Perfect Sampling for (Atomic) Lovász Local Lemma

Kun He[*] Xiaoming Sun[†] Kewen Wu[‡]

Abstract

We give a Markov chain based perfect sampler for uniform sampling solutions of constraint satisfaction problems (CSP). Under some mild Lovász local lemma conditions where each constraint has a small number of forbidden local configurations, our algorithm is accurate and efficient: it outputs a *perfect* uniform random solution of the CSP and its expected running time is *quasilinear* in the number of variables of the CSP. Prior to our work, perfect samplers are only shown to exist for CSPs under much more restrictive conditions (Guo, Jerrum, and Liu, JACM'19).

Our algorithm is a natural combination of *bounding chains* (Huber, STOC'98; Haggstrom and Nelander, Scandinavian Journal of Statistics'99) and *state compression* (Feng, He, and Yin, STOC'21). The crux of our analysis is a simple *information percolation* argument which still allows us to achieve bounds[1] obtained by current best approximate samplers (Jain, Pham, and Vuong, ArXiv'21).

Previous related works either use intricate algorithms or needs sophisticated analysis or even both. Thus we view the simplicity of *both* our algorithm and analysis as a strength of our work.

1 Introduction

The *constraint satisfaction problem* (CSP) is one of the most important topics in computer science (both theoretically and practically). A CSP instance is a collection of constraints defined on a set of variables, and a solution to the instance is an assignment of variables that satisfies all the constraints. For each CSP instance, it is natural to ask the following questions:

- DECISION. Does the instance *have* a solution?
- SEARCH. If the CSP instance is satisfiable, can we *find* a solution efficiently?
- SAMPLING. If we can efficiently find a solution, can we efficiently *sample* a uniform random solution from the whole solution space?

These questions, each deepening one above, progressively enhance our understanding of the computational complexity of CSPs. One can easily imagine the hardness of fully resolving these broad questions. Thus, not surprisingly, despite enormous results centered around them, we only have partial answers. Here we mention those related to our work.

[*]Institute of Computing Technology, Chinese Academy of Sciences. Email: hekun.threebody@foxmail.com
[†]Institute of Computing Technology, Chinese Academy of Sciences. Email: sunxiaoming@ict.ac.cn
[‡]Department of EECS, University of California at Berkeley. Email: shlw_kevin@hotmail.com

[1]In current version we obtain matching bounds for Boolean k-CNF Formulas and q-coloring of k-uniform Hypergraphs which are the (arguably) most important CSP instances in sampling Lovász local lemma. In our full version, we will include a matching bound for the general atomic CSPs. See detailed discussion in Subsection 4.3.

The Decision Problem. A fundamental criterion for the existence of constraint satisfaction solutions is given by the famous Lovász local lemma (LLL) [EL75]. Interpreting the space of all possible assignments as a probability space and the violation of each constraint as a bad event, the local lemma provides a sufficient condition for the existence of an assignment to avoid all the bad events. This sufficient condition, commonly referred to as the *local lemma regime*, is characterized in terms of the violation probability of each constraint and the dependency relation among the constraints.

The Search Problem. The *algorithmic LLL* (also called *constructive LLL*) provides efficient algorithms to find a constraint satisfaction solution in the local lemma regime. Plenty of works have been devoted to this topic [Bec91, Alo91, MR98, CS00, Mos09, MT10, KM11, HSS11, HS17, HS19]. The Moser-Tardos algorithm [MT10] is a milestone along this line: it finds a satisfying assignment efficiently up to a sharp condition known as the Shearer's bound [She85, KM11].

The Sampling Problem. The *sampling LLL* asks for efficient algorithms to sample a uniform random constraint satisfaction solution from all the satisfying assignments in the local lemma regime. It serves as a standard toolkit for the probabilistic inference problem in graphical models [Moi19], and has many applications in the theory of computing, such as all-terminal network reliability [GJL19, GJ19, GH20]. With a better understanding of the decision and search problem, much attention has been devoted to the sampling LLL in recent years [GJL19, Moi19, GLLZ19, GGGY20, FGYZ20, FHY20, JPV20, JPV21]. Since this is also our focus, we elaborate it in the next subsection.

1.1 Sampling LLL

To state the long list of results on sampling LLL, we need the following notations. Given a CSP instance, let n denote the number of variables, m denote the number of constraints, k denote the number of variables in each constraint, q denote the number of values that each variable can take, d denote the maximum variable degree, Δ denote the maximum constraint degree, p denote the maximum probability that a constraint is violated, and N denote the maximum number of falsifying assignments for each constraint. A constraint is called *atomic* if it is violated by exactly one assignment to its variables. A CSP instance is called atomic if all of its constraints are atomic.

The sampling LLL turns out to be computationally more challenging than the algorithmic LLL. For example, for $\mathsf{Boolean}$ k-$\mathsf{CNF\ Formulas}$ the Moser-Tardos algorithm can efficiently find a solution if $k \gtrsim \log_2(d)$ (where $\gtrsim$ hides lower order terms). However, it is NP-hard to approximately sample a solution if $k \lesssim 2\log_2(d)$, even when the formula is monotone [BGG+19].

On the algorithmic side, most efforts are on the *approximate* sampling, where the output distribution is close to uniform under *total variation distance*. The breakthrough of Moitra [Moi19] shows k-CNF solutions can be sampled in time $n^{\mathsf{poly}(dk)}$ if $k \gtrsim 60\log_2(d)$, where they novelly use the algorithmic LLL to *mark/unmark* variables and then convert the problem into solving linear programs of size $n^{\mathsf{poly}(dk)}$. We remark that this algorithm is deterministic if we only need a multiplicative approximation of the number of satisfying assignments, which is another topic closely related with approximate sampling [JVV86a]. Moitra's method has been successfully applied to hypergraph colorings [GLLZ19] and random CNF formulas [GGGY20][2].

[2][GGGY20] only provides an approximate counting algorithm for random CNF formulas. But with a close inspection, their algorithm can be turned to do approximate sampling. This is due to standard reductions and noticing fixing *bad variables* (defined in [GGGY20]) does not influence their (deterministic) algorithm.

Recently, a much faster algorithm for sampling solutions of k-CNF is given in [FGYZ20], which implements a Markov chain on the assignments of the marked variables chosen via Moitra's method. The resulting sampling algorithm has a near linear running time $\widetilde{O}\left(d^2k^3n^{1.001}\right)$ together with an improved regime $k \gtrsim 20\log_2(d)$, where $\widetilde{O}(\cdot)$ hides polylogarithmic factors. We also remark that this algorithm is inherently randomized even if we move to approximate counting.

This *nonadapive* mark/unmark approach seems to only work for the Boolean variables, where each variable has two possible values. To extend the approach to more general CSPs, Feng, He, and Yin [FHY20] introduced *states compression*, which considerably expands the applicability of the method used in [FGYZ20]. Their sampling algorithm runs in time $\widetilde{O}\left(\left(\Delta^2N^2k+q\right)n^{1.001}\right)$ if $p(\Delta N)^{350} \lesssim 1$. This algorithm is limited to the special case of CSPs where each constraint is violated by a small number of local configurations (i.e., N is small).

Recently, Jain, Pham, and Vuong [JPV20], shaving the dependency on N, provides a sampling algorithm with running time $n^{\mathsf{poly}(\Delta,k,\log(q))}$ when $p\Delta^7 \lesssim 1$. They revisit Moitra's mark/unmark framework and use it in an *adaptive* way. This is the first polytime algorithm (assuming $\Delta, k, q = O(1)$) for general CSPs under LLL-type conditions. By a novel (but highly sophisticated) information percolation argument, they [JPV21] also prove that the sampling algorithm in [FHY20] runs in time $\widetilde{O}\left(\left(\Delta^2N^2k\right)n^{1.001}\right)$ if $p(\Delta N)^{7.043} \lesssim 1$.

Method	k-CNF	Coloring	General CSPs	Time
[Moi19]	$k \gtrsim 60\log(d)$			$n^{\mathsf{poly}(dk)}$
[GLLZ19]		$q \gtrsim \Delta^{\frac{16}{k-16/3}}$		$n^{\mathsf{poly}(dk)}$
[FGYZ20]	$k \gtrsim 20\log(d)$			$\widetilde{O}\left(d^2k^3n^{1.001}\right)$
[FHY20]	$k \gtrsim 13\log(d)$	$q \gtrsim \Delta^{\frac{9}{k-12}}$	$p(\Delta N)^{350} \lesssim 1$	$\widetilde{O}\left(\left(\Delta^2N^2k+q\right)n^{1.001}\right)$
[JPV20]	$k \gtrsim 7\log(d)$	$q \gtrsim \Delta^{\frac{7}{k-4}}$	$p\Delta^7 \lesssim 1$	$n^{\mathsf{poly}(dk)}$
[JPV21]	$k \gtrsim 5.741\log(d)$	$q \gtrsim \Delta^{\frac{3}{k-4}}$	$p(\Delta N)^{7.043} \lesssim 1$	$\widetilde{O}\left(\left(\Delta^2N^2k\right)n^{1.001}\right)$

Table 1: Approximate sampling algorithms in the local lemma regime.

Table 1 summarizes the efficient regimes of these algorithms. We emphasize that all these sampling results, via standard reductions [JVV86b, ŠVV09], also imply efficient algorithms for (random) approximate counting, which estimates the number of satisfying assignments with some multiplicative error. In addition, for algorithms using Moitra's linear programming approach [Moi19, GLLZ19, GGGY20, JPV20], their approximate counting counterparts are deterministic. For the approaches using Markov chains [FGYZ20, FHY20, JPV21], the running time of their approximate counting counterparts is $\widetilde{O}(m \cdot T)$, where T is the running time of the corresponding approximate sampling algorithm and m is the number of constraints.

Though much progress has been made for the *approximate* sampling, much less are known for the *perfect* sampling. As far as we know, the only result on the perfect sampling in the local lemma regime is due to Guo, Jerrum, and Liu [GJL19], which provides a perfect sampler for the *extremal* CSPs where any two constraints sharing common variables cannot be violated simultaneously by the same assignment. It is still unclear how to perform perfect sampling through approximate sampling/counting in the local lemma regime using standard reductions [JVV86b].

Meanwhile, perfect sampling is an important topic in theoretical computer science. Plenty of works have been devoted to the study of perfect samplers [JVV86b, HN99, Hub98, Hub04, BC20, JSS20, Fil97, FMMR00, ACG12, FVY19]. Apart from its mathematical interest, one advantage of perfect sampler over approximate sampler is that the quality of the output of perfect samplers is never in question. In contrast, some satisfying assignments may never be outputted by an approximate sampler[3] which is indeed the case for [FGYZ20, GGGY20]. Another advantage comes from implementation. When the mixing time of the Markov chain is not rigorously analyzed, it is not sure when to stop a Markov chain based approximate sampler; while if the perfect sampler is implemented using techniques like *coupling from the past* [PW96], it always gives desired distribution when it stops even if we may not know any bounds on its expected running time. This is particularly important for practical purposes and heuristic algorithms: correctness comes before efficiency.

1.2 Our Results

In this paper, we provide perfect samplers for the solutions of atomic CSPs in the local lemma regime. Recall the notations $d, k, p, \Delta, q, n, m, N$ defined in Subsection 1.1.

Theorem 1.1 (Informal)**.** *There exists a Las Vegas algorithm such that the following holds. Given an atomic CSP instance Φ satisfying $p\Delta^C \lesssim 1$ where C is a constant determined by k, d, Δ, q, the algorithm outputs a uniform random satisfying solution of Φ in expected time $O\left(kqd^2\Delta^2 \cdot n\log(n)\right)$.*

The formal statement, with explicit conditions for parameters, is given in Theorem 3.2. Applying Theorem 1.1 to special instances, we can obtain perfect samplers for q-coloring of k-uniform Hypergraphs and Boolean k-CNF Formulas. These results match current best bounds from polytime approximate samplers.

Corollary 1.2 (See Theorem 4.2 for detail)**.** *There exists a Las Vegas algorithm such that the following holds. Let $H = (V, E)$ be a k-uniform hypergraph.*[4] *If $d \le q^{(1/3-o(1))k}$ for positive integers k and q large enough, then the algorithm outputs a uniform random q-coloring of H in expected time $O\left(kqd^2\Delta^2 \cdot |V|\log(|V|)\right)$.*

Corollary 1.3 (See Theorem 4.4 for detail)**.** *There exists a Las Vegas algorithm such that the following holds. Let Φ be a Boolean k-CNF formula on n variables.*[5] *If $d \le 2^{0.175k}$ for positive integer k large enough, then the algorithm outputs a uniform random satisfying assignment of Φ in expected time $O\left(kqd^2\Delta^2 \cdot n\log(n)\right)$.*

For general atomic CSPs, our algorithm and analysis need some modification (see Subsection 4.3 for detail) which will be included in the full version. For general non-atomic CSP, we can also obtain a perfect sampler by decomposing the non-atomic constraints to atomic ones as in [FHY20], which replaces Δ with $\Delta \cdot N$ in the final bounds. The algorithm is still efficient if each constraint has only a small number of forbidden local configurations.

1.3 Proof Overview

To illustrate our main idea, we first focus on Boolean k-CNF Formulas:

- there are n Boolean variables and m constraints,
- each constraint is a clause depending on exactly k variables and has exactly one local forbidden assignment,
- each variable appears in at most d constraints.

[3] Consider two distributions $\mathcal{D}_1$ and $\mathcal{D}_2$ where $\mathcal{D}_1$ is uniform over $\{1, 2, \ldots, n\}$ and $\mathcal{D}_2$ is uniform over $\{\sqrt{n}+1, \sqrt{n}+2, \ldots, n\}$. Then the total variation distance between $\mathcal{D}_1$ and $\mathcal{D}_2$ is only $1/\sqrt{n} = o(1)$.

[4] Here d is the maximum vertex degree and Δ is the maximum edge degree.

[5] Here d is the maximum number of clauses that intersect a common variable, and Δ is the maximum number of clauses that intersect a common clause.

For example, $(x_1 \vee x_2 \vee \neg x_3) \wedge (x_1 \vee x_5 \vee x_7) \wedge (x_2 \vee \neg x_4 \vee \neg x_6)$ is a Boolean 3-CNF formula where $n = 6, m = 3$ and $k = 3, d = 2$.

