x}_{v,a}$ for every $v \in V$. Then, $\boldsymbol{\mu}$ can be transformed to $\boldsymbol{\mu}'$ in such a way that*

$$\begin{aligned}
\Pr_{\beta\sim\boldsymbol{\mu}'_P}[\beta_v = a] &= (1-\delta)\boldsymbol{x}_{v,a} + \delta\frac{1}{q} \quad \forall P \in \mathcal{P}, v \in V(P), a \in [q], \\
||\boldsymbol{\mu}_P - \boldsymbol{\mu}'_P||_1 &\leq 2\delta \quad \forall P \in \mathcal{P}.
\end{aligned}$$

where $\delta = kq^3\epsilon$.

Proof. Let us fix a predicate $P \in \mathcal{P}$ and $S = V(P)$. We may assume $S = \{1, \dots, k\}$ where $k \leq s$. We can think of $\boldsymbol{\mu}_P$ as a function $f : [q]^k \to \mathbb{R}$ such that $f(\beta)$ is the probability of the assignment β under the distribution $\boldsymbol{\mu}_P$.

Let $\chi_1, \dots, \chi_q$ be an orthonormal basis of the vector space $\{f : [q] \to \mathbb{R}\}$ such that $\chi_1 \equiv 1$. Here, orthonormal means that $E_{a \in [q]}[\chi_i(a)\chi_j(a)] = \delta_{ij}$ for all $i, j \in [q]$ where δ is Kronecker's delta. By tensoring this basis, we obtain the orthonormal basis $\{\chi_\rho\}_{\rho \in [q]^k}$ of the vector space $\{f : [q]^k \to \mathbb{R}\}$. That is, for $\rho \in [q]^k, \beta \in [q]^k$, we have $\chi_\rho(\beta) = \chi_{\rho_1}(\beta_1) \cdots \chi_{\rho_k}(\beta_k)$. For a function $f : [q]^k \to \mathbb{R}$, we define $\hat{f}(\sigma) = \sum_{\beta \in [q]^k} f(x)\chi_\sigma(\beta)$. Note that $f(\beta) = E_{\sigma \in [q]^k}[\hat{f}(\sigma)\chi_\sigma(\beta)]$. Therefore, if we let f again be the function corresponding to $\boldsymbol{\mu}_P$, we have

$$\Pr_{\beta \sim \boldsymbol{\mu}_P}[\beta_i = a] = \sum_{\beta \in [q]^k, \beta_i = a} \mathop{\mathrm{E}}_{\sigma \in [q]^k}\left[\widehat{f}(\sigma)\chi_\sigma(\beta)\right] = \mathop{\mathrm{E}}_{\sigma \in [q]}\left[\widehat{f_i}(\sigma)\chi_\sigma(a)\right].$$

Here, $\widehat{f_i}(s) = \widehat{f}(\sigma)$ where $\sigma_i = s$ and $\sigma_r = 1$ for all $r \in [k] \setminus \{i\}$. In the second inequality, we used that for every σ with $\sigma_r \neq 1$ for some $r \in [k] \setminus \{i\}$, the sum over the values of χ_σ vanishes.

We let $g_i : [q] \to \mathbb{R}$ be the function $g_i(a) = \boldsymbol{x}_{i,a}$. We define a function $f' : [q]^k \to \mathbb{R}$ as follows.

$$\widehat{f'}(\sigma) = \begin{cases} \widehat{g_i}(s) & \text{if } \sigma_i = s \text{ and } \sigma_r = 1 \text{ for all } r \in [k] \setminus \{i\}, \\ \widehat{f}(\sigma) & \text{otherwise.} \end{cases}$$

This is well-defined since for any $i \in [k]$, it holds that $\widehat{g_i}(1) = \sum_{a \in [q]} g_i(a) = \sum_{a \in [q]} \boldsymbol{x}_{i,a} = 1$. Therefore, the function f' satisfies $\sum_{\beta \in [q]^k} f'(\beta) = \widehat{f}(1) = 1$, Then, we can define a distribution $\boldsymbol{\mu}'_P$ corresponding to f', and we have

$$\Pr_{\beta \sim \boldsymbol{\mu}'_P}[\beta_i = a] = \mathop{\mathrm{E}}_{\sigma \in [q]}\left[\widehat{f'_i}(\sigma)\chi_\sigma(a)\right] = \boldsymbol{x}_{v,a}.$$

Thus, it looks that the $\boldsymbol{\mu}'_P$ is the desired distribution. However, in general, the function f' might take negative values. We will show that these values cannot be too negative and that the function can be made to a proper distribution by smoothing.

Let K be an upper bound on the values of the functions $\chi_1, \dots, \chi_q$. From the orthonormality of the functions, it follows that $K \leq \sqrt{q}$. Let $f_i(a) = \Pr_{\beta \sim \boldsymbol{\mu}_P}[\beta_i = a]$. Since the LP solution $(\boldsymbol{x}, \boldsymbol{\mu})$ is ϵ-infeasible, we have

$$\left|\widehat{g_i}(s) - \widehat{f_i}(s)\right| = \left|\sum_{a \in [q]} g_i(a)\chi_s(a) - \sum_{a \in [q]} f_i(a)\chi_s(a)\right| \leq Kq\epsilon.$$