For a k-CNF, the first step of our algorithm is to mark variables which ensures every clause has a certain amount of marked and unmarked variables. Let V the the set of variables and $M \subseteq V$ be the set of marked variables. Then by the local lemma (Theorem 2.4), for any $\sigma \in \{0,1\}^M$ and any $v \in M$ the following two distributions are close under total variation distance:

- an unbiased coin in $\{0, 1\}$;
- the distribution of $\sigma'(v)$ where $\sigma' \in \{0,1\}^V$ is a uniform random satisfying assignment conditioning on $\sigma'(M \setminus \{v\}) = \sigma(M \setminus \{v\})$.

We call this *local uniformity* (Lemma 3.11). This step is similar to Moitra's algorithm [Moi19].

To sample a solution *approximately*, all previous works [FGYZ20, FHY20, JPV21] simulate an idealized Glauber dynamics P_{Glauber} (Algorithm 6) on the assignments of the marked variables as follows:

(A) Initialize $\sigma(v) \in \{0, 1\}$ arbitrarily for each $v \in M$.

(B) Going forward from time 0 to $T \to +\infty$, let v_t be the variable selected at time $t \geq 0$. Then update $\sigma(v_t) \leftarrow \sigma'(v_t)$, where $\sigma' \in \{0,1\}^V$ is a uniform random satisfying assignment conditioning on $\sigma'(M \setminus \{v_t\}) = \sigma(M \setminus \{v_t\})$.

(C) After Step (A) and (B), we extend σ to unmarked variables $V \setminus M$ by sampling a uniform random satisfying assignment conditioning on $\sigma(M)$.

To sample a solution perfectly, we simulate *bounding chains* P_{BChains} (Algorithm 4) of the idealized Glauber dynamics as follows: The algorithm guarantees at any point, each variable $v \in M$ is assigned with a value in $\{0, 1, \star\}$ where $\star$ means uncertainty.

(1) Initialize $\sigma(v) = \star$ for each $v \in M$ at a starting time $-T - 1 < 0$.

(2) Going forward from time $-T$ to -1, let v_t be the variable selected at time $-T \leq t < 0$. To update $\sigma(v_t)$, the algorithm iteratively finds all clauses that are not yet satisfied by $\sigma(M \setminus \{v_t\})$ and is connected with v_t (Algorithm 3).

   - If all marked variables connected to v_t have value 0 or 1, then we say v_t is *coupled* which can be updated by rejection sampling on v_t and unmarked variables (Algorithm 1). In this case $\sigma(v_t)$ is always updated to 0 or 1.
   - Otherwise, $\sigma(v_t)$ is updated based on the local uniformity, which may be assigned to $\star$ with a small probability (See the $\mathsf{SafeSampling}$ subroutine in Algorithm 4).

(3) After Step (1) and (2),

   - if there exists some marked variable with assignment $\star$, then we double T and re-run Step (1) and (2); (We remark that the randomness is reused. That is, the randomness used for time $t < 0$ is the same one regardless of T.)
   - otherwise we stop and extend σ to unmarked variables $V \setminus M$ by sampling a uniform random satisfying assignment conditioning on $\sigma(M)$.

To simplify the analysis of the algorithm, we implement P_{Glauber} and P_{BChains} with systematic scan rather than random scan [HDSMR16]. Specifically, at time $t \in \mathbb{Z}$ the algorithm always updates the variable with index $(t \mod m)$ (Algorithm 6) where $m = |M|$.

Let μ_M be the marginal distribution of the satisfying assignments on the marked variables M. Our goal is to prove the following claims for P_{BChains}:

1. When we stop at Step (3), the assignment on M has distribution μ_M (Subsection 3.3).
2. In expectation, we stop with $T = O(m \log(m))$ (Subsection 3.2.1 and Subsection 3.2.2).
3. In expectation, each update step in P_{BChains} is efficient.

Proof of Item 1. Firstly we show P_{Glauber} converges to μ_M in Step (B) when $T \to +\infty$. Though it is a time inhomogeneous Markov chain, we are able to embed it into a time homogeneous Markov chain P' by viewing $|M|$ consecutive time stamps as one step. Then we show P' is aperiodic and irreducible with unique stationary distribution μ_M. After that, we unpack P' back to P_{Glauber} to show it also converges to μ_M (Lemma 3.26).

Next, we prove Item 1 where we use the idea of *coupling from the past* [PW96] and *bounding chains* [Hub98, HN99].

- Coupling from the Past. Observe that for any positive integer L if we run P_{Glauber} from $-L \cdot m$ to -1, it has the *same* distribution as we run it from 0 to $L \cdot m - 1$. Thus by the argument above, Step (B) also has distribution μ_M if we run P_{Glauber} from time $-\infty$ to -1.
- Bounding Chains. The second key observation is, for each $t \in \mathbb{Z}$, if the assignment in P_{BChains} has no $\star$, then the update process is exactly P_{Glauber}. This means P_{BChains} is a *coupling* of P_{Glauber} (Lemma 3.29). On the other hand, we use $\star$ to denote uncertainty which incorporates all possible assignments that we need to couple. Thus when P_{BChains} stops at time T with $\widehat{\sigma} \in \{0,1\}^M$ at Step (3), any initial assignment of P_{Glauber}, updated from time $-T$ to -1, converges to $\widehat{\sigma}$.

Combining the two observations above, we know $\widehat{\sigma}$ has distribution exactly μ_M as desired.

Proof of Item 2. To upper the round T, we employ the *information percolation* argument similarly used in [LS16, JPV21].

The crucial observation is the following. Once $\sigma(v_{t_0})$ is updated to $\star$ at time t_0, then at this point there must be some variable $u \neq v_{t_0}$ assigned with $\star$ and connected to v_{t_0}. Let t_1 be the last update time of u before t_0, and thus $u = v_{t_1}$. Then we can find another variable $u' \neq v_{t_1}$ assigned with $\star$ and connected to v_{t_1} at time t_1. Continuing this process until we reach the initialization phase, we will find a list of time $0 > t_0 > t_1 > \cdots > t_\ell \geq -T$ such that for each time t_i,

- $\sigma(v_{t_i})$ is updated to $\star$,
- v_{t_i} is connected to $v_{t_{i+1}}$ and $\sigma(v_{t_{i+1}}) = \star$.

Therefore, we define the extended constraint (e, C) (Definition 3.13), where C is a clause and $e = \{t'_1, \ldots, t'_k\} \subseteq \{-T, \ldots, -1\}$ is a time sequence such that

- C depends on $v_{t'_1}, \ldots, v_{t'_k}$,
- $t'_1, \ldots, t'_k$ are succinct rounds of update for each $v_{t'_1}, \ldots, v_{t'_k}$.

Since each v_{t_i} and $v_{t_{i+1}}$ are connected at time t_i as we discussed above, it means we are able to find extended constraints $(e^i_1, C^i_1), \ldots, (e^i_{s_i}, C^i_{s_i})$ such that $v_{t_i} \in e^i_1$ and $v_{t_{i+1}} \in e^i_{s_i}$ and $e^i_1, \ldots, e^i_{s_i}$ are connected hyperedges over $\{-T, \ldots, -1\}$. Thus all the hyperedges $\left\{e^i_j\right\}, i \in \{0, \ldots, \ell - 1\}, j \in [s_i]$ forms a connected hypergraph H on vertex set $\{-T, \ldots, -1\}$.

Since each extended constraint (e, C) represents succinct rounds of update for variables in C, e has range less than $m = |M|$, i.e., $\max_{t_1, t_2 \in e} |t_1 - t_2| < m$. Thus the longest path $\mathcal{P}$ in H has length $|\mathcal{P}| = \Omega(T/m)$. On the other hand, Step (2) of P_{BChains} only finds clauses that are *not* satisfied, which means each fixed extended constraint (e, C) appears in H with extremely low (roughly 2^{-k}) probability.

Putting everything together, when we do not stop at round T. We should find a path $\mathcal{P}$ of extended constraints of length $|\mathcal{P}| = \Omega(T/m)$. Meanwhile, each fixed extended constraint is found with probability at most roughly 2^{-k}. Thus any fixed $\mathcal{P}$ exists with probability at most $2^{-k\cdot|\mathcal{P}|/2}$, since extended constraints in odd positions of $\mathcal{P}$ do not overlap[6] in H and there are $|\mathcal{P}|/2$. Moreover, it is easy to see each extended constraint overlaps with $O(k^2d)$ many other extended constraints, which provides an upper bound $O(k^2d)^{|\mathcal{P}|}$ for the number of possible $\mathcal{P}$. By union bound, the probability that we do not stop at round T (Coalescence Part of Proposition 3.5). is roughly

$$m \cdot \left(\frac{k^4d^2}{2^k}\right)^{\Omega(T/m)} \approx m2^{-T/m},$$

where we assume $k^4d^2 \ll 2^k$ and the additional m comes from choosing $t_0 \in \{-1, \ldots, -m\}$, i.e., the last update resulting in $\star$.

Comparing to the argument in [JPV21], our argument is much simpler. One main reason is that we use systematic scan instead of random scan, which makes the updates of each variable well behave through time. Moreover, our main data structure, extended constraint (Definition 3.13), is also much simpler than the discrepancy check used in their argument. We remark that our analysis can be used for approximate samplers as well after switching the random scan order to systematic scan in previous papers.

Proof of Item 3. To bounded the expected running time of each step, we prove concentration bounds on the number of clauses found for each Step (2) of P_{BChains} (Lemma 3.12). This follows the same 2-tree argument (a clever union bound method) as in [Moi19, FGYZ20]. Then the running time of the step (Fact 2.6 and Lemma 3.4) is a geometric distribution with expectation controlled by the local uniformity.

We remark that the total running time of P_{BChains} depends on both T and each update step, where they can be arbitrarily correlated. Thus we need to calculate the second moment of the running time and apply Cauchy-Schwarz inequality (Efficiency Part of Proposition 3.5).

From k-CNF to General Atomic CSPs. To generalize P_{Glauber} and P_{BChains} to variables with larger domains, we use the *state compression* technique [FHY20]. In one word, the mark/unmark procedure is replaced with projections for each variable; and then P_{Glauber} and P_{BChains} will first obtain a random partial assignment after the projections, and in the end, complete it to a uniform satisfying assignment. See Algorithm 4 for detail.

Though this modification provides *some* bound for general atomic CSPs, its bound deteriorates as variable domains grow unevenly larger or the number of dependent variables in constraints grows unevenly larger. This can be fixed with some tweak on the algorithm and analysis which we discuss in Subsection 4.3. On the other hand, we emphasize that the current version of algorithms and analysis already covers many important applications, including the Boolean k-CNF Formulas and q-coloring of k-uniform Hypergraphs.

Organization. We give formal definitions in Section 2. Our algorithm and its analysis are provided in Section 3. We discuss our result for different applications in Section 4.

[6] We say extended constraints (e_1, C_1) and (e_2, C_2) overlap iff $e_1 \cap e_2 \neq \emptyset$.

2 Preliminaries

We use $\log(\cdot)$ and $\ln(\cdot)$ to denote the logarithm with base 2 and e respectively. We use $[N]$ to denote $\{1,2,\ldots,N\}$; and use $\mathbb{Z}$ to denote the set of all integers. We say V is a *disjoint union* of $(V_i)_{i\in[s]}$ if $V=\bigcup_{i\in[s]} V_i$ and $V_i\cap V_j=\emptyset$ holds for any distinct $i,j\in[s]$. For positive integer m, $t \mod m = t - m\cdot\lfloor t/m\rfloor$ for non-negative integer t; and $t \mod m = t\cdot(1-m) \mod m$ for negative integer t.

For any index set I and domains $(\Omega_i)_{i\in I}$, we use $\prod_{i\in I}\Omega_i$ to denote their product space. For some vector $\mathsf{vec}\in\prod_{i\in I}\Omega_i$, we use $\mathsf{vec}(i)\in\Omega_i$ to denote the entry of vec indexed by i; and use $\mathsf{vec}(J)\in\prod_{i\in J}\Omega_i$ to denote the entries of vec on indices $J\subseteq I$.

For a finite set $\mathcal{X}$ we use $x\sim\mathcal{X}$ to denote that x is a random variable sampled *uniformly* from $\mathcal{X}$. For two events $\mathcal{E}_1,\mathcal{E}_2$ with $\mathbf{Pr}[\mathcal{E}_2]=0$, we define the conditional probability $\mathbf{Pr}[\mathcal{E}_1(x)\mid\mathcal{E}_2(x)]=0$. We say event $\mathcal{E}$ happens *almost surely* if $\mathbf{Pr}[\mathcal{E}]=1$.

Constraint Satisfaction Problems. Let V be a set of variables with finite domains $(\Omega_v)_{v\in V}$. A *constraint* C on V is a mapping $C\colon\prod_{v\in V}\Omega_v\to\{\mathsf{True},\mathsf{False}\}$. We say C depends on $v\in V$ if there exists $\sigma_1,\sigma_2\in\prod_{v\in V}\Omega_v$ such that $C(\sigma_1)\neq C(\sigma_2)$ and σ_1,σ_2 differ in (and only in) v. We use $\mathsf{vbl}(C)$ to denote the set of variables that C depends on, then C can be viewed as a mapping from $\prod_{v\in\mathsf{vbl}(C)}\Omega_v$ to $\{\mathsf{True},\mathsf{False}\}$.

For convenience we use $\sigma^C_{\mathsf{False}}\subseteq\prod_{v\in\mathsf{vbl}(C)}\Omega_v$ (resp., $\sigma^C_{\mathsf{True}}\subseteq\prod_{v\in\mathsf{vbl}(C)}\Omega_v$) to denote the set of falsifying (resp., satisfying) assignments of C. More generally, for $\mathcal{C}$ being a set of constraints, we use $\sigma^{\mathcal{C}}_{\mathsf{False}}$ (resp., $\sigma^{\mathcal{C}}_{\mathsf{True}}$) to denote the set of falsifying (resp., satisfying) assignments of $\mathcal{C}$, i.e., $C(\sigma)=\mathsf{False}$ for all $\sigma\in\sigma^{\mathcal{C}}_{\mathsf{False}}$ and *some* $C\in\mathcal{C}$ (resp., $C(\sigma)=\mathsf{True}$ for all $\sigma\in\sigma^{\mathcal{C}}_{\mathsf{True}}$ and *all* $C\in\mathcal{C}$).

Definition 2.1 ((Atomic) Constraint Satisfaction Problem)**.** A *constraint satisfaction problem* is specified by $\Phi=\left(V,(\Omega_v)_{v\in V},\mathcal{C}\right)$ where $\mathcal{C}$ is a set of constraints on V. We say Φ is *atomic* if $\left|\sigma^C_{\mathsf{False}}\right|=1$ holds for all $C\in\mathcal{C}$. In this case, we abuse the notation to define $\sigma^C_{\mathsf{False}}$ as the unique falsifying assignment of C.

In addition, we define the following measures of a CSP Φ:

- the *width* is $k=k(\Phi)=\max_{C\in\mathcal{C}}|\mathsf{vbl}(C)|$;
- the *variable degree* is $d=d(\Phi)=\max_{v\in V}|\{C\in\mathcal{C}\mid v\in\mathsf{vbl}(C)\}|$;
- the *constraint degree* is $\Delta=\Delta(\Phi)=\max_{C\in\mathcal{C}}|\{C'\in\mathcal{C}\mid\mathsf{vbl}(C)\cap\mathsf{vbl}(C')\neq\emptyset\}|$;[7]
- the *maximal individual falsifying probability* is $p=p(\Phi)=\max_{C\in\mathcal{C}}\frac{\left|\sigma^C_{\mathsf{False}}\right|}{\prod_{v\in\mathsf{vbl}(C)}|\Omega_v|}$.

We will simply use k,d,Δ,p when Φ is clear from the context. In addition we assume $\Delta\geq 2$, $d\geq 2$, and $|V|\geq 2$ since otherwise the constraints in Φ are independent and the sampling problem becomes trivial.

Projections. A *projection* π from domain Ω to domain Q is a mapping $\pi\colon\Omega\to Q$.

Let $\Phi=\left(V,(\Omega_v)_{v\in V},\mathcal{C}\right)$ be a constraint satisfaction problem. Let $\boldsymbol{\pi}=(\pi_v)_{v\in V}$ where each $\pi_v\colon\Omega_v\to Q_v$ is a projection from Ω_v to some domain Q_v. Then for any $\sigma\in\prod_{v\in V}\Omega_v$ we define $\sigma|_{\boldsymbol{\pi}}\in\prod_{v\in V}Q_v$ by setting $\sigma|_{\boldsymbol{\pi}}(v)=\pi_v(\sigma(v))$ for all $v\in V$. For any constraint $C\in\mathcal{C}$ we define $C|_{\boldsymbol{\pi}}\colon\prod_{v\in\mathsf{vbl}(C)}Q_v\to\{\mathsf{True},\mathsf{False}\}$ by

$$C|_{\boldsymbol{\pi}}(\sigma)=\begin{cases}\mathsf{False} & \exists\sigma'\in\sigma^C_{\mathsf{False}}\text{ such that }\sigma'|_{\boldsymbol{\pi}}=\sigma,\\ \mathsf{True} & \text{otherwise.}\end{cases}\tag{1}$$

[7]Here Δ is one plus the maximum degree of the dependency graph of Φ since $C\in\{C'\in\mathcal{C}\mid\mathsf{vbl}(C)\cap\mathsf{vbl}(C')\neq\emptyset\}$.

Then we define $\Phi|_{\boldsymbol{\pi}} = \left(V, (Q_v)_{v\in V}, \mathcal{C}|_{\boldsymbol{\pi}}\right)$ where $\mathcal{C}|_{\boldsymbol{\pi}} = \{C|_{\boldsymbol{\pi}} \mid C \in \mathcal{C}\}$ and note the following fact.

Fact 2.2. *If Φ is atomic, then $\Phi|_{\boldsymbol{\pi}}$ is also atomic. Moreover, $\sigma_{\mathsf{False}}^{C|_{\boldsymbol{\pi}}} = \sigma_{\mathsf{False}}^{C}|_{\boldsymbol{\pi}}$ for all $C \in \mathcal{C}$, and $\sigma_{\mathsf{False}}^{\mathcal{C}|_{\boldsymbol{\pi}}} = \left\{\sigma|_{\boldsymbol{\pi}} \,\middle|\, \sigma \in \sigma_{\mathsf{False}}^{\mathcal{C}}\right\}$.*

We will reserve $\star$ as a special symbol. Our algorithms will use $\star$ to represent all the possibilities in some projected domain. In the rest of the paper, for any projection $\pi\colon \Omega \to Q$ we assume $\star \notin Q$. In addition, we abuse notation to define $\pi^{-1}(\star) = \Omega$ and extend (1) to include $\star$ as follows: $C|_{\boldsymbol{\pi}}\colon \prod_{v\in\mathsf{vbl}(C)} (\{\star\} \cup Q_v) \to \{\mathsf{True}, \mathsf{False}\}$ and

$$C|_{\boldsymbol{\pi}}(\sigma) = \begin{cases} \mathsf{False} & \exists \sigma' \in \sigma_{\mathsf{False}}^{C} \text{ such that } \pi_v(\sigma'(v)) = \sigma(v) \text{ holds for all } v \in \mathsf{vbl}(C), \sigma(v) \neq \star, \\ \mathsf{True} & \text{otherwise.} \end{cases} \tag{2}$$

Lovász Local Lemma. The Lovász local lemma (LLL) provides sufficient conditions to guarantee the existence of a satisfying assignment of CSPs.

Theorem 2.3 ([EL75])**.** *Let $\Phi = \left(V, (\Omega_v)_{v\in V}, \mathcal{C}\right)$ be a CSP. If $e \cdot p \cdot \Delta \leq 1$, then*

$$\frac{\left|\sigma_{\mathsf{True}}^{\mathcal{C}}\right|}{\prod_{v\in V} |\Omega_v|} \geq (1 - e \cdot p)^{|\mathcal{C}|} > 0.$$

Here we note the following two more general version: Theorem 2.4 studies the uniform distribution over $\sigma_{\mathsf{True}}^{\mathcal{C}}$; and Theorem 2.5 supplements the algorithmic aspect of Theorem 2.3.

Theorem 2.4 ([HSS11, Theorem 2.1])**.** *Let $\Phi = \left(V, (\Omega_v)_{v\in V}, \mathcal{C}\right)$ be a CSP. If $e \cdot p \cdot \Delta \leq 1$, then $\sigma_{\mathsf{True}}^{\mathcal{C}} \neq \emptyset$ and for any constraint B (not necessarily from $\mathcal{C}$) we have*

$$\Pr_{\sigma\sim\sigma_{\mathsf{True}}^{\mathcal{C}}}\left[B(\sigma) = \mathsf{True}\right] \leq (1 - e \cdot p)^{-|\Gamma(B)|} \Pr_{\sigma\sim\prod_{v\in V}\Omega_v}\left[B(\sigma) = \mathsf{True}\right],$$

where $\Gamma(B) = \{C \in \mathcal{C} : \mathsf{vbl}(C) \cap \mathsf{vbl}(B) \neq \emptyset\}$.

Theorem 2.5 ([MT10])**.** *Let $\Phi = \left(V, (\Omega_v)_{v\in V}, \mathcal{C}\right)$ be a CSP. If $e \cdot p \cdot \Delta \leq 1$, then $\sigma_{\mathsf{True}}^{\mathcal{C}} \neq \emptyset$ and there exists a randomized algorithm which outputs some $\sigma \in \sigma_{\mathsf{True}}^{\mathcal{C}}$ in time $O(k\Delta|V|)$ with probability at least 0.99.*

Rejection Sampling. The following simple perfect sampler, which is based on the standard rejection sampling technique, will be a building block of our main algorithm.

Algorithm 1: The $\mathsf{RejectionSampling}$ algorithm

Input: a CSP $\Phi = \left(V, (\Omega_v)_{v\in V}, \mathcal{C}\right)$ and a randomness tape r
/* Assume $V = \bigcup_{C\in\mathcal{C}} \mathsf{vbl}(C)$ when $\mathcal{C} \neq \emptyset$; and $|V| = 1$ when $\mathcal{C} = \emptyset$ */
. **Output:** an assignment $\sigma \in \sigma_{\mathsf{True}}^{\mathcal{C}}$
while True **do**
  Sample $\sigma \sim \prod_{v\in V} \Omega_v$ with fresh randomness from r
  if $C(\sigma) = \mathsf{True}$ *for all* $C \in \mathcal{C}$ **then return** σ
end

We have the following result on Algorithm 1 by basic facts of geometric distributions.

Fact 2.6. *The following holds for* `RejectionSampling`(Φ, r) *over random r.*

- *It halts almost surely, and outputs $\sigma \sim \sigma^{\mathcal{C}}_{\mathsf{True}}$ when it halts.*
- *Let T be the number of* while *iterations it takes before halting. Then*

$$\mathbb{E}[T] = \frac{\prod_{v\in V}|\Omega_v|}{\left|\sigma^{\mathcal{C}}_{\mathsf{True}}\right|} \quad \text{and} \quad \mathbb{E}\left[T^2\right] = 2\left(\frac{\prod_{v\in V}|\Omega_v|}{\left|\sigma^{\mathcal{C}}_{\mathsf{True}}\right|}\right)^2 - \frac{\prod_{v\in V}|\Omega_v|}{\left|\sigma^{\mathcal{C}}_{\mathsf{True}}\right|}.$$

- *Let X be its total running time.*[8] *Then*

$$\mathbb{E}[X] = O\left(\frac{\prod_{v\in V}|\Omega_v|}{\left|\sigma^{\mathcal{C}}_{\mathsf{True}}\right|}\cdot(|\mathcal{C}|+|V|)\right) \quad \text{and} \quad \mathbb{E}\left[X^2\right] = O\left(\left(\frac{\prod_{v\in V}|\Omega_v|}{\left|\sigma^{\mathcal{C}}_{\mathsf{True}}\right|}\cdot(|\mathcal{C}|+|V|)\right)^2\right).$$

Hypergraphs. Here we give definitions related with hypergraphs. All the definitions directly translate to graphs if we restrict the every edge in the hypergraph contains 2 vertices.

Let H be a hypergraph with finite *vertex set* $V(H)$ and finite *edge set* $E(H)$. Each edge $e \in E(H)$ is a non-empty subset of $V(H)$. We *allow* multiple occurrence of a same edge. For any CSP $\Phi = \left(V, (\Omega_v)_{v\in V}, \mathcal{C}\right)$ we naturally view it as a hypergraph $H(\Phi)$ where $V(H(\Phi)) = V$ and $E(H(\Phi)) = \{\mathsf{vbl}(C)\}_{C\in\mathcal{C}}$. Similar as the measures of CSPs, we define the following measures of a hypergraph H:

- the *width* is $k = k(H) = \max_{e\in E(H)}|e|$;
- the *vertex degree* is $d = d(H) = \max_{v\in V(H)}|\{e \in E(H) \mid v \in e\}|$;
- the *edge degree* is $\Delta = \Delta(H) = \max_{e\in E(H)}|\{e' \in E(H) \mid e\cap e' \neq \emptyset\}|$.

For any two vertices $u, v \in V(H)$, we say they are *adjacent* if there exists some $e \in E(H)$ such that $u \in e$ and $v \in e$; we say they are *connected* if there exists a vertex sequence $w_1, w_2, \ldots, w_d \in V(H)$ such that $w_1 = u, w_d = v$ and each w_i, w_{i+1} are adjacent. Then hypergraph H is *connected* if any two vertices $u, v \in V(H)$ are connected. Furthermore, we have the following basic fact regarding connected hypergraphs.

Fact 2.7. *Assume H is a connected hypergraph. Then for any $e, e' \in E(H)$, there exists a sequence of hyperedges $e_1, e_2, \ldots, e_\ell$ such that the following holds.*

- *$e_1 = e$, $e_\ell = e'$, and $e_i \cap e_{i+1} \neq \emptyset$ for all $i \in [\ell-1]$.*
- *$e_i \cap e_j = \emptyset$ for all $i, j \in [\ell]$ with $|i-j| > 1$.*

A hypergraph H' is a *sub-hypergraph* of H if $V(H') \subseteq V(H)$ and $E(H') \subseteq E(H)$. If in addition $e \cap V(H') = \emptyset$ holds for all $e \in E(H) \setminus E(H')$, we say H' is an *induced sub-hypergraph* of H.

3 The AtomicCSPSampling Algorithm

We first formally describe our main algorithm AtomicCSPSampling in Algorithm 2. The missing subroutines will be provided as we prove the correctness and efficiency of Algorithm 2.

Intuitively, the $\sigma_{\mathsf{Partial}}$ after the while iterations will be a random partial assignment in the projected domain with certain distribution; and the final output σ will be a uniform satisfying

[8] Each while iteration can be performed in time $O(|\mathcal{C}|+|V|)$ where $O(|V|)$ is for Line 2 and $O(|\mathcal{C}|)$ is for Line 3 assuming checking $C(\sigma) \stackrel{?}{=} \mathsf{True}$ takes unit time.

assignment of the original CSP conditioning on its projection being $\sigma_{\mathsf{Partial}}$. Putting them together, we will prove σ is a uniform random satisfying assignment of the original CSP.

Algorithm 2: The $\mathsf{AtomicCSPSampling}$ algorithm

Input: an atomic CSP $\Phi = \left(V, (\Omega_v)_{v\in V}, \mathcal{C}\right)$ and projections $\boldsymbol{\pi} = (\pi_v)_{v\in V}$
Output: an assignment $\sigma \in \sigma_{\mathsf{True}}^{\mathcal{C}}$

Assign infinitely long randomness r_i independently for each $i \in \mathbb{Z}$
Initialize $T \leftarrow 1$
while True **do**
  $\sigma_{\mathsf{Partial}} \leftarrow \texttt{BoundingChain}(\Phi, \boldsymbol{\pi}, -T, r_{-T}, \ldots, r_{-1})$
  if $\sigma_{\mathsf{Partial}}(v) \neq \star$ *for all* $v \in V$ **then** break
  else Update $T \leftarrow 2 \cdot T$
end
$\sigma \leftarrow \texttt{FinalSampling}(\Phi, \boldsymbol{\pi}, \sigma_{\mathsf{Partial}})$
return σ

The following additional notations are needed to state our main result. The idea behind each formula will be clear as we proceed.

Notation 3.1. Let $\Phi = \left(V, (\Omega_v)_{v\in V}, \mathcal{C}\right)$ be an atomic CSP and $\boldsymbol{\pi} = (\pi_v)_{v\in V}$ be projections where $\pi_v \colon \Omega_v \to Q_v$. We define the following notations, which can be easily computed given Φ and $\boldsymbol{\pi}$.