Therefore, $|\widehat{f'}(\sigma) - \widehat{f}(\sigma)| \leq Kq\epsilon$ for all $\sigma \in [q]^k$. Recall that $|\widehat{f'}(\sigma) - \widehat{f}(\sigma)| = 0$ for $\sigma \in [q]^k$ if there are $i \neq j$ such that $\sigma_i \neq 1, \sigma_j \neq 1$. Thus,

$$|f'(\beta) - f(\beta)| = \left|\mathop{\mathrm{E}}_{\sigma \in [q]^k}\left[\widehat{f'}(\sigma)\chi_\sigma(\beta) - \widehat{f}(\sigma)\chi_\sigma(\beta)\right]\right| \leq \delta/q^k, \tag{6}$$

where $\delta = K^2kq^2\epsilon$. Hence, if we let $h = (1 - \delta)f' + \delta U$, where $U : [q]^k \to \mathbb{R}$ is the uniform distribution $U \equiv 1/q^k$, then

$$h(x) = (1 - \delta)f'(x) + \delta/q^k \geq (1 - \delta)f(x) \geq 0.$$

It follows that h corresponds to another distribution $\boldsymbol{\mu}'_P$ over assignments $[q]^k$. Furthermore, it holds

$$\Pr_{\beta\sim\boldsymbol{\mu}'_P}[\beta_i = a] = (1-\delta)\boldsymbol{x}_{i,a} + \frac{\delta}{q}.$$

Finally, let us estimate the statistical distance between the distributions $\boldsymbol{\mu}_P$ and $\boldsymbol{\mu}'_P$.

$$||f - h||_1 = ||(1-\delta)(f-f') + \delta(f-U)||_1 \leq ||f-f'||_1 + \delta \leq 2\delta.$$

The first inequality is from the triangle inequality and the second inequality is from (6). □

Proof of Lemma 2.4. Let us consider an ϵ-infeasible LP solution $(\boldsymbol{x}, \boldsymbol{\mu})$ for a Λ-CSP instance $\mathcal{I}$ of value $cW_{\mathcal{I}}$. First, we construct vector $\boldsymbol{x}'$ as in Lemma 5.1. These variables together with the original local distributions $\boldsymbol{\mu}$ form an 3ϵ-infeasible LP solution for $\mathcal{I}$. Next, we construct local distributions $\boldsymbol{\mu}'$ as in Lemma 5.2. Define new variables

$$\boldsymbol{x}''_{i,a} = (1-\delta)\boldsymbol{x}'_{i,a} + \delta/q.$$

It follows that $(\boldsymbol{x}'', \boldsymbol{\mu}')$ is a feasible LP solution for $\mathcal{I}$. The LP value of this solution is

$$\begin{aligned}
\sum_{P\in\mathcal{P}} \boldsymbol{w}_P \mathop{\mathrm{E}}_{\beta\sim\boldsymbol{\mu}'_P}[P(\beta)] &= cW_{\mathcal{I}} - \sum_{P\in\mathcal{P}} \boldsymbol{w}_P \sum_{\beta\in[q]^{V(P)}} P(\beta)\left(\boldsymbol{\mu}_{P,\beta} - \boldsymbol{\mu}'_{P,\beta}\right) \\
&\geq cW_{\mathcal{I}} - \sum_{P\in\mathcal{P}} \boldsymbol{w}_P ||\boldsymbol{\mu}_P - \boldsymbol{\mu}'_P||_1 \\
&\geq cW_{\mathcal{I}} - \epsilon\cdot\mathrm{poly}(kq)W_{\mathcal{I}}.
\end{aligned}$$

We used $|P(x)| \leq 1$ for the first inequality, and the second inequality follows from Lemma 5.2. □

6 Lower Bounds

In this section, we prove Theorem 1.2. As we described in the introduction, we utilize Yao's minimax principle to show lower bounds on the query complexity for approximating Λ-CSP. That is, we construct two distributions of instances such that they have much different optimal values and also it is hard to distinguish them in constant time. We fix a Λ-CSP instance $\mathcal{I} = (V, \mathcal{P}, \boldsymbol{w})$ with the optimal LP solution $(\boldsymbol{x}^*, \boldsymbol{\mu}^*)$ throughout this section, and let n and m be the number of variables and constraints in $\mathcal{I}$, respectively. To convert the LP integrality gap of $\mathcal{I}$ to hardness results, we construct two distributions $\mathcal{D}^{\mathbf{opt}}_{N,T}$ and $\mathcal{D}^{\mathbf{lp}}_{N,T}$ using $\mathcal{I}$ and $(\boldsymbol{x}^*, \boldsymbol{\mu}^*)$. Here, N and T will determine the number of variables and the maximum degree of instances generated by $\mathcal{D}^{\mathbf{opt}}_{N,T}$ and $\mathcal{D}^{\mathbf{lp}}_{N,T}$, respectively. We show that, by taking T as a large constant (irrespective of N), almost all instances $\mathcal{J} \in \mathcal{D}^{\mathbf{opt}}_{N,T}$ satisfy that $\mathbf{opt}(\mathcal{J})/W_{\mathcal{J}} \leq \mathbf{opt}(\mathcal{I}) + \epsilon$. Also, we show that all instances $\mathcal{J} \in \mathcal{D}^{\mathbf{lp}}_{N,T}$ satisfy that $\mathbf{opt}(\mathcal{J})/W_{\mathcal{J}} \geq \mathbf{lp}(\mathcal{I})$. Finally, we define $\mathcal{D}_{N,T}$ as the distribution that takes an instance from $\mathcal{D}^{\mathbf{opt}}_{N,T}$ with probability 1/2 and from $\mathcal{D}^{\mathbf{lp}}_{N,T}$ with probability 1/2. Then, given an oracle access $\mathcal{O}_{\mathcal{J}}$ to an instance $\mathcal{J}$ generated by $\mathcal{D}_{N,T}$, an algorithm is supposed to guess the original distribution ($\mathcal{D}^{\mathbf{opt}}_{N,T}$ or $\mathcal{D}^{\mathbf{lp}}_{N,T}$) of $\mathcal{J}$. By showing that such an algorithm requires $\Omega(\sqrt{N})$ queries, we conclude that any $(\mathbf{opt}(\mathcal{I})/\mathbf{lp}(\mathcal{I}) + \epsilon, \delta n)$-approximation algorithm requires $\Omega(\sqrt{N})$ queries. By choosing as $\mathcal{I}$ an instance with the worst integrality gap, we have the desired result.