- The *maximal conditional falsifying probability* of $\Phi|_{\boldsymbol{\pi}}$ is
$$\alpha = \alpha(\Phi|_{\boldsymbol{\pi}}) = \max_{C\in\mathcal{C}} \frac{1}{\prod_{v\in\mathsf{vbl}(C)} \left|\pi_v^{-1}\left(\sigma_{\mathsf{False}}^{C|_\pi}(v)\right)\right|}.$$
- When $e \cdot \alpha < 1$, define
  - the *multiplicative bias* of $\Phi|_{\boldsymbol{\pi}}$ as $\beta = \beta(\Phi|_{\boldsymbol{\pi}}) = (1 - e\cdot\alpha)^{-d}$,
  - the *maximal multiplicative-biased falsifying probability* of $\Phi|_{\boldsymbol{\pi}}$ as
$$\rho = \rho(\Phi|_{\boldsymbol{\pi}}) = \max_{C\in\mathcal{C}} \prod_{v\in\mathsf{vbl}(C)} \min\left\{1, \beta\cdot\frac{\left|\pi_v^{-1}\left(\sigma_{\mathsf{False}}^{C|_{\boldsymbol{\pi}}}(v)\right)\right|}{|\Omega_v|}\right\},$$
  - the *uniformity parameter* of $\Phi|_{\boldsymbol{\pi}}$ on $v \in V$ as
$$\gamma_v = \gamma_v(\Phi|_{\boldsymbol{\pi}}) = (\beta - 1)\cdot\max_{q\in Q_v}\left(1 - \frac{\left|\pi_v^{-1}(q)\right|}{|\Omega_v|}\right),$$
  - the *total uniformity parameter* of $\Phi|_{\boldsymbol{\pi}}$ as
$$\lambda = \lambda(\Phi|_{\boldsymbol{\pi}}) = \max_{C\in\mathcal{C}} \prod_{v\in\mathsf{vbl}(C)} \left(\frac{\left|\pi_v^{-1}(\sigma_{\mathsf{False}}^{C|_{\boldsymbol{\pi}}}(v))\right|}{|\Omega_v|} + (|Q_v| - 1)\cdot\gamma_v\right).$$

Likewise, when context is clear, we will simply write $\alpha, \beta, \rho, \gamma_v, \lambda$.

Now we present our main theorem, the proof of which is the focus of the rest of the section.

Theorem 3.2. *Let $\Phi = \left(V, (\Omega_v)_{v\in V}, \mathcal{C}\right)$ be an atomic CSP. Let $\boldsymbol{\pi} = (\pi_v)_{v\in V}$ be projections. If $e\cdot\alpha\cdot\Delta \le 1$, $d^2k^4\lambda \le 1/2$, and $e\cdot\Delta^2\rho \le 1/32$, then the following holds for* $\texttt{AtomicCSPSampling}(\Phi, \boldsymbol{\pi})$.

- *Correctness. It halts almost surely and outputs $\sigma \sim \sigma_{\mathsf{True}}^{\mathcal{C}}$ when it halts.*
- *Efficiency. Its expected total running time is $O\left(kd^2\Delta^2|V|\log(|V|)\right)$.*

We remark that we ignore the dependency of the running time on $|\Omega_v|, v \in V$ in current version for simplicity.

3.1 A **Component** Subroutine and **RejectionSampling** under Projections

Before we proceed to the analysis of Theorem 3.2, we set up the following `Component`$(\Phi, \boldsymbol{\pi}, \sigma, v^*)$ subroutine, which uses current partial assignment σ to decompose CSP Φ into two disjoint parts: one containing v^* and one isolated from v^*.

Algorithm 3: The Component subroutine

Input: an atomic CSP $\Phi = \left(V, (\Omega_v)_{v\in V}, \mathcal{C}\right)$, projections $\boldsymbol{\pi} = (\pi_v)_{v\in V}, \pi_v \colon \Omega_v \to Q_v$, a partial assignment $\sigma \in \prod_{v\in V}(\{\star\} \cup Q_v)$, and $v^* \in V$

Output: (Φ^*, Token) where $\mathsf{Token} \in \{\mathsf{True}, \mathsf{False}\}$ and $\Phi^* = \left(V^*, \left(\pi_v^{-1}(\sigma(v))\right)_{v\in V^*}, \mathcal{C}^*\right)$ is a CSP

1 Initialize $V^* \leftarrow \{v^*\}$ and $\mathcal{C}^* \leftarrow \emptyset$
2 **while** $\exists C \in \mathcal{C} \setminus \mathcal{C}^*$ *with* $\mathsf{vbl}(C) \cap V^* \neq \emptyset$ *and* $C|_{\boldsymbol{\pi}}(\sigma) = \mathsf{False}$ **do**
3   **if** $\exists v \in \mathsf{vbl}(C) \setminus \{v^*\}$ *with* $\sigma(v) = \star$ **then return** (Φ^*, False)
4   **else** Update $V^* \leftarrow V^* \cup \mathsf{vbl}(C)$ and $\mathcal{C}^* \leftarrow \mathcal{C}^* \cup \{C\}$
5 **end**
6 **return** (Φ^*, True)

Here we note the following observation regarding Algorithm 3.

Lemma 3.3. *The following holds for* `Component`$(\Phi, \boldsymbol{\pi}, \sigma, v^*)$*.*

(1) It runs in time $O\left(k|\mathcal{C}^*| + d|V^*|\right) = O\left(dk|\mathcal{C}^*| + d\right)$.

(2) $v^* \in V^*$ and $\sigma(v) = \sigma_{\mathsf{False}}^{C|_{\boldsymbol{\pi}}}(v)$ for all $C \in \mathcal{C}^*$ and $v \in \mathsf{vbl}(C) \setminus \{v^*\}$.

(3) Φ^* is an atomic CSP and hypergraph $H(\Phi^*)$ is connected.[9]

(4) If $\mathsf{Token} = \mathsf{True}$, let $\Phi' = \left(V', \left(\pi_v^{-1}(\sigma(v))\right)_{v\in V'}, \mathcal{C}'\right)$ where $V' = V \setminus V^*$ and

$$\mathcal{C}' = \left\{C \in \mathcal{C} \setminus \mathcal{C}^* \mid C|_{\boldsymbol{\pi}}(\sigma) = \mathsf{False}\right\}.$$

Then Φ' is an atomic CSP and $\sigma_{\mathsf{True}}^{\mathcal{C}} \cap \left(\prod_{v\in V} \pi_v^{-1}(\sigma(v))\right) = \sigma_{\mathsf{True}}^{\mathcal{C}^*} \times \sigma_{\mathsf{True}}^{\mathcal{C}'}$.

Proof. Item (2) is evident from Line 1 and Line 3 of Algorithm 3.

For Item (3), note that each time we add a constraint C into $\mathcal{C}^*$, we add $\mathsf{vbl}(C)$ into V^*. Meanwhile Φ is atomic. Thus by Fact 2.2 Φ^* is also an atomic CSP. In addition, we only consider C with $\mathsf{vbl}(C) \cap V^* \neq \emptyset$ in each while iteration, hence $H(\Phi^*)$ is connected.

For Item (1), firstly the algorithm can be performed in time $O\left(k|\mathcal{C}^*| + d|V^*|\right)$ by iteratively adding variables inside constraints and checking constraints connected with the newly added variables. On the other hand $|V^*| \leq k|\mathcal{C}^*| + 1$ since $H(\Phi^*)$ is connected by Item (3).[10] Therefore $O\left(k|\mathcal{C}^*| + d|V^*|\right) = O\left(dk|\mathcal{C}^*| + d\right)$.

---

[9] This means each Φ_i satisfies the assumption in Algorithm 1.

[10] The additional $+1$ is for the case $\mathcal{C}^* = \emptyset$ and $|V^*| = 1$.

Now we focus on Item (4) when $\mathsf{Token} = \mathsf{True}$. The condition in the while iteration implies for any $C' \in \mathcal{C}'$, $\mathsf{vbl}(C') \cap V^* = \emptyset$ and thus $\mathsf{vbl}(C') \subseteq V'$. Therefore Φ' is a CSP, and since Φ is atomic, Φ' is also atomic. Fix some $\sigma \in \prod_{v \in V} \pi_v^{-1}(\sigma(v))$. Since

$$\sigma_{\mathsf{True}}^{\mathcal{C}^*} \times \sigma_{\mathsf{True}}^{\mathcal{C}'} \subseteq \left(\prod_{v \in V^*} \pi_v^{-1}(\sigma(v))\right) \times \left(\prod_{v \in V'} \pi_v^{-1}(\sigma(v))\right) = \prod_{v \in V} \pi_v^{-1}(\sigma(v)),$$

it suffices to show $\sigma \in \sigma_{\mathsf{True}}^{\mathcal{C}}$ if and only if $\sigma \in \sigma_{\mathsf{True}}^{\mathcal{C}^*} \times \sigma_{\mathsf{True}}^{\mathcal{C}'}$ as follows

$$\begin{aligned}
\sigma \in \sigma_{\mathsf{True}}^{\mathcal{C}} &\iff C(\sigma) = \mathsf{True}, \forall C \in \mathcal{C} \\
&\iff C(\sigma) = \mathsf{True}, \forall C \in \{C \in \mathcal{C} \mid C|_{\boldsymbol{\pi}}(\sigma) = \mathsf{False}\} && \text{(by (2))} \\
&\iff C(\sigma) = \mathsf{True}, \forall C \in \mathcal{C}^* \cup \mathcal{C}' \\
&\iff \left(\sigma(V^*) \in \sigma_{\mathsf{True}}^{\mathcal{C}^*}\right) \wedge \left(\sigma(V') \in \sigma_{\mathsf{True}}^{\mathcal{C}'}\right) \\
&\iff \sigma \in \sigma_{\mathsf{True}}^{\mathcal{C}^*} \times \sigma_{\mathsf{True}}^{\mathcal{C}'}. && \text{(since } V^* \cap V' = \emptyset)
\end{aligned}$$

□

The following lemma is useful when $\mathsf{Token} = \mathsf{True}$ and we perform rejection sampling on Φ^*.

Lemma 3.4. *Let $\sigma \in \prod_{v \in V} (\{\star\} \cup Q_v)$ be arbitrary. Let $\Phi^* = \left(V^*, \left(\pi_v^{-1}(\sigma(v))\right)_{v \in V^*}, \mathcal{C}^*\right)$ with $V^* \subseteq V, \mathcal{C}^* \subseteq \mathcal{C}$ be an arbitrary CSP that $H(\Phi^*)$ is connected. Let X be the running time of* `RejectionSampling`(Φ^*, r) *over random r.*

Let $k = k(\Phi)$, $\Delta = \Delta(\Phi)$, and $\alpha = \alpha(\Phi|_{\boldsymbol{\pi}})$. If $e \cdot \alpha \cdot \Delta \leq 1$, then $X < +\infty$ almost surely and

$$\mathbb{E}[X] = O\left(\frac{k|\mathcal{C}^*| + 1}{(1 - e \cdot \alpha)^{|\mathcal{C}^*|}}\right) \quad \textit{and} \quad \mathbb{E}\left[X^2\right] = O\left(\frac{(k|\mathcal{C}^*| + 1)^2}{(1 - e \cdot \alpha)^{2 \cdot |\mathcal{C}^*|}}\right).$$

Proof. Firstly $\Delta(\Phi^*) \leq \Delta$. Then for any $C \in \mathcal{C}^*$ with $\sigma_{\mathsf{False}}^{C} \neq \emptyset$, we have

$$\begin{aligned}
\frac{\left|\sigma_{\mathsf{False}}^{C}\right|}{\prod_{v \in \mathsf{vbl}(C)} \left|\pi_v^{-1}(\sigma(v))\right|} &= \frac{1}{\prod_{v \in \mathsf{vbl}(C)} \left|\pi_v^{-1}(\sigma(v))\right|} && \text{(since } \Phi \text{ is atomic and by Fact 2.2)} \\
&\leq \frac{1}{\prod_{v \in \mathsf{vbl}(C)} \left|\pi_v^{-1}\left(\sigma_{\mathsf{False}}^{C|_{\boldsymbol{\pi}}}(v)\right)\right|} \leq \alpha. && \text{(since } \pi_v^{-1}(\star) = \Omega_v)
\end{aligned}$$

Thus $p(\Phi^*) \leq \alpha$ and we apply Theorem 2.3 to obtain

$$\frac{\left|\sigma_{\mathsf{True}}^{\mathcal{C}^*}\right|}{\prod_{v \in V^*} \left|\pi_v^{-1}(\sigma(v))\right|} \geq (1 - e \cdot \alpha)^{|\mathcal{C}^*|} > 0.$$

Since $H(\Phi^*)$ is connected and thus $|V^*| \leq k|\mathcal{C}^*| + 1$, the bound follows naturally from Fact 2.6. □

3.2 The BoundingChain Subroutine

Now we present and analyze the missing $\texttt{BoundingChain}(\Phi, \boldsymbol{\pi}, -T, r_{-T}, \ldots, r_{-1})$ subroutine.

Algorithm 4: The BoundingChain subroutine

Input: an atomic CSP $\Phi = \left(V, (\Omega_v)_{v \in V}, \mathcal{C}\right)$, projections $\boldsymbol{\pi} = (\pi_v)_{v \in V}, \pi_v \colon \Omega_v \to Q_v$, starting time $-T$, and randomness tapes $r_{-T}, \ldots, r_{-1}$

Output: a partial assignment $\sigma \in \prod_{v \in V} (\{\star\} \cup Q_v)$

Initialize $\sigma(x_i) \leftarrow \star$ for all $i = 0, 1, \ldots, n-1$ /* Assume $V = \{x_0, \ldots, x_{n-1}\}$. */

for $t = -T$ **to** -1 **do**

  $i_t \leftarrow t \mod n$, and $\sigma(x_{i_t}) \leftarrow \star$ /* Update $\sigma(x_{i_t})$ in this round. */

  $(\Phi_t, \mathsf{Token}_t) \leftarrow \texttt{Component}(\Phi, \boldsymbol{\pi}, \sigma, x_{i_t})$ where $\Phi_t = \left(V_t, \left(\pi_v^{-1}(\sigma(v))\right)_{v \in V_t}, \mathcal{C}_t\right)$

  if $\mathsf{Token}_t = \mathsf{True}$ **then**

    $\sigma' \leftarrow \texttt{RejectionSampling}(\Phi_t, r_t)$

    Update $\sigma(x_{i_t}) \leftarrow \pi_{x_{i_t}}(\sigma'(x_{i_t}))$

  else

    $b \leftarrow \texttt{SafeSampling}(\Omega_{x_{i_t}}, \pi_{x_{i_t}}, \gamma_{x_{i_t}}(\Phi|_{\boldsymbol{\pi}}), r_t)$ /* $\gamma_v(\Phi|_{\boldsymbol{\pi}})$ is in Notation 3.1 */

    Update $\sigma(x_{i_t}) \leftarrow b$

  end

end

return σ

Procedure $\texttt{SafeSampling}(\Omega, \pi, \gamma, r)$**:**

  Sample c from $\mathcal{D}$ using r where $\mathcal{D}$ is a distribution over $Q \cup \{\star\}$ by

$$\mathcal{D}(q) = \begin{cases} \max\left\{0, \frac{|\pi^{-1}(q)|}{|\Omega|} - \gamma\right\} & q \in Q, \\ 1 - \sum_{q \in Q} \max\left\{0, \frac{|\pi^{-1}(q)|}{|\Omega|} - \gamma\right\} & q = \star. \end{cases}$$

  return c

end

We will show the following result for *one single call* of Algorithm 4.

Proposition 3.5. *If $e \cdot \alpha \cdot \Delta \le 1$ and $e \cdot \Delta^2 \rho \le 1/32$, then the following holds over random $r_{-T}, \ldots, r_{-1}$ for* $\texttt{BoundingChain}(\Phi, \boldsymbol{\pi}, -T, r_{-T}, \ldots, r_{-1})$.

- *Efficiency. Let X_t be the running time of the t-th* for *iteration. Then $X_t < +\infty$ almost surely and $\mathbb{E}\left[X_t^2\right] = O\left(k^2 d^3 \Delta^3\right)$.*
- *Coalescence. Let $\mathcal{E}$ be the event "in the returned assignment σ, there exists some $v \in V$ with $\sigma(v) = \star$". If $T \ge 2|V| - 1$, we have*

$$\mathbf{Pr}\left[\mathcal{E}\right] \le dk|V| \cdot \left(d^2 k^4 \lambda\right)^{\frac{T}{2(|V|-1)} - 1}.$$

3.2.1 Moment Bounds on the Running Time

To establish the efficiency part of Proposition 3.5, we first control the size of Φ_t. This needs some additional definitions.

Definition 3.6 (2-tree)**.** Let $G = (V, E)$ be an undirected graph. A set of vertices $S \subseteq V$ is a 2-*tree* if the following holds.

- $\mathsf{dist}_G(u, v) \ge 2$ holds for any distinct $u, v \in S$ where $\mathsf{dist}_G(u, v)$ is the length of the shortest path in G from u to v.[11]
- If we add an edge between every $u, v \in S$ with $\mathsf{dist}_G(u, v) = 2$, then S is connected.

Intuitively a 2-tree is an independent set that is not very spread out. The following lemmas bounds the number of 2-trees and show how to extract a large 2-tree from any connected subgraph.

Lemma 3.7 ([FGYZ20, Corollary 5.7])**.** *Let $G = (V, E)$ be a graph with maximum degree d. Then for any $v \in V$ and integer $\ell \ge 1$, the number of 2-trees in G of size ℓ containing v is at most $\left(e \cdot d^2\right)^{\ell-1}/2$.*

Lemma 3.8 ([JPV21, Lemma 4.5])**.** *Let $G = (V, E)$ be a graph with maximum degree d. Let G' be a connected subgraph of G. Then for any $v \in V(H)$, there exists a 2-tree $S \subseteq V(G')$ with $v \in S$ and size $|S| \ge |V(H)|/(d + 1)$.*

Lemma 3.9 ([FGYZ20, Observation 5.5])**.** *If a graph $G = (V, E)$ has a 2-tree of size $\ell > 1$ containing $v \in V$, then G also has a 2-tree of size $\ell - 1$ containing v.*

The following result is a immediate corollary of Lemma 3.8 and Lemma 3.9.

Corollary 3.10. *Let $G = (V, E)$ be a graph with maximum degree d. Let G' be a connected subgraph of G. Then for any $v \in V(H)$ and any integer $\ell \le \lceil |V(H)|/(d+1) \rceil$, there exists a 2-tree $S \subseteq V(G')$ with $v \in S$ and size $|S| = \ell$.*

Finally we need the following comparison result which, under some mild assumption, shows the marginal probability does *not* change much after conditioning.

Lemma 3.11. *Let $\sigma \in \prod_{v \in V} (\{\star\} \cup Q_v)$ be arbitrary. Let $\Phi^* = \left(V^*, \left(\pi_v^{-1}(\sigma(v))\right)_{v \in V^*}, \mathcal{C}^*\right)$ with $V^* \subseteq V, \mathcal{C}^* \subseteq \mathcal{C}$ be an arbitrary CSP.*

Let $\Delta = \Delta(\Phi)$, $\alpha = \alpha(\Phi|_{\boldsymbol{\pi}})$, and $\beta = \beta(\Phi|_{\boldsymbol{\pi}})$. If $e \cdot \alpha \cdot \Delta \le 1$, then for any $v \in V$ with $\sigma(v) = \star$ and any $q \in Q_v$ we have

$$\Pr_{\sigma' \sim \sigma_{\mathsf{True}}^{\mathcal{C}^*}} \left[\pi_v(\sigma'(v)) = q\right] \le \beta \cdot \frac{\left|\pi_v^{-1}(q)\right|}{|\Omega_v|}.$$

Proof. Similar as the proof of Lemma 3.4, $p(\Phi^*) \le \alpha$. Let $B(\sigma')$ be the event (i.e., constraint) "$\pi_v(\sigma'(v)) = q$". Then $\mathsf{vbl}(B) = \{v\}$ and

$$\Pr_{\sigma' \sim \sigma_{\mathsf{True}}^{\mathcal{C}^*}} \left[B(\sigma')\right] = \Pr_{\sigma \sim \sigma_{\mathsf{True}}^{\mathcal{C}^*}} \left[\pi_v(\sigma'(v)) = q\right] \quad \text{and} \quad \Pr_{\sigma' \sim \prod_{v \in V^*} \pi_v^{-1}(\sigma(v))} \left[B(\sigma')\right] = \frac{\left|\pi_v^{-1}(q)\right|}{|\Omega_v|}.$$

Thus the bound follows immediately from Theorem 2.4 by the definition of β and noticing

$$|\{C \in \mathcal{C}^* \mid \mathsf{vbl}(B) \cap \mathsf{vbl}(C) \ne \emptyset\}| = |\{C \in \mathcal{C}^* \mid v \in \mathsf{vbl}(C)\}| \le d(\Phi). \qquad \square$$

Now we show the size of Φ_t is concentrated.

[11] For example $\mathsf{dist}_G(u, u) \equiv 0$ for all $u \in V$, and $\mathsf{dist}_G(u, v) = 1$ iff (u, v) is an edge in E.

Lemma 3.12. *Let $d = d(\Phi)$, $\Delta = \Delta(\Phi)$, $\alpha = \alpha(\Phi|_{\boldsymbol{\pi}})$, $\beta = \beta(\Phi|_{\boldsymbol{\pi}})$, and $\rho = \rho(\Phi|_{\boldsymbol{\pi}})$.*

For any $t \in \{-T, \dots, -1\}$, if $e \cdot \alpha \cdot \Delta \le 1$ then recall $\mathcal{C}_t$ defined in Algorithm 4 and we have

$$\mathbf{Pr}\left[|\mathcal{C}_t| \ge \ell \cdot \Delta\right] \le \frac{d}{2} \cdot \left(e \cdot \Delta^2 \rho\right)^{\ell - 1} \quad \text{for any integer } \ell \ge 1.$$

Proof. Construct the *line graph* $\mathsf{Lin}(\Phi) = \left(V^\Phi, E^\Phi\right)$ of $\Phi = \left(V, (\Omega_v)_{v \in V}, \mathcal{C}\right)$ where

$$V^\Phi = \mathcal{C} \quad \text{and} \quad E^\Phi = \left\{\{e_1, e_2\} \in \mathcal{C} \times \mathcal{C} \mid \mathsf{vbl}(e_1) \cap \mathsf{vbl}(e_2) \ne \emptyset, e_1 \ne e_2\right\}.$$

Then $\mathsf{Lin}(\Phi)$ is an undirected graph with maximum degree $\Delta - 1$.

Let G' be the subgraph of $\mathsf{Lin}(\Phi)$ induced by vertex set $\mathcal{C}_t$. Then by Lemma 3.3, G' is a connected subgraph of G. For any $C \in \mathcal{C}_t$ with $x_{i_t} \in \mathsf{vbl}(C)$, by Corollary 3.10 there exists a 2-tree $S^* \subseteq \mathcal{C}_t$ with $C \in S^*$ and size $|S^*| = \ell$ provided $\ell \le \lceil |\mathcal{C}_t| / \Delta \rceil$. Define

$$\mathcal{S} = \left\{\text{2-tree } S \subseteq V^\Phi \,\middle|\, (|S| = \ell) \wedge (\exists C \in S, x_{i_t} \in \mathsf{vbl}(C))\right\}.$$

Then by Lemma 3.7 and noticing there are at most d many choices of C, we have

$$|\mathcal{S}| \le \frac{d \cdot \left(e \cdot (\Delta - 1)^2\right)^{\ell - 1}}{2} \le \frac{d \cdot \left(e \cdot \Delta^2\right)^{\ell - 1}}{2}.$$

By Item (2) of Lemma 3.3, $\sigma(v) = \sigma_{\mathsf{False}}^{C|_{\boldsymbol{\pi}}}(v)$ for any $C \in \mathcal{C}_t$ and $v \in \mathsf{vbl}(C) \setminus \{x_{i_t}\}$. Note that $\sigma(v)$ is initialized as $\star \notin Q_v$. Thus $\sigma(v)$ must be updated before the t-th **for** iteration. We analyze the two possible ways of the last update of $\sigma(v)$ separately.

- $\mathsf{SafeSampling}$. Then it is set to $\sigma_{\mathsf{False}}^{C|_{\boldsymbol{\pi}}}(v)$ with probability at most $\frac{\left|\pi_v^{-1}(\sigma_{\mathsf{False}}^{C|_{\boldsymbol{\pi}}}(v))\right|}{|\Omega_v|}$.
- $\mathsf{RejectionSampling}$. Then by Lemma 3.3, Fact 2.6, and Lemma 3.11, it is set to $\sigma_{\mathsf{False}}^{C|_{\boldsymbol{\pi}}}(v)$ with probability at most $\min\left\{1, \beta \cdot \frac{\left|\pi_v^{-1}(\sigma_{\mathsf{False}}^{C|_{\boldsymbol{\pi}}}(v))\right|}{|\Omega_v|}\right\}$.

Since $\beta \ge 1$, the bound from $\mathsf{RejectionSampling}$ dominates both. Therefore

$$\begin{aligned}
\mathbf{Pr}\left[|\mathcal{C}_t| \ge \ell \cdot \Delta\right] &\le \mathbf{Pr}\left[\sigma(v) = \sigma_{\mathsf{False}}^{C|_{\boldsymbol{\pi}}}(v), \forall C \in \mathcal{C}_t, v \in \mathsf{vbl}(C) \setminus \{x_{i_t}\}\right] \\
&\le \mathbf{Pr}\left[\sigma(v) = \sigma_{\mathsf{False}}^{C|_{\boldsymbol{\pi}}}(v), \forall C \in S^*, v \in \mathsf{vbl}(C) \setminus \{x_{i_t}\}\right] \\
&\le \sum_{S \in \mathcal{S}} \mathbf{Pr}\left[\sigma(v) = \sigma_{\mathsf{False}}^{C|_{\boldsymbol{\pi}}}(v), \forall C \in S, v \in \mathsf{vbl}(C) \setminus \{x_{i_t}\}\right] && \text{(by union bound)} \\
&\le \sum_{S \in \mathcal{S}} \prod_{C \in S} \prod_{v \in \mathsf{vbl}(C) \setminus \{x_{i_t}\}} \min\left\{1, \beta \cdot \frac{\left|\pi_v^{-1}(\sigma_{\mathsf{False}}^{C|_{\boldsymbol{\pi}}}(v))\right|}{|\Omega_v|}\right\} \\
& && \text{(since } \beta \ge 1 \text{ and } (\mathsf{vbl}(C))_{C \in S} \text{ are pairwise disjoint)} \\
&\le \sum_{S \in \mathcal{S}} \prod_{C \in S, x_{i_t} \notin \mathsf{vbl}(C)} \prod_{v \in \mathsf{vbl}(C)} \min\left\{1, \beta \cdot \frac{\left|\pi_v^{-1}(\sigma_{\mathsf{False}}^{C|_{\boldsymbol{\pi}}}(v))\right|}{|\Omega_v|}\right\} \\
&\le \sum_{S \in \mathcal{S}} \rho^{|S| - 1} = \frac{d}{2} \cdot \left(e \cdot \Delta^2 \rho\right)^{\ell - 1}.
\end{aligned}$$

$\square$

Now we obtain moment bounds for the running time of each `for` iteration in Algorithm 4.