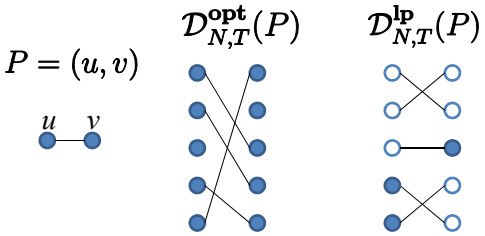

Figure 2: Construction of $\mathcal{D}^{\mathbf{opt}}_{N,T}(P)$ and $\mathcal{D}^{\mathbf{lp}}_{N,T}(P)$. Here, the alphabet size $q = 2$, and we choose $N = 5$ and $T = 1$. Also, $\boldsymbol{\mu}^*_{P,00} = 0.4$, $\boldsymbol{\mu}^*_{P,01} = 0.2$, $\boldsymbol{\mu}^*_{P,10} = 0.4$, and $\boldsymbol{\mu}^*_{P,11} = 0$. It follows that $\boldsymbol{x}^*_{u,0} = 0.6$, $\boldsymbol{x}^*_{u,1} = 0.4$, $\boldsymbol{x}^*_{v,0} = 0.8$, and $\boldsymbol{x}^*_{v,1} = 0.2$. White (resp., black) variables in $\mathcal{D}^{\mathbf{lp}}_{N,T}$ indicate that they are assigned to 0 (resp., 1).

Construction of $\mathcal{D}^{\mathbf{opt}}_{N,T}$: Before stating the construction of $\mathcal{D}^{\mathbf{opt}}_{N,T}$, we introduce a distribution $\mathcal{D}^{\mathbf{opt}}_{N,T}(P)$ for each constraint $P \in \mathcal{P}$ (see Fig. 2). An instance $\mathcal{J}_P$ of $\mathcal{D}^{\mathbf{opt}}_{N,T}(P)$ is generated as follows. Let $k = |V(P)|$ be the arity of P. Then, the variable set of $\mathcal{J}_P$ is $V(P) \times [N]$ and we regard that it consists of k parts. Next, we create TN constraints among those variables. To this end, we split each variable of $\mathcal{J}_P$ into T variables. Then, we take random perfect k-partite matchings in such a way that each matching takes one variable from each part. For each matching $\{v_1, \dots, v_k\}$, we create a copy of P on the variable set $\{v_1, \dots, v_k\}$ of weight $\boldsymbol{w}_P$. Finally, we merge the split variables again.

We define the distribution $\mathcal{D}^{\mathbf{opt}}_{N,T}$ using $\mathcal{D}^{\mathbf{opt}}_{N,T}(P)$. An instance $\mathcal{J}$ of $\mathcal{D}^{\mathbf{opt}}_{N,T}$ is generated as follows. For each $P \in \mathcal{P}$, we create an instance $\mathcal{J}_P$ according to the distribution $\mathcal{D}^{\mathbf{opt}}_{N,T}(P)$. Then, $\mathcal{J}$ is a union of $\{\mathcal{J}_P\}_{P \in \mathcal{P}}$ obtained by merging variable sets as follows. Let $P_1, \dots, P_k \in \mathcal{P}$ be the set of constraints containing a variable $v \in V$. We let $V_i (1 \le i \le k)$ denote the set of variables in $\mathcal{J}_{P_i}$ corresponding to v. Then, we take random perfect k-partite matchings among $V_1, \dots, V_k$ and we merge vertices in each matching. We repeat the same process for every $v \in V$. We note that the variable set of $\mathcal{J}$ is $V \times [N]$ and the number of constraints of $\mathcal{J}$ is $|\mathcal{P}|TN$. Now, we decide the indices of constraints, which are used as arguments of the oracle access $\mathcal{O}_{\mathcal{J}}$. We use the following rule. If P is the ith constraint where $v \in P$ appears (in the sense of $\mathcal{I}$), then for vertices $\{(v, j) \mid j \in [N]\} \subseteq V \times [N]$, we use indices $\{(T-1)i + 1, \dots, Ti\}$ to designate constraints in $\mathcal{J}_P$. Finally, the labels of vertices are randomly permuted.

Let $\mathcal{J}_P$ be an instance generated by $\mathcal{P}^N_T(P)$. Let $P_i (1 \le i \le 2)$ be a constraint on a variable set $\{u^i_1, \dots, u^i_k\}$ in $\mathcal{J}_P$. Note that the arities of P_i are the same since they both are copies of P. For each $j \in [k]$, we choose $v^1_j \in \{u^1_j, u^2_j\}$ arbitrarily and v^2_j be the remaining one, i.e., $\{u^1_j, u^2_j\} \setminus \{v^1_j\}$. Then, we define a constraint $Q_i (1 \le i \le 2)$ on the variable set $\{v^i_1, \dots, v^i_k\}$. We create another instance $\mathcal{J}'_P$ by replacing $\{P_1, P_2\}$ by $\{Q_1, Q_2\}$. We call this method *switching*. The following concentration bound holds from a simple modification of Theorem 2.19 in [28].