Proof of the Efficiency Part of Proposition 3.5. Let Z_t be the running time of Line 4. Then by Lemma 3.3, $Z_t = O\left(dk\left|\mathcal{C}_t\right| + d\right)$. By Lemma 3.12, we also have

$$\mathbf{Pr}\left[\left|\mathcal{C}_t\right| \ge \ell \cdot \Delta\right] \le \frac{d}{2} \cdot \left(e \cdot \Delta^2 \rho\right)^{\ell-1} \le \frac{d}{2} \cdot \left(\frac{1}{32}\right)^{\ell-1} \quad \text{for any integer } \ell \ge 1.$$

Let Y_t be the running time of Line 5-11. By Lemma 3.4 and noticing SafeSampling takes unit time, we have $Y_t < +\infty$ almost surely and

$$\mathbb{E}\left[Y_t^2 \,\middle|\, \left|\mathcal{C}_t\right|\right] \le O\left(\frac{\left(k\left|\mathcal{C}_t\right| + 1\right)^2}{(1 - e \cdot \alpha)^{2 \cdot \left|\mathcal{C}_t\right|}}\right).$$

Since $X_t = Z_t + Y_t$, we have

$$\begin{aligned}
\mathbb{E}\left[X_t^2\right] &\le \mathbb{E}\left[2 \cdot \left(Y_t^2 + Z_t^2\right)\right] = 2\sum_{L=0}^{+\infty} \mathbf{Pr}\left[\left|\mathcal{C}_t\right| = L\right] \mathbb{E}\left[Z_t^2 + Y_t^2 \,\middle|\, \left|\mathcal{C}_t\right| = L\right] \\
&\le \sum_{\ell=0}^{+\infty} \mathbf{Pr}\left[\left|\mathcal{C}_t\right| \ge \ell \cdot \Delta\right] \cdot \Delta \cdot O\left(\left(dk\Delta(\ell+1)\right)^2 + \frac{\left(k\Delta(\ell+1)\right)^2}{(1 - e \cdot \alpha)^{2(\ell+1)\Delta}}\right) \\
&\le \sum_{\ell=0}^{+\infty} \mathbf{Pr}\left[\left|\mathcal{C}_t\right| \ge \ell \cdot \Delta\right] \cdot \Delta \cdot O\left(\left(dk\Delta(\ell+1)\right)^2 + \left(k\Delta(\ell+1)\right)^2 \cdot 16^{\ell+1}\right) \\
&\qquad\qquad \text{(since } e \cdot \alpha \ge 1/\Delta \text{ and } \Delta \ge 2\text{, we have } (1 - e \cdot \alpha)^{\Delta} \ge 1/4\text{)} \\
&\le O\left(d\Delta\right) \sum_{\ell=1}^{+\infty} \left(\frac{1}{32}\right)^{\ell-1} \left(\left(dk\Delta(\ell+1)\right)^2 + \left(k\Delta(\ell+1)\right)^2 \cdot 16^{\ell+1}\right) \\
&= O\left(k^2 d^3 \Delta^3\right).
\end{aligned}$$

□

3.2.2 Concentration Bounds for the Coalescence

Here we analyze the coalescence part of Proposition 3.5, which is the stopping condition for the `while` iterations in Algorithm 2. We use *information percolation* argument which needs additional setup.

We follow the notation convention in Algorithm 4: $V = \{x_0, \ldots, x_{n-1}\}$ and $i_t = t \mod n$ for $t \in \{-T, \ldots, -1\}$. For convenience, we will use $\sigma_t, t \in \{-T, \ldots, -1\}$ to denote the assignment σ at Line 4 of the t-th `for` iteration in Algorithm 4. In particular $\sigma_t(x_{i_t}) = \star$ since it is about to update $\sigma_t(x_{i_t})$. We also define σ_0 as the final returned assignment.

Define $\mathsf{UpdTime}(x_j, t)$ as the last update time of x_j *before* the t-th `for` iteration, i.e.,

$$\mathsf{UpdTime}(x_j, t) = \max\left\{-T-1\right\} \cup \left\{t' < t \,\middle|\, t' \equiv j \mod n\right\}. \tag{3}$$

The additional $-T-1$ is to set up the boundary condition which corresponds to the initialization step.

Definition 3.13 (Extended Constraints)**.** For any $C \in \mathcal{C}$ and any $e = \{t_1, \ldots, t_m\} \subseteq \{-T, \ldots, -1\}$, we say (e, C) is an *extended constraint* if the following holds.

(1) $m = \left|\mathsf{vbl}(C)\right|$ and $\mathsf{vbl}(C) = \left\{x_{i_{t_1}}, x_{i_{t_2}}, \ldots, x_{i_{t_m}}\right\}$.

(2) $e = \{\mathsf{UpdTime}(v, t_{\mathsf{max}} + 1) \mid v \in \mathsf{vbl}(C)\}$ where $t_{\mathsf{max}} = \max_{t' \in e} t'$.

Intuitively an extended constraint of C is a consecutive rounds of updates on $\mathsf{vbl}(C)$, and we have the following fact.

Fact 3.14. *The following holds for extended constraints.*

(1) If (e_1, C_1) and (e_2, C_2) are two extended constraints and $\mathsf{vbl}(C_1) \cap \mathsf{vbl}(C_2) = \emptyset$, then $e_1 \cap e_2 = \emptyset$.

(2) If (e, C) is an extended constraint, then $0 \le t_{\mathsf{max}}(e) - t_{\mathsf{min}}(e) < n$, where $t_{\mathsf{min}}(e) = \min_{t' \in e} t'$. Moreover, $e = \{\mathsf{UpdTime}(v, t') \mid v \in \mathsf{vbl}(C)\}$ for any $t_{\mathsf{max}} + 1 \le t' \le t_{\mathsf{min}} + n$.

(3) For any $t \in \{-T, \ldots, -1\}$ and $C \in \mathcal{C}$, we have

$$|\{e \mid (e, C) \text{ is an extended constraint with } t \in e\}| \le |\mathsf{vbl}(C)| \le k(\Phi).$$

Proof. Item (1) is evident from Item (1) of Definition 3.13.

For Item (2) and (3), assume $|\mathsf{vbl}(C)| = m$ and $\mathsf{vbl}(C) = \{x_{a_1}, \ldots, x_{a_m}\}$. Let

$$S = \{-T, \ldots, -1\} \cap \{a_i - j \cdot n \mid i \in [m], j \in \mathbb{Z}\} = \{b_1, b_2, \ldots, b_{T'}\}$$

where $-T \le b_1 < \cdots < b_{T'} \le -1$. Note that $b_i \equiv b_{i+m} \mod n$ for all i. If (e, C) is an extended constraint, then by Definition 3.13 e consists of a consecutive interval of S, i.e., $e = \{b_o, b_{o+1}, \ldots, b_{o+m-1}\}$ for some $o \in [T']$. Thus $t_{\mathsf{max}}(e) - t_{\mathsf{min}}(e) = b_{o+m-1} - b_o < n$. Hence $\mathsf{UpdTime}(x_{a_i}, b_{o+m-1} + 1) = \mathsf{UpdTime}(x_{a_i}, t')$ for all $i \in [m]$ and $b_{o+m-1} + 1 \le t' \le b_o + n$. Meanwhile, if we fix C and some $t \in e$, there are at most $m \le k(\Phi)$ choices of (e, C). $\square$

Definition 3.15 (Extended Hypergraph)**.** *Extended hypergraph* $H^{\mathsf{ext}} = (V^{\mathsf{ext}}, E^{\mathsf{ext}})$ has vertex set $V^{\mathsf{ext}} = \{-T, \ldots, -1\}$ and extended constraints as hyperedges:

$$E^{\mathsf{ext}} = \{e \subseteq \{-T, \ldots, -1\} \mid (e, C) \text{ is an extended constraint}\}.$$

Moreover, we label each hyperedge e with C if it is added into E^{ext} by extended constraint (e, C). We allow multiple occurrence of the same edge but the label should be different.

We note the following simple fact.

Fact 3.16. $\Delta(H^{\mathsf{ext}}) \le d(\Phi) k(\Phi)^2$.

Proof. Assume $e \in E^{\mathsf{ext}}$ achieves $\Delta(H^{\mathsf{ext}})$. Then by Item (3) of Fact 3.14 we have

$$\Delta(H^{\mathsf{ext}}) \le \sum_{t \in e} \sum_{C' \in \mathcal{C}, x_{i_t} \in \mathsf{vbl}(C')} \left|\left\{\text{extended constraint } (e', C') \,\middle|\, t \in e'\right\}\right| \le k(\Phi) \cdot d(\Phi) \cdot k(\Phi). \qquad \square$$

Now we present the following algorithm to sequentially find some constraints that are not satisfied during the $\mathsf{BoundingChain}$ process. We remark that this algorithm is only for our analysis, and we do *not* actually run it in $\mathsf{AtomicCSPSampling}$.

Algorithm 5: Finding failed constraints during the $\mathsf{BoundingChain}$ process

Input: partial assignments $(\sigma_t)_{t\in\{-T,\dots,0\}}$ from $\texttt{BoundingChain}(\Phi, \boldsymbol{\pi}, -T, r_{-T}, \dots, r_{-1})$ and some $v^* \in V$ with $\sigma_0(v^*) = \star$

Output: $H^* = (V^*, E^*)$ where $V^* \subseteq V^{\mathsf{ext}}$ and $E^* \subseteq E^{\mathsf{ext}}$

1 $t_0 \leftarrow \mathsf{UpdTime}(v^*, 0)$
2 Initialize $V^* \leftarrow \{t_0\}$ and $E^* \leftarrow \emptyset$
3 $\texttt{FailedConstraints}(t_0)$
4 **return** (V^*, E^*)
5 **Procedure** $\texttt{FailedConstraints}(t)$**:**
6   **if** $t < -T + n - 1$ **then return** /* $\mathsf{Token}_t = \mathsf{False}$ if $t \geq -T$. */
7   Initialize $V_t \leftarrow \{x_{i_t}\}$ and $\mathcal{C}_t \leftarrow \emptyset$
8   **while** $\exists C \in \mathcal{C} \setminus \mathcal{C}_t$ *with* $\mathsf{vbl}(C) \cap V_t \neq \emptyset$ *and* $C|_{\boldsymbol{\pi}}(\sigma_t) = \mathsf{False}$ **do**
9         $e \leftarrow \{\mathsf{UpdTime}(v, t+1) \mid v \in \mathsf{vbl}(C)\}$ /* (e, C) is an extended constraint */
10   Update $\mathcal{C}_t \leftarrow \mathcal{C}_t \cup \{C\}$ and $V_t \leftarrow V_t \cup \mathsf{vbl}(C)$
11   Update $E^* \leftarrow E^* \cup \{e\}$ and $V^* \leftarrow V^* \cup e$ /* e is labelled by C. */
12   **end**
13   **foreach** $v \in V_t \setminus \{x_{i_t}\}$ *with* $\sigma_t(v) = \star$ **do**
14         $\texttt{FailedConstraints}(\mathsf{UpdTime}(v, t))$ /* $\mathsf{UpdTime}(v, t) = \mathsf{UpdTime}(v, t+1)$ */
15   **end**
16 **end**

We have the following observation regarding Algorithm 5.

Lemma 3.17. *Algorithm 5 halts always. Furthermore, if $T \geq 2n - 1$ then*

(1) for each (e, C) from Line 8 *when we execute* $\texttt{FailedConstraints}(t)$*,*

(1a) it is an extended constraint,

(1b) for each $v \in \mathsf{vbl}(C)$, the assignment on v is updated to $\star$ or $\sigma^{C|_{\boldsymbol{\pi}}}_{\mathsf{False}}(v)$ in the $\mathsf{UpdTime}(v, t+1)$-th for *iteration in Algorithm 4;*

(2) each time when we call $\texttt{FailedConstraints}(t)$, t is already in V^*;

(3) H^* is a connected sub-hypergraph of H^{ext};

(4) there exists some $e_0, e_1 \in E^*$ such that $t_{\mathsf{max}}(e_0) \geq -n$ and $t_{\mathsf{min}}(e_1) < -T + n - 1$.

Proof. Since $\mathsf{UpdTime}(v, t) < t$ for all $v \in V$ and $t \in \{-T, \dots, 0\}$, Algorithm 5 always halts.

We prove Item (1) by induction on the calls of $\texttt{FailedConstraints}(t)$. The first call t_0 represents the final update of the assignment on v^*, which results in $\sigma_0(v^*) = \star$.

- $\boxed{\text{Item (1a) for } t_0.}$ Note that $t_0 \geq -T + n - 1$ since $T \geq 2n - 1$ and $t_0 \geq -n$. Then $\mathsf{UpdTime}(v, t_0 + 1) = t_0$ for $v = x_{i_{t_0}} = v^*$; and $\mathsf{UpdTime}(v, t_0 + 1) \geq -T$ for all $v \neq x_{i_{t_0}}$.
- $\boxed{\text{Item (1b) for } t_0.}$ By (2), $\sigma_t(v) = \star$ or $\sigma^{C|_{\boldsymbol{\pi}}}_{\mathsf{False}}(v)$ for all $v \in \mathsf{vbl}(C)$. This means, if $v \neq x_{i_{t_0}}$, the assignment on v is updated to such value in the $\mathsf{UpdTime}(v, t_0) = \mathsf{UpdTime}(v, t_0 + 1)$-th for iteration in Algorithm 4. On the other hand, since $\sigma_0(v^*) = \star$ and $t_0 = \mathsf{UpdTime}(v^*, 0)$, the assignment on $v^* = x_{i_{t_0}}$ is updated to $\star$ in the $t_0 = \mathsf{UpdTime}(v, t_0 + 1)$-th for iteration.

To complete the induction, we note that each later call of $\mathsf{FailedConstraints}$ relies on some v from Line 14 when we execute some $\texttt{FailedConstraints}(t)$. This means $v \neq x_{i_t}$, $\sigma_t(v) = \star$, and

$t \geq -T+n-1$ and thus the assignment on v is updated to $\star$ in the $-T \leq \mathsf{UpdTime}(v,t)$-th for iteration in Algorithm 4. Then the argument above goes through.

For Item (2), note that V^* is initialized as $\{t_0\}$. Upon Line 14, we have $\mathsf{UpdTime}(v,t) = \mathsf{UpdTime}(v,t+1)$ since $v \neq x_{i_t}$, and $\mathsf{UpdTime}(v,t+1)$ was added into V^* by Line 9-11 earlier.

Now we turn to Item (3). By Item (1), $E^* \subseteq E^{\mathsf{ext}}$. Meanwhile by Line 11 of Algorithm 5, E^* are hyperedges over vertex set V^*. Thus it suffices to show $H^* = (V^*, E^*)$ is connected. By Item (2), we only need to show vertices added during the while iterations in `FailedConstraints`(t) is connected to t. This can be proved by induction: when we find $C \in \mathcal{C}$ satisfying the condition in Line 8, let $u \in \mathsf{vbl}(C) \cap V_t$; then each $\mathsf{UpdTime}(v,t+1) \in e$ will be connected with $\mathsf{UpdTime}(u,t+1) \in e$, since either (A) $u = x_{i_t}$ and thus $\mathsf{UpdTime}(u,t+1) = t \in V^*$, or (B) $\mathsf{UpdTime}(u,t+1)$ was added into V^* and connected to t in an earlier time.