Lemma 6.1. *If $\mathbf{X}$ is a random variable defined on $\mathcal{D}^{\mathbf{opt}}_{N,T}(P)$ such that $|\mathbf{X}(\mathcal{J}_P) - \mathbf{X}(\mathcal{J}'_P)| \le c$ holds where $\mathcal{J}_P$ and $\mathcal{J}'_P$ are instances of $\mathcal{D}^{\mathbf{opt}}_{N,T}(P)$ that only differ by a switching, then*

$$\Pr_{\mathcal{J}_P \sim \mathcal{D}^{\mathbf{opt}}_{N,T}(P)} [|\mathbf{X}(\mathcal{J}_P) - \mathrm{E}[\mathbf{X}(\mathcal{J}_P)]| \ge t] \le 2 \exp\left(-\frac{t^2}{TNc^2}\right)$$

for all $t > 0$. □

Lemma 6.2. *For every ϵ, there is a $T > 0$ satisfying the following. Let $\mathcal{I} = (V, \mathcal{P}, \boldsymbol{w})$ be a Λ-CSP instance and $\mathcal{J}$ be an instance generated by $\mathcal{D}^{\mathbf{opt}}_{N,T}$. With probability $1 - o(1)$, $\mathbf{opt}(\mathcal{J})/W_{\mathcal{J}} \leq \mathbf{opt}(\mathcal{I})/W_{\mathcal{I}} + \epsilon$.*

Proof. Let $\alpha : V \times [N] \to [q]$ be an assignment to $\mathcal{J}$, and we define $\boldsymbol{x}_{v,a} = \#\{i \in [N] \mid \alpha(v,i) = a\}/N$ for $v \in V$ and $a \in [q]$. Note that $\boldsymbol{x}_v$ gives a probability distribution on assignments to a variable $v \in V$. We also define $\boldsymbol{\mu}_{P,\beta} = \prod_{v \in P} \boldsymbol{x}_{v,\beta_v}$ for $P \in \mathcal{P}$ and $\beta \in [q]^{V(P)}$.

For each $P \in \mathcal{P}$, let $\mathcal{J}_P$ be the sub-instance of $\mathcal{J}$ generated by $\mathcal{D}^{\mathbf{opt}}_{N,T}(P)$. The expectation (over $\mathcal{D}^{\mathbf{opt}}_{N,T}(P)$) of the value gained by a constraint P in $\mathcal{J}_P$ is $\mathrm{E}_{\beta_P \sim \boldsymbol{\mu}_P}[P(\beta_P)]$. Thus, it holds that

$$\begin{aligned}
\mathop{\mathrm{E}}_{\mathcal{J} \sim \mathcal{D}^{\mathbf{opt}}_{N,T}}[\mathbf{val}(\mathcal{J}, \alpha)] &= \sum_{P \in \mathcal{P}} \mathop{\mathrm{E}}_{\mathcal{J}_P \sim \mathcal{D}^{\mathbf{opt}}_{N,T}(P)}[\mathbf{val}(\mathcal{J}_P, \alpha_{|V(P)})] = TN \sum_{P \in \mathcal{P}} \boldsymbol{w}_P \mathop{\mathrm{E}}_{\beta_P \sim \boldsymbol{\mu}_P}[P(\beta_P)] \\
&= TN \sum_{P \in \mathcal{P}} \boldsymbol{w}_P \mathop{\mathrm{E}}_{\beta \sim \boldsymbol{\mu}}[P(\beta_{|V(P)})] = TN \mathop{\mathrm{E}}_{\beta \sim \boldsymbol{\mu}}[\sum_{P \in \mathcal{P}} \boldsymbol{w}_P P(\beta_{|V(P)})].
\end{aligned}$$

Note that $\mathrm{E}[\sum_{P \in \mathcal{P}} \boldsymbol{w}_P P(\beta_{|V(P)})]$ is the mean of values attained by several assignments. Thus,

$$\mathop{\mathrm{E}}_{\mathcal{J} \sim \mathcal{D}^{\mathbf{opt}}_{N,T}}[\mathbf{val}(\mathcal{J}, \alpha)] \leq TN\mathbf{opt}(\mathcal{I}). \tag{7}$$

Note that, for instances $\mathcal{J}_P$ and $\mathcal{J}'_P$ generated by $\mathcal{D}^{\mathbf{opt}}_{N,T}(P)$ such that they differ by a switching, $\mathbf{val}(\mathcal{J}_P, \alpha_{|V(P)})$ and $\mathbf{val}(\mathcal{J}'_P, \alpha_{|V(P)})$ can differ by at most $2\boldsymbol{w}_P \leq 2w$. Then, from Lemma 6.1,

$$\Pr\left[\left|\mathbf{val}(\mathcal{J}_P, \alpha_{|V(P)}) - \mathrm{E}[\mathbf{val}(\mathcal{J}_P, \alpha_{|V(P)})]\right| \geq t\right] \leq 2\exp\left(-\frac{t^2}{4TNw^2}\right).$$

Then,

$$\begin{aligned}
\Pr\left[|\mathbf{val}(\mathcal{J}, \alpha) - \mathrm{E}[\mathbf{val}(\mathcal{J}, \alpha)]| \geq tm\right] &\leq \Pr\left[\exists P \in \mathcal{P},\ |\mathbf{val}(\mathcal{J}_P, \alpha_{|V(P)}) - \mathrm{E}[\mathbf{val}(\mathcal{J}_P, \alpha_{|V(P)})]| \geq t\right] \\
&\leq 2m\exp\left(-\frac{t^2}{4TNw^2}\right).
\end{aligned}$$

The last inequality is from the union bound.

We choose $t = \epsilon TN$ so that $tm = \epsilon mTN \leq \epsilon W_{\mathcal{J}}$. We have

$$\Pr\Big[|\mathbf{val}(\mathcal{J}, \alpha) - \mathrm{E}[\mathbf{val}(\mathcal{J}, \alpha)]| \geq \epsilon W_{\mathcal{J}}\Big] \leq 2m\exp\left(-\frac{\epsilon^2 TN}{4w^2}\right). \tag{8}$$

We combine (7) and (8) with the union bound over all q^{nN} assignments. It holds that

$$\begin{aligned}
\Pr\Big[\exists \alpha, \mathbf{val}(\mathcal{J}, \alpha) \geq TN\mathbf{opt}(\mathcal{I}) + \epsilon W_{\mathcal{J}}\Big] &= \Pr\Big[\exists \alpha, \mathbf{val}(\mathcal{J}, \alpha)/W_{\mathcal{J}} \geq \mathbf{opt}(\mathcal{I})/W_{\mathcal{I}} + \epsilon\Big] \\
&\leq 2m\exp\left(-\frac{\epsilon^2 TN}{4w^2}\right) q^{nN}.
\end{aligned}$$

by choosing $T = \Theta(w^2 \log q/\epsilon^2)$, we have the desired result. Note that n can be seen as a constant when N is sufficiently large. □

Construction of $\mathcal{D}^{\mathbf{lp}}_{N,T}$: Before stating the construction of $\mathcal{D}^{\mathbf{lp}}_{N,T}$, again we introduce another distribution $\mathcal{D}^{\mathbf{lp}}_{N,T}(P)$ for each constraint $P \in \mathcal{P}$ (see Fig. 2). An instance $\mathcal{J}_P$ is generated as follows. The variable set of $\mathcal{J}_P$ is $V(P) \times [N]$ and we regard that it consists of k parts. For each $\beta \in [q]^{V(P)}$, we take a $\boldsymbol{\mu}^*_{P,\beta}$-fraction of variables from each part. We say that a variable of the form $(v, i) \in V(P) \times [N]$ in this set of variables is *assigned to* $\beta_v \in [q]$. In total, we take $k\boldsymbol{\mu}^*_{P,\beta}N$ variables from the whole variable set. We split each variable into T copies. Then, we take random perfect k-partite matchings in such a way that each matching takes one variable from each part. For each matching $\{v_1, \ldots, v_k\}$, we create a copy of P on the variable set $\{v_1, \ldots, v_k\}$ of weight $\boldsymbol{w}_P$. Finally, we merge the split variables again. We note that, for fixed $v \in V(P)$, an $\boldsymbol{x}^*_{v,a}$-fraction of variables of $\{(v, i) \mid i \in [N]\}$ is assigned to a. A subtlety here is that $\boldsymbol{\mu}_{P,\beta}N$ may not be an integer. Since we can make this error arbitrarily small by choosing N large enough, we ignore this issue for simplicity.

We define the distribution $\mathcal{D}^{\mathbf{lp}}_{N,T}$ using $\mathcal{D}^{\mathbf{lp}}_{N,T}(P)$. An instance $\mathcal{J}$ of $\mathcal{D}^{\mathbf{lp}}_{N,T}$ is generated as follows. For each $P \in \mathcal{P}$, we create an instance $\mathcal{J}_P$ according to the distribution $\mathcal{D}^{\mathbf{lp}}_{N,T}(P)$. Then, $\mathcal{J}$ is a union of $\{\mathcal{J}_P\}_{P\in\mathcal{P}}$ obtained by merging variable sets as follows. Let $P_1, \ldots, P_k \in \mathcal{P}$ be the set of constraints containing a variable $v \in V$. We let $V_{i,a} (1 \leq i \leq k, a \in [q])$ denote the set of variables in $\mathcal{J}_{P_i}$ that are corresponding to v and assigned to a. Then, we take random perfect k-partite matchings among $V_{1,a}, \ldots, V_{k,a}$ and we merge vertices in each matching. We repeat the same process for every $v \in V$ and $a \in [q]$. We note that the variable set of $\mathcal{J}$ is $V \times [N]$ and the number of constraints is $|\mathcal{P}|TN$. To decide the indices of constraints, we use the same rule as $\mathcal{D}^{\mathbf{opt}}_{N,T}$. Finally, the labels of vertices are randomly permuted.

Lemma 6.3. *Let $\mathcal{I} = (V, \mathcal{P}, \boldsymbol{w})$ be an Λ-CSP instance and $\mathcal{J}$ be an instance generated by $\mathcal{D}^{\mathbf{lp}}_{N,T}$. Then, $\mathbf{opt}(\mathcal{J})/W_{\mathcal{J}} \geq \mathbf{lp}(\mathcal{I})/W_{\mathcal{I}}$ holds.*

Proof. Let $\alpha : V \times [N] \to [q]$ be the natural assignment to variables in $\mathcal{J}$. From the construction,

$$\mathbf{val}(\mathcal{J}, \alpha) = \sum_{P\in\mathcal{P}} \mathbf{val}(\mathcal{J}_P, \alpha_{|V(P)}) = TN \sum_{P\in\mathcal{P}} \boldsymbol{w}_P \mathop{\mathrm{E}}_{\beta_P\sim\boldsymbol{\mu}^*_P} [P(\beta_P)] = TN\mathbf{lp}(\mathcal{I}).$$

□

Lemma 6.4. *Let $\mathcal{I} = (V, \mathcal{P}, \boldsymbol{w})$ be a Λ-CSP instance. Any deterministic algorithm that, given an oracle access to $\mathcal{O}_{\mathcal{J}}$ generated by $\mathcal{D}_{N,T}$, correctly guesses the original distribution of $\mathcal{J}$ with probability of at least $2/3$ requires at least $\Omega(\sqrt{N})$ queries.*

Proof. It is convenient to think that labels of variables are determined on the fly. Thus, $\mathcal{D}_{N,T}$ decides labels of variables at the time that the variable appears for the first time in the interaction between an algorithm and $\mathcal{D}_{N,T}$. The distribution never change by this modification. Also, we can think that the sequence of labels is determined beforehand, and for each time that a new variable appears, a new label for the variable is taken from the front of the sequence. Let $\mathcal{D}_{N,T,\ell}$ be the resulting distribution by fixing the sequence to ℓ in $\mathcal{D}_{N,T}$. It is clear that $\mathcal{D}_{N,T}$ coincides with the distribution that takes ℓ uniformly at random and uses $\mathcal{D}_{N,T,\ell}$. Thus, by showing that any deterministic algorithm for $\mathcal{D}_{N,T,\ell}$ requires $\Omega(\sqrt{N})$ queries, we have the desired result.