Finally we prove Item (4). As we discussed above, each time we call `FailedConstraints`(t), it implies the assignment on x_{i_t} is updated to $\star$ in the t-th for iteration in Algorithm 4. This means $\mathsf{Token}_t = \mathsf{False}$ and $\mathsf{SafeSampling}$ is performed. Thus by Algorithm 3, x_{i_t} is connected by falsified constraints to some $v \neq x_{i_t}$ that $\sigma_t(v) = \star$. This implies at least one round of Line 14 of `FailedConstraints`(t) will be executed. Thus the recursion continues until $t < -T+n-1$. By Item (2), there exists some $t_1 \in V^*$ with $t_1 < -T+n-1$. Meanwhile $t_0 \in V^*$ and $t_0 \geq -n$. By Item (3), there exists $e_0, e_1 \in E^*$ such that $t_0 \in e_0$ and $t_1 \in e_1$. Thus $t_{\mathsf{max}}(e_0) \geq t_0 \geq -n$ and $t_{\mathsf{min}}(e_1) \leq t_1 < -T+n-1$. □

Now we complete the proof of Proposition 3.5.

Proof of the Coalescence Part of Proposition 3.5. Since σ_0 is defined as the final returned assignment, we know $\mathcal{E}$ is essentially the event "there exists some $v^* \in V$ with $\sigma_0(v^*) = \star$". Then using Algorithm 5 we obtain $H^* = (V^*, E^*)$. By Item (3) and (4) of Lemma 3.17 and Fact 2.7, there exists $e_1, e_2, \ldots, e_\ell \in E^* \subseteq E^{\mathsf{ext}}$ with labels $C_1, C_2, \ldots, C_\ell$ such that

- $e_i \cap e_{i+1} \neq \emptyset$ for all $i \in [\ell-1]$, which implies $t_{\mathsf{min}}(e_i) \leq t_{\mathsf{max}}(e_{i+1})$;
- $e_i \cap e_j = \emptyset$ for all $i, j \in [\ell]$ with $|i-j| > 1$;
- $t_{\mathsf{max}}(e_1) \geq -n$ and $t_{\mathsf{min}}(e_\ell) < -T+n-1$.

Let $\mathcal{P}$ be the set of all possible sequence $e_1, \ldots, e_\ell \in E^{\mathsf{ext}}$ satisfying these conditions. Then we have

$$\begin{aligned}
|\mathcal{P}| &\leq \sum_{e_1 \in E^{\mathsf{ext}}} \Delta(H^{\mathsf{ext}})^{\ell-1} && \text{(enumerating } e_2, \ldots, e_\ell \text{ given } e_1\text{)} \\
&\leq \Delta(H^{\mathsf{ext}})^{\ell-1} \cdot n \cdot dk && \text{(} n \text{ for } t_{\mathsf{max}}(e_1),\ d \text{ for } C_1, \text{ and } k \text{ is by Item (3) of Fact 3.14)} \\
&\leq n \cdot d^\ell k^{2\ell-1}. && \text{(by Fact 3.16)}
\end{aligned}$$

Meanwhile

$$\begin{aligned}
t_{\mathsf{max}}(e_1) &= t_{\mathsf{max}}(e_\ell) + \sum_{i=1}^{\ell-1} \left(t_{\mathsf{max}}(e_i) - t_{\mathsf{max}}(e_{i+1})\right) \\
&\leq t_{\mathsf{min}}(e_\ell) + n - 1 + \sum_{i=1}^{\ell-1} \left(t_{\mathsf{min}}(e_i) + n - 1 - t_{\mathsf{max}}(e_{i+1})\right) && \text{(by Item (2) of Fact 3.14)} \\
&\leq (-T+n-2) + \ell \cdot (n-1),
\end{aligned}$$

which implies $\ell \geq -2 + \lceil T/(n-1) \rceil$. For convenience we set $\ell = -2 + \lceil T/(n-1) \rceil$.

Let $\mathcal{E}_{t,C}$ be the event "the assignment on x_{i_t} is updated to $\sigma_{\mathsf{False}}^{C|\boldsymbol{\pi}}(x_{i_t})$ or $\star$ in the t-th for iteration in Algorithm 4". Then by Item (1b) of Lemma 3.17, $\mathcal{E}_{t,C_i}$ happens for all i and all t in e_{C_i}. We analyze two possible updates separately.

- SafeSampling. Then $\mathbf{Pr}\left[\mathcal{E}_{t,C_i}\right] \le \frac{\left|\pi_v^{-1}(\sigma_{\mathsf{False}}^{C_i|\boldsymbol{\pi}}(v))\right|}{|\Omega_v|} + (|Q_v|-1)\cdot\gamma_v$.
- RejectionSampling. Then by Lemma 3.3, Fact 2.6, and Lemma 3.11, $\mathcal{E}_{t,C_i}$ happens with probability at most
$$\min\left\{1, \beta\cdot\frac{\left|\pi_v^{-1}(\sigma_{\mathsf{False}}^{C_i|\boldsymbol{\pi}}(v))\right|}{|\Omega_v|}\right\}.$$

Notice that if $|Q_v|=1$ then both bounds give 1 as the upper bound. Otherwise we have

$$\begin{aligned}
&\left(\frac{\left|\pi_v^{-1}(\sigma_{\mathsf{False}}^{C_i|\boldsymbol{\pi}}(v))\right|}{|\Omega_v|} + (|Q_v|-1)\cdot\gamma_v\right) - \beta\cdot\frac{\left|\pi_v^{-1}(\sigma_{\mathsf{False}}^{C_i|\boldsymbol{\pi}}(v))\right|}{|\Omega_v|}\\
&\ge (\beta-1)\cdot\left(-\frac{\left|\pi_v^{-1}(\sigma_{\mathsf{False}}^{C_i|\boldsymbol{\pi}}(v))\right|}{|\Omega_v|} + \max_{q\in Q_v}\left(1-\frac{\left|\pi_v^{-1}(q)\right|}{|\Omega_v|}\right)\right) \qquad \text{(by } |Q_v|\ge 2 \text{ and unpacking } \gamma_v\text{)}\\
&\ge 0. \quad \left(\text{since } |Q_v|\ge 2\text{, we have } 1-\frac{|\pi_v^{-1}(q)|}{|\Omega_v|} \ge \frac{\left|\pi_v^{-1}(\sigma_{\mathsf{False}}^{C_i|\boldsymbol{\pi}}(v))\right|}{|\Omega_v|} \text{ for any } q\in Q_v\setminus\left\{\pi_v^{-1}(\sigma_{\mathsf{False}}^{C_i|\boldsymbol{\pi}}(v))\right\}\right)
\end{aligned}$$

Thus the bound from SafeSampling dominates both and

$$\begin{aligned}
\mathbf{Pr}\left[\mathcal{E}\right] &\le \mathbf{Pr}\left[\mathcal{E}_{t,C_i}, \forall i\in[\ell], t\in C_i\right]\\
&\le \sum_{(e_1',\ldots,e_\ell')\in\mathcal{P}} \mathbf{Pr}\left[\mathcal{E}_{t,C_i'}, \forall i\in[\ell], t\in C_i'\right] && (C_i' \text{ is the label of } e_i')\\
&\le \sum_{(e_1',\ldots,e_\ell')\in\mathcal{P}} \mathbf{Pr}\left[\mathcal{E}_{t,C_i'}, \forall i\in\{1,3,5,\ldots\}, t\in C_i'\right]\\
&\le \sum_{(e_1',\ldots,e_\ell')\in\mathcal{P}}\ \prod_{i=1,3,5,\ldots}\ \prod_{v\in\mathsf{vbl}(C_i')}\left(\frac{\left|\pi_v^{-1}(\sigma_{\mathsf{False}}^{C_i'|\boldsymbol{\pi}}(v))\right|}{|\Omega_v|} + (|Q_v|-1)\cdot\gamma_v\right)\\
& && \text{(since } (e_i')_{i\equiv 1 \mod 2} \text{ are pairwise disjoint)}\\
&\le |\mathcal{P}|\cdot\lambda^{\lceil \ell/2\rceil} \le n\cdot d^{-2+\left\lceil\frac{T}{n-1}\right\rceil}\cdot k^{-5+2\left\lceil\frac{T}{n-1}\right\rceil}\cdot\lambda^{-1+\left\lceil\frac{1}{2}\left\lceil\frac{T}{n-1}\right\rceil\right\rceil}\\
&\le dkn\cdot\left(d^2k^4\lambda\right)^{\frac{T}{2(n-1)}-1}. \qquad \square
\end{aligned}$$

3.3 The Distribution after BoundingChain Subroutines

Recall in `AtomicCSPSampling`$(\Phi,\boldsymbol{\pi})$ (Algorithm 2), we keep doubling T and performing the corresponding `BoundingChain`$(\Phi,\boldsymbol{\pi},-T,r_{-T},\ldots,r_{-1})$ until the returned partial assignment has no $\star$. Thus before we present the FinalSampling subroutine, we pause for now to analyze these BoundingChain calls *in a whole*.

Definition 3.18 (Projected Uniform Distribution)**.** Given an atomic CSP $\Phi = \left(V, (\Omega_v)_{v\in V}, \mathcal{C}\right)$ and projections $\boldsymbol{\pi} = (\pi_v)_{v\in V}, \pi_v\colon \Omega_v \to Q_v$, define the *projected uniform distribution* $\mu^{\Phi|\boldsymbol{\pi}} \in \mathbb{R}^\Lambda$ by

$$\mu^{\Phi|\boldsymbol{\pi}}(\sigma) = \frac{\left|\sigma_{\mathsf{True}}^{\mathcal{C}} \cap \left(\prod_{v\in V}\pi_v^{-1}(\sigma(v))\right)\right|}{\left|\sigma_{\mathsf{True}}^{\mathcal{C}}\right|} \quad \text{for all } \sigma\in\Lambda.$$

We will show the distribution after all BoundingChain subroutines is exactly $\mu^{\Phi|\boldsymbol{\pi}}$.

Proposition 3.19. *If $e \cdot \alpha \cdot \Delta \le 1$, $d^2 k^4 \lambda < 1$, and $e \cdot \Delta^2 \rho \le 1/32$, then the* while *iterations in* `AtomicCSPSampling`$(\Phi, \boldsymbol{\pi})$ *halts almost surely and the final partial assignment $\sigma_{\mathsf{Partial}}$ has distribution $\mu^{\Phi|\boldsymbol{\pi}}$.*

We introduce and analyze the following `SystematicScan`$(\Phi, \boldsymbol{\pi}, \sigma_{\mathsf{in}}, L, R, r_L, \ldots, r_R)$ algorithm, then couple it with BoundingChain.

Algorithm 6: The SystematicScan algorithm

Input: an atomic CSP $\Phi = \left(V, (\Omega_v)_{v\in V}, \mathcal{C}\right)$, projections $\boldsymbol{\pi} = (\pi_v)_{v\in V}, \pi_v\colon \Omega_v \to Q_v$, a partial assignment $\sigma_{\mathsf{in}} \in \prod_{v\in V} Q_v$, starting time L, stopping time R, and randomness tapes $r_L, \ldots, r_R$.

Output: a partial assignment $\sigma \in \prod_{v\in V} Q_v$.

Initialize $\sigma \leftarrow \sigma_{\mathsf{in}}$

for $t = L$ **to** R **do** `/*` Assume $V = \{x_0, \ldots, x_{n-1}\}$. `*/`

  $i_t \leftarrow t \mod n$, and $\sigma(x_{i_t}) \leftarrow \star$ `/*` Update $\sigma(x_i)$ in this round. `*/`

  $\Phi_t = \left(V_t, \left(\pi_v^{-1}(\sigma(v))\right)_{v\in V_t}, \mathcal{C}_t\right) \leftarrow$ `Component`$(\Phi, \boldsymbol{\pi}, \sigma, x_{i_t})$

  `/* Ignore the returned Token since it is always True here. */`

  $\sigma' \leftarrow$ `RejectionSampling`(Φ_t, r_t)

  Update $\sigma(x_{i_t}) \leftarrow \pi_{x_{i_t}}(\sigma'(x_{i_t}))$

end

return σ

3.3.1 Convergence of SystematicScan

We first show SystematicScan converges to $\mu^{\Phi|\boldsymbol{\pi}}$. We set up basic notations for Markov chains.

Let Λ be a finite state space. We view any distribution μ over Λ as a *horizontal* vector in $\mathbb{R}^\Lambda$ where $\sum_{a\in\Lambda} \mu(a) = 1$ and $\mu(a) \ge 0$ holds for all $a \in \Lambda$. We denote $1_a \in \mathbb{R}^\Lambda$ as the point distribution of $a \in \Lambda$, i.e., $1_a(b) = 1$ if $a = b$.

A *Markov chain* $(X_t)_{t\ge 0}$ over Λ is given by its transition matrices $(\mathsf{P}_t)_{t\ge 0}$ where each $\mathsf{P}_t \in \mathbb{R}^{\Lambda\times\Lambda}$ has non-negative entries and $\sum_{b\in\Lambda} \mathsf{P}_t(a, b) \equiv 1$ for all $a \in \Lambda$. Then $X_t = X_0 \mathsf{P}_0 \mathsf{P}_1 \cdots \mathsf{P}_{t-1}$ where $X_0 \in \mathbb{R}^\Lambda$ is the starting distribution. In particular,

- if $\mathsf{P}_t \equiv \mathsf{P}$ for all $t \ge 0$, then $(X_t)_{t\ge 0}$ is a *time homogeneous Markov chain* given by P;
- if $(\mathsf{P}_t)_{t\ge 0}$ are possibly different, then $(X_t)_{t\ge 0}$ is a *time inhomogeneous Markov chain.*

Assume $(X_t)_{t\ge 0}$ is a time homogeneous Markov chain over Λ given by transition matrix $\mathsf{P} \in \mathbb{R}^{\Lambda\times\Lambda}$. We say P is

- *irreducible* if for any $a, b \in \Lambda$, there exists some integer $t \ge 0$ such that $\mathsf{P}^t(a, b) > 0$;
- *aperiodic* if for any $a \in \Lambda$, $\mathsf{gcd}\left\{\text{integer } t > 0 \,\middle|\, \mathsf{P}^t(a, a) > 0\right\} = 1$;[12]
- *stationary with respect to distribution* μ if $\mu\mathsf{P} = \mu$;
- *reversible with respect to distribution* μ if $\mu(a)\mathsf{P}(a, b) = \mu(b)\mathsf{P}(b, a)$ holds for all $a, b \in \Lambda$.

Here we note the following two classical results.

[12] gcd stands for greatest common divisor.