A deterministic algorithm $\mathcal{A}$ with query complexity τ can be expressed as a decision tree of depth at most τ. We transform $\mathcal{A}$ to a non-adaptive algorithm with τ queries. Recall that, from the rule of indices, if we fix an index, the oracle always returns a constraint that uses some particular predicate. Also, since we have fix the sequence of labels ℓ, at each node in the decision tree, there is just one branch corresponding to the case that $\mathcal{A}$ finds a constraint such that any variable in

the constraint is not seen by $\mathcal{A}$ before. We define the *transcript* as the set of variables appeared in the queries or the answers for them. We assume that $\mathcal{A}$ can output the correct answer for other branches, i.e., when the oracle returns a constraint containing a variable in the transcript at that time. This only improves the ability of $\mathcal{A}$. Ignoring branches for which $\mathcal{A}$ outputs an answer, the decision tree has the property that the number of children of each node is one. Thus, $\mathcal{A}$ is essentially a non-adaptive algorithm. Without loss of generality, we assume that $\mathcal{A}$ outputs that the instance is generated by $\mathcal{D}^{\mathbf{opt}}_{N,T}$ if $\mathcal{A}$ arrives at the leaf of the decision tree.

We assume that $\tau = o(N)$. In the ith query, the probability that a constraint containing a variable in the transcript is returned is at most $is^2T/(TN - isT)$ (the algorithm has seen at most is variables and they may touch isT constraints). By the union bound, the probability that $\mathcal{A}$ outputs the correct answer is at most

$$\sum_{i=1}^{\tau} \frac{is^2T}{TN - isT} + \frac{1}{2} \leq \frac{\tau^2 s^2}{N} + \frac{1}{2}.$$

To make this probability at least 2/3, we have to choose $\tau = \Omega(\sqrt{N})$. □

Proof of Theorem 1.2. Let us fix $c \in [0, 1]$ and $s = c\alpha_\Lambda(c)$. Then, there exists a Λ-CSP instance $\mathcal{I}$ such that $\mathbf{lp}(\mathcal{I}) = cW_\mathcal{I}$ and $\mathbf{opt}(\mathcal{I}) = sW_\mathcal{I}$. Suppose that there exists an $(\alpha_\Lambda(c) + \epsilon, \delta n)$-approximation algorithm $\mathcal{A}$ with query complexity $o(\sqrt{n})$, where $\delta > 0$ is a constant determined later. Let T be a constant given by Lemma 6.2 substituting $\epsilon = \epsilon_{\mathbf{opt}}$ where $\epsilon_{\mathbf{opt}} < \epsilon/2$ is a constant. We define $n_\mathcal{J}$ as the number of variables in an instance generated by $\mathcal{D}_{N,T}$. Note that $W_\mathcal{J} \leq wTn_\mathcal{J}$.

Let $\mathcal{J}$ be an instance generated by $\mathcal{D}^{\mathbf{opt}}_{N,T}$. Then, from Lemma 6.2, with probability of at least $(1-o(1))\cdot 2/3$, the output x by $\mathcal{A}$ given $\mathcal{J}$ satisfies that $x/W_\mathcal{J} \leq \mathbf{opt}(\mathcal{J})/W_\mathcal{J} \leq \mathbf{opt}(\mathcal{I})/W_\mathcal{I} + \epsilon_{\mathbf{opt}} \leq s + \epsilon_{\mathbf{opt}}$.

Let $\mathcal{J}$ be an instance generated by $\mathcal{D}^{\mathbf{lp}}_{N,T}$. Then, from Lemma 6.3, with probability of at least 2/3, the output x by $\mathcal{A}$ given $\mathcal{J}$ satisfies that $x/W_\mathcal{J} \geq (\alpha_\Lambda(c) + \epsilon)\mathbf{opt}(\mathcal{J})/W_\mathcal{J} - \delta n_\mathcal{J}/W_\mathcal{J} \geq (\alpha_\Lambda(c) + \epsilon)\mathbf{opt}(\mathcal{I})/W_\mathcal{I} - \delta n_\mathcal{J}/W_\mathcal{J} \geq (1 + \epsilon)s - \delta/wT = s + \epsilon_{\mathbf{lp}}$ where $\epsilon_{\mathbf{lp}} = \epsilon - \delta/wT$. Thus, by choosing δ small enough, we have $\epsilon_{\mathbf{lp}} \geq \epsilon/2$. Then, the probability that $\mathcal{A}$ correctly guesses the original distribution of $\mathcal{J}$ is at least

$$\frac{1}{2}(1 - o(1))\frac{2}{3} + \frac{1}{2}\frac{2}{3} = \frac{2}{3} - o(1).$$

By running $\mathcal{A}$ constant times and take the majority of outputs, the probability can be increased to at least 2/3. This contradicts Lemma 6.4. □

Acknowledgements

The author is grateful to Hiro Ito and Suguru Tamaki for valuable comments on an earlier draft of this paper.

References

[1] Noga Alon. On constant time approximation of parameters of bounded degree graphs, 2010. manuscript.

[2] Noga Alon, Wenceslas Fernandez de la Vega, Ravi Kannan, and Marek Karpinski. Random sampling and approximation of MAX-CSPs. *Journal of Computer and System Sciences*, 67(2):212–243, 2003.

[3] Noga Alon and Asaf Shapira. Testing satisfiability. In *Proc. of SODA 2002*, pages 645–654, 2002.

[4] Gunnar Andersson and Lars Engebretsen. Property testers for dense constraint satisfaction programs on finite domains. *Random Struct. Algorithms*, 21(1):14–32, 2002.

[5] Per Austrin. Balanced MAX 2-SAT might not be the hardest. In *Proc. of STOC 2007*, pages 189–197, 2007.

[6] Per Austrin and Elchanan Mossel. Approximation resistant predicates from pairwise independence. In *Proc. of CCC 2008*, pages 249–258, 2008.

[7] Andrej Bogdanov, Kenji Obata, and Luca Trevisan. A lower bound for testing 3-colorability in bounded-degree graphs. In *Proc. of FOCS 2002*, pages 93–102, 2002.

[8] Bernard Chazelle, Ronitt Rubinfeld, and Luca Trevisan. Approximating the minimum spanning tree weight in sublinear time. In *Proc. of ICALP 2001*, pages 190–200, 2001.

[9] Dimitris A. Fotakis and Paul G. Spirakis. Linear programming and fast parallel approximability, 1997. manuscript.

[10] Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 42(6):1115–1145, 1995.

[11] Oded Goldreich and Dana Ron. A sublinear bipartiteness tester for bounded degree graphs. *Combinatorica*, 19(3):335–373, 1999.

[12] Oded Goldreich and Dana Ron. Property testing in bounded degree graphs. *Algorithmica*, 32(2):302–343, 2008.

[13] Venkatesan Guruswami, Rajsekar Manokaran, and Prasad Raghavendra. Beating the random ordering is hard: Inapproximability of maximum acyclic subgraph. In *Proc. of FOCS 2008*, pages 573–582, 2008.

[14] Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001.

[15] David Karger, Rajeev Motwani, and Madhu Sudan. Approximate graph coloring by semidefinite programming. *J. ACM*, 45(2):246–265, 1998.

[16] Subhash Khot. On the power of unique 2-prover 1-round games. In *Proc. of STOC 2002*, pages 767–775, 2002.

[17] Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. Optimal inapproximability results for MAX-CUT and other 2-variable CSPs? In *Proc. of FOCS 2004*, pages 146–154, 2004.

[18] Subhash Khot and Assaf Naor. Sharp kernel clustering algorithms and their associated grothendieck inequalities. *CoRR*, abs/0906.4816, 2009.

[19] Fabian Kuhn, Thomas Moscibroda, and Roger Wattenhofer. The price of being near-sighted. In *Proc. of SODA 2006*, pages 980–989, 2006.

[20] Amit Kumar, Rajsekar Manokaran, Madhur Tulsiani, and Nisheeth K. Vishnoi. On the optimality of a class of LP-based algorithms. *CoRR*, abs/0912.1776, 2009.

[21] Michael Lewin, Dror Livnat, and Uri Zwick. Improved rounding techniques for the MAX 2-SAT and MAX DI-CUT problems. In *Proc. of IPCO 2002*, pages 67–82, 2002.

[22] Rajsekar Manokaran, Joseph (Seffi) Naor, Prasad Raghavendra, and Roy Schwartz. SDP gaps and UGC hardness for multiway cut, 0-extension, and metric labeling. In *Proc. of STOC 2008*, pages 11–20, 2008.

[23] Huy N. Nguyen and Krzysztof Onak. Constant-time approximation algorithms via local improvements. In *Proc. of FOCS 2008*, pages 327–336, 2008.

[24] Michal Parnas and Dana Ron. Approximating the minimum vertex cover in sublinear time and a connection to distributed algorithms. *Theor. Comput. Sci.*, 381(1-3):183–196, 2007.

[25] Prasad Raghavendra. Optimal algorithms and inapproximability results for every CSP? In *Proc. of STOC 08*, pages 245–254, 2008.

[26] Prasad Raghavendra and David Steurer. How to round any CSP. In *Proc. of FOCS 2009*, pages 586–594, 2009.

[27] Alex Samorodnitsky and Luca Trevisan. Gowers uniformity, influence of variables, and PCPs. In *Proc. of STOC 2006*, pages 11–20. ACM, 2006.

[28] Nick Wormald. Models of random regular graphs. In *Surveys in Combinatorics*, pages 239–298. Cambridge University Press, 1999.

[29] Yuichi Yoshida. Lower bounds on query complexity for testing bounded-degree CSPs, 2010. manuscript.

[30] Yuichi Yoshida, Masaki Yamamoto, and Hiro Ito. An improved constant-time approximation algorithm for maximum matchings. In *Proc. of STOC 2009*, pages 225–234, 2009.

Appendix

A Proof of Lemma 3.4

In this section, we give a proof of Lemma 3.4. We consider a more restricted form of a packing LP.

$$\begin{array}{ll} \max & \mathbf{1}^T \boldsymbol{z} \\ \text{s.t.} & A^T \boldsymbol{z} \leq \boldsymbol{c} \\ & \boldsymbol{z} \geq 0, \end{array} \tag{9}$$

where $A \in \mathbb{R}_+^{m\times n}$ is a non-negative matrix such that $a_{ji} = 0$ or $a_{ji} \geq 1$ for any $j \in [m], i \in [n]$, and $\boldsymbol{c} \in \mathbb{R}_+^n$ is a non-negative vector.

Define

$$c_{\max} = \max_i c_i, \quad \Gamma_p = \max_i \frac{c_{\max}}{c_i} \sum_{j=1}^{m} a_{ji}, \quad \Gamma_d = \max_j \sum_{i=1}^{n} a_{ji}.$$

Then, there is a distributed algorithm that solves this packing LP.

Lemma A.1 ([19])**.** *For sufficiently small $\epsilon > 0$, there exists a deterministic distributed algorithm that computes a feasible $(1-\epsilon, 0)$-approximate solution of LP (9) in $O(\log \Gamma_p \log \Gamma_d/\epsilon^4)$ rounds.* □

In order to apply Lemma A.1 to LP (3), we transform it to the form LP (9). Note that, in the objective function, the coefficient of $\boldsymbol{\mu}_{P,\beta}$ is $\boldsymbol{w}_P P(\beta) + 1$ and the coefficients of $\boldsymbol{x}_{v,a}, \overline{\boldsymbol{x}}_{v,a}, \overline{\boldsymbol{\mu}}_{P,\beta}$ are 1. Thus, by replacing $\boldsymbol{\mu}_{P,\beta}$ with $\boldsymbol{\mu}_{P,\beta}/(\boldsymbol{w}_P P(\beta) + 1)$, we obtain the following LP.

$$\begin{aligned}
\max \quad & \mathbf{1}^T(\boldsymbol{x} + \overline{\boldsymbol{x}} + \boldsymbol{\mu} + \overline{\boldsymbol{\mu}}) \\
\text{s.t.} \quad & \sum_{a\in[q]} \boldsymbol{x}_{v,a} \leq q - 1 + \epsilon && \forall v \in V \\
& \sum_{a\in[q]} \overline{\boldsymbol{x}}_{v,a} \leq 1 + \epsilon && \forall v \in V \\
& \boldsymbol{x}_{v,a} + \sum_{\beta\in[q]^{V(P)}, \beta_v = a} \frac{\boldsymbol{\mu}_{P,\beta}}{\boldsymbol{w}_P P(\beta)+1} \leq 1 + \epsilon && \forall P \in \mathcal{P}, v \in V(P), a \in [q] \\
& \overline{\boldsymbol{x}}_{v,a} + \sum_{\beta\in[q]^{V(P)}, \beta_v = a} \overline{\boldsymbol{\mu}}_{P,\beta} \leq q^{V(P)-1} + \epsilon && \forall P \in \mathcal{P}, v \in V(P), a \in [q] \\
& \boldsymbol{x}_{v,a} + \overline{\boldsymbol{x}}_{v,a} \leq 1, \quad \boldsymbol{x}_{v,a} \geq 0, \quad \overline{\boldsymbol{x}}_{v,a} \geq 0 && \forall v \in V \\
& \frac{\boldsymbol{\mu}_{P,\beta}}{\boldsymbol{w}_P P(\beta)+1} + \overline{\boldsymbol{\mu}}_{P,\beta} \leq 1, \quad \boldsymbol{\mu}_{P,\beta} \geq 0, \quad \overline{\boldsymbol{\mu}}_{P,\beta} \geq 0 && \forall P \in \mathcal{P}, \beta \in [q]^{V(P)}.
\end{aligned}$$

We multiply each constraint in order to make every coefficient in the LHS at least 1. Then, we have the following LP.

$$\begin{aligned}
\max \quad & \mathbf{1}^T(\boldsymbol{x} + \overline{\boldsymbol{x}} + \boldsymbol{\mu} + \overline{\boldsymbol{\mu}}) \\
\text{s.t.} \quad & \sum_{a\in[q]} \boldsymbol{x}_{v,a} \leq q - 1 + \epsilon && \forall v \in V \\
& \sum_{a\in[q]} \overline{\boldsymbol{x}}_{v,a} \leq 1 + \epsilon && \forall v \in V \\
& (w+1)\boldsymbol{x}_{v,a} + \sum_{\beta\in[q]^{V(P)}, \beta_v = a} \frac{(w+1)\boldsymbol{\mu}_{P,\beta}}{\boldsymbol{w}_P P(\beta)+1} \leq (1+\epsilon)(w+1) && \forall P \in \mathcal{P}, v \in V(P), a \in [q] \quad (10) \\
& \overline{\boldsymbol{x}}_{v,a} + \sum_{\beta\in[q]^{V(P)}, \beta_v = a} \overline{\boldsymbol{\mu}}_{P,x} \leq q^{V(P)-1} + \epsilon && \forall P \in \mathcal{P}, v \in V(P), a \in [q] \\
& \boldsymbol{x}_{v,a} + \overline{\boldsymbol{x}}_{v,a} \leq 1, \quad \boldsymbol{x}_{v,a} \geq 0, \quad \overline{\boldsymbol{x}}_{v,a} \geq 0 && \forall v \in V \\
& \frac{(w+1)\boldsymbol{\mu}_{P,\beta}}{\boldsymbol{w}_P P(\beta)+1} + (w+1)\overline{\boldsymbol{\mu}}_{P,\beta} \leq w+1, \quad \boldsymbol{\mu}_{P,\beta} \geq 0, \quad \overline{\boldsymbol{\mu}}_{P,\beta} \geq 0 && \forall P \in \mathcal{P}, \beta \in [q]^{V(P)}.
\end{aligned}$$

Proof of Lemma 3.4. Note that LP (10) is of the form LP (9). After a calculation, we have

$$c_{\max} = O(w + q^s), \quad \Gamma_p = O((s+t)w(w+q^s)), \quad \Gamma_d = O(wq^s).$$

We define the *degree* of a variable in an LP as the number of inequalities where the variable appears. Let Δ_p and Δ_d be the maximum degree of primal variables and dual variables, respectively. Here, we treat LP (10) as a dual formulation. We have

$$\Delta_p = O(q^s), \quad \Delta_d = O(s+t).$$

Applying the algorithm given in Lemma A.1 to LP (10), we obtain a distributed algorithm that calculates $(1-\epsilon, 0)$-approximate solution. The number of rounds is $O(\log \Gamma_p \log \Gamma_d/\epsilon^4)$. Note that, given a variable, we can simulate the computation of the distributed algorithm involved by the variable with $(\Delta_p \Delta_d)^r$ queries, where r is the number of rounds. Thus, the query complexity becomes

$$(\Delta_p \Delta_d)^{O(\log \Gamma_p \log \Gamma_d/\epsilon^4)} = \exp(\mathrm{poly}(qstw/\epsilon)).$$

□

ECCC ISSN 1433-8092

http://eccc.hpi-web.de