Fact 3.20 (e.g., [LP17, Proposition 1.20]). *If P is reversible with respect to μ, then it is also stationary with respect to μ.*

Theorem 3.21 (The Convergence Theorem, e.g., [LP17, Theorem 4.9]). *Suppose $(X_t)_{t\ge 0}$ is an irreducible and aperiodic time homogeneous Markov chain over finite state space Λ with stationary distribution μ and transition matrix P. Then for any X_0, we have $\lim_{t\to+\infty} X_t = \mu$.*[13]

Now we turn to $\mathsf{SystematicScan}$. For our purpose we fix Λ to be $\prod_{v\in V} Q_v$, and follow the notation convention as in Algorithm 6: $V = \{x_0,\ldots,x_{n-1}\}$ and $i_t = t \mod n$.

Definition 3.22 (One-step Transition Matrix). For any $i \in \{0,\ldots,n-1\}$, define the *one-step transition matrix* on $x_i \in V$ as $\mathsf{P}_i \in \mathbb{R}^{\Lambda\times\Lambda}$ where

$$\mathsf{P}_i(\sigma_1,\sigma_2) = \begin{cases} \frac{\left|\left\{\sigma\in\sigma^{\mathcal{C}}_{\mathsf{True}} \,\middle|\, \pi_v(\sigma(v))=\sigma_2(v),\forall v\in V\right\}\right|}{\left|\left\{\sigma\in\sigma^{\mathcal{C}}_{\mathsf{True}} \,\middle|\, \pi_v(\sigma(v))=\sigma_2(v),\forall v\in V\setminus\{x_i\}\right\}\right|} & \sigma_1(v)=\sigma_2(v), \forall v\in V\setminus\{x_i\}, \\ 0 & \text{otherwise.} \end{cases}$$

We remark that by Lemma 3.4, if $e\cdot\alpha\cdot\Delta\le 1$ then each P_i is well-defined, i.e., the denominators are all non-zero.

Fact 3.23. *If $e\cdot\alpha\cdot\Delta \le 1$ and $L\le R$, then $\mathtt{SystematicScan}(\Phi,\boldsymbol{\pi},\sigma_{\mathsf{in}},L,R,r_L,\ldots,r_R)$ halts almost surely over random $r_L,\ldots,r_R$. Moreover the output distribution is $\mu = 1_{\sigma_{\mathsf{in}}}\mathsf{P}_{i_L}\mathsf{P}_{i_{L+1}}\cdots\mathsf{P}_{i_R}$.*

Proof. By Lemma 3.3 and Lemma 3.4, each $\mathsf{RejectionSampling}$ halts almost surely. Moreover by Fact 2.6 and Item (4) of Lemma 3.3, we know for any fixed $\sigma\in\Lambda$ and $t\in\{L,\ldots,R\}$, $\sigma(x_{i_t})$ is updated to $q\in Q_{x_{i_t}}$ with probability

$$\frac{\left|\left\{\sigma_1\in\sigma^{\mathcal{C}_t}_{\mathsf{True}} \,\middle|\, \sigma_1(x_{i_t})=q\right\}\right|}{\left|\sigma^{\mathcal{C}_t}_{\mathsf{True}}\right|} = \frac{\left|\left\{\sigma_2\in\sigma^{\mathcal{C}}_{\mathsf{True}} \,\middle|\, \pi_v(\sigma_2(v))=\sigma'(v),\forall v\in V\right\}\right|}{\left|\left\{\sigma_2\in\sigma^{\mathcal{C}}_{\mathsf{True}} \,\middle|\, \pi_v(\sigma_2(v))=\sigma'(v),\forall v\in V\setminus\{x_{i_t}\}\right\}\right|},$$

where σ' differs from σ only on x_{i_t} and $\sigma'(x_{i_t})=q$. Thus the partial assignment after this update is distributed as $1_\sigma\mathsf{P}_{i_t}$. Repeatedly applying the updates, we obtain the desired the result. □

Fact 3.24. *If $e\cdot\alpha\cdot\Delta\le 1$, then for any $i\in\{0,\ldots,n-1\}$ and any $\sigma_1,\sigma_2\in\Lambda$ with $\sigma_1(v)=\sigma_2(v)$ for all $v\in V\setminus\{x_i\}$, we have $\mathsf{P}_i(\sigma_1,\sigma_2)>0$.*

Proof. By Definition 3.22, it suffices to show $\left\{\sigma\in\sigma^{\mathcal{C}}_{\mathsf{True}} \,\middle|\, \pi_v(\sigma(v))=\sigma_2(v),\forall v\in V\right\}\ne\emptyset$. Let $\Phi' = \left(V, \left(\pi_v^{-1}(\sigma_2(v))\right)_{v\in V}, \mathcal{C}\right)$. Then it is equivalent to Φ' has satisfying assignment. Since $p(\Phi')\le\alpha(\Phi)$ and $\Delta(\Phi')\le\Delta(\Phi)$, this follows immediately from Theorem 2.3. □

Fact 3.25. *If $e\cdot\alpha\cdot\Delta\le 1$, then for any $\sigma_1,\sigma_2\in\Lambda$ and any sequence $i_1,i_2,\ldots,i_m\in\{0,\ldots,n-1\}$ of finite length m, we have $\mu^{\Phi|\boldsymbol{\pi}}\mathsf{P}_{i_1}\cdots\mathsf{P}_{i_m} = \mu^{\Phi|\boldsymbol{\pi}}$.*

Proof. Since $\mu^{\Phi|\boldsymbol{\pi}}\mathsf{P}_{i_1}\cdots\mathsf{P}_{i_m} = \left(\mu^{\Phi|\boldsymbol{\pi}}\mathsf{P}_{i_1}\right)\mathsf{P}_{i_2}\cdots\mathsf{P}_{i_m}$, it suffices to show for $m=1$ and then apply induction. By Fact 3.20, it suffices to show for each $i\in\{0,\ldots,n-1\}$, P_i is reversible with respect to $\mu^{\Phi|\boldsymbol{\pi}}$, i.e.,

$$\mu^{\Phi|\boldsymbol{\pi}}(\sigma_1)\mathsf{P}_i(\sigma_1,\sigma_2) = \mu^{\Phi|\boldsymbol{\pi}}(\sigma_2)\mathsf{P}_i(\sigma_2,\sigma_1) \quad \text{for any } \sigma_1,\sigma_2\in\Lambda. \tag{4}$$

[13] The convergence is entry-wise.

If σ_1 differs from σ_2 in entries other than x_{i_1}, then $\mathsf{P}_i(\sigma_1,\sigma_2) = \mathsf{P}_i(\sigma_2,\sigma_1) = 0$ and (4) holds trivially. Otherwise

$$\begin{aligned}
&\mu^{\Phi|\boldsymbol{\pi}}(\sigma_1)\mathsf{P}_i(\sigma_1,\sigma_2)\\
&= \frac{\left|\sigma^{\mathcal{C}}_{\mathsf{True}} \cap \left(\prod_{v\in V}\pi_v^{-1}(\sigma_1(v))\right)\right|}{\left|\sigma^{\mathcal{C}}_{\mathsf{True}}\right|} \cdot \frac{\left|\left\{\sigma\in\sigma^{\mathcal{C}}_{\mathsf{True}} \,\middle|\, \pi_v(\sigma(v)) = \sigma_2(v), \forall v\in V\right\}\right|}{\left|\left\{\sigma\in\sigma^{\mathcal{C}}_{\mathsf{True}} \,\middle|\, \pi_v(\sigma(v)) = \sigma_2(v), \forall v\in V\setminus\{x_i\}\right\}\right|}\\
&= \frac{\left|\left\{\sigma\in\sigma^{\mathcal{C}}_{\mathsf{True}} \,\middle|\, \pi_v(\sigma(v)) = \sigma_2(v), \forall v\in V\right\}\right|}{\left|\sigma^{\mathcal{C}}_{\mathsf{True}}\right|} \cdot \frac{\left|\sigma^{\mathcal{C}}_{\mathsf{True}} \cap \left(\prod_{v\in V}\pi_v^{-1}(\sigma_1(v))\right)\right|}{\left|\left\{\sigma\in\sigma^{\mathcal{C}}_{\mathsf{True}} \,\middle|\, \pi_v(\sigma(v)) = \sigma_2(v), \forall v\in V\setminus\{x_i\}\right\}\right|}\\
&= \frac{\left|\sigma^{\mathcal{C}}_{\mathsf{True}} \cap \left(\prod_{v\in V}\pi_v^{-1}(\sigma_2(v))\right)\right|}{\left|\sigma^{\mathcal{C}}_{\mathsf{True}}\right|} \cdot \frac{\left|\left\{\sigma\in\sigma^{\mathcal{C}}_{\mathsf{True}} \,\middle|\, \pi_v(\sigma(v)) = \sigma_1(v), \forall v\in V\right\}\right|}{\left|\left\{\sigma\in\sigma^{\mathcal{C}}_{\mathsf{True}} \,\middle|\, \pi_v(\sigma(v)) = \sigma_1(v), \forall v\in V\setminus\{x_i\}\right\}\right|}\\
&= \mu^{\Phi|\boldsymbol{\pi}}(\sigma_2)\mathsf{P}_i(\sigma_2,\sigma_1).
\end{aligned}$$ ☐

Fact 3.23 shows $\mathsf{SystematicScan}$ is a *time inhomogeneous Markov chain.* General theory regarding time inhomogeneous Markov chains can be much more complicated but luckily we can embed this one into a time homogeneous Markov chain.

Lemma 3.26. *Assume $e\cdot\alpha\cdot\Delta \le 1$. Let L and $\sigma_{\mathsf{in}}\in\Lambda$ be arbitrary. Define $\mu_R = 1_{\sigma_{\mathsf{in}}}\mathsf{P}_{i_L}\cdots\mathsf{P}_{i_R}$. Then $\lim_{R\to+\infty}\mu_R = \mu^{\Phi|\boldsymbol{\pi}}$.*

Proof. Let $\mathsf{F} = \mathsf{P}_{i_L}\cdots\mathsf{P}_{i_{L+n-1}}$. Since $i_t = t \mod n$, the one-step transition matrices repeatedly applied to $1_{\sigma_{\mathsf{in}}}$ has period n. Hence $\mu_R = 1_{\sigma_{\mathsf{in}}}\mathsf{F}^m$ if $R = L + m\cdot n - 1$ and $m \ge 1$. Let $Y_0 = 1_{\sigma_{\mathsf{in}}}$ and $Y_t = Y_0\mathsf{F}^i$ for $t\ge 1$, then $(Y_t)_{t\ge 0}$ is a time homogeneous Markov chain with transition matrix F. Here we verify the following properties of F.

- Stationary with respect to $\mu^{\Phi|\boldsymbol{\pi}}$. This follows immediately from Fact 3.25.
- Aperiodic. By Fact 3.24, for any $i\in\{0,\ldots,n-1\}$ and any $\sigma\in\Lambda$ we have $\mathsf{P}_i(\sigma,\sigma) > 0$. Thus $\mathsf{F}(\sigma,\sigma) > 0$ which implies F is aperiodic.
- Irreducible. Let $\sigma_1,\sigma_2\in\Lambda$ be arbitrary. For each $j\in\{0,\ldots,n\}$, define $\sigma^j\in\Lambda$ by

$$\sigma^j(x_{i'}) = \begin{cases}\sigma_2(x_{i'}) & i'\in\{i_L,\ldots,i_{L+j-1}\},\\ \sigma_1(x_{i'}) & \text{otherwise.}\end{cases}$$

  Then $\sigma_1 = \sigma^0$ and $\sigma_2 = \sigma^n$. Moreover $\mathsf{P}_{i_{L+j}}(\sigma^j,\sigma^{j+1}) > 0$ for all $j\in\{0,\ldots,n-1\}$ by Fact 3.24. Thus $\mathsf{F}(\sigma_1,\sigma_2) \ge \mathsf{P}_{i_L}(\sigma^0,\sigma^1)\mathsf{P}_{i_{L+1}}(\sigma^1,\sigma^2)\cdots\mathsf{P}_{i_{L+n-1}}(\sigma^{n-1},\sigma^n) > 0$.

Therefore by Theorem 3.21, $\lim_{m\to+\infty}\mu_{L+m\cdot n-1} = \lim_{t\to+\infty} Y_t = \mu^{\Phi|\boldsymbol{\pi}}$. Since each P_i is stationary with respect to $\mu^{\Phi|\boldsymbol{\pi}}$ by Fact 3.25, for any finite integer $o\ge 0$

$$\lim_{m\to+\infty}\mu_{L+m\cdot n-1+o} = \left(\lim_{m\to+\infty}\mu_{L+m\cdot n-1}\right)\mathsf{P}_{i_L}\cdots\mathsf{P}_{i_{L+o-1}} = \mu^{\Phi|\boldsymbol{\pi}}\mathsf{P}_{i_L}\cdots\mathsf{P}_{i_{L+o-1}} = \mu^{\Phi|\boldsymbol{\pi}}.$$

Hence $\lim_{R\to+\infty}\mu_R = \mu^{\Phi|\boldsymbol{\pi}}$. ☐

3.3.2 Coupling from the Past and the Bounding Chain

We have showed $\mathsf{SystematicScan}$ *converges* to distribution $\mu^{\Phi|\boldsymbol{\pi}}$, but to obtain a sample distributed *exactly* according to $\mu^{\Phi|\boldsymbol{\pi}}$ we need to run for infinite time. The trick for making it finite is to think *backwards.* That is the idea of *coupling from the past* [PW96]; then the *bounding chain* [Hub98, HN99] is used to make the process more computationally efficient.

Let $\mathsf{P} \in \mathbb{R}^{\Lambda \times \Lambda}$ be some transition matrix. We say $f \colon \Lambda \times [0,1] \to \Lambda$ is a *coupling* of P if for all $a, b \in \Lambda$, $\mathbf{Pr}_{r \sim [0,1]}\left[f(a,r) = b\right] = \mathsf{P}(a,b)$. We use random function $f^r \colon \Lambda \to \Lambda$ to denote the coupling f with randomness r, i.e., $f^r(a) = f(a,r)$ for all $a \in \Lambda$.

Recall our definition of P_i from Definition 3.22 and $i_t = t \mod n$ from Algorithm 6.

Lemma 3.27 (Coupling from the Past)**.** *Let $f_t \colon \Lambda \times [0,1] \to \Lambda$ be a coupling of P_{i_t} for all $t \in \mathbb{Z}$. Define random functions $F_{L,R} \colon \Lambda \to \Lambda$ over random $(r_t)_{t \in \mathbb{Z}}$ for $-\infty < L \le R < +\infty$ as*

$$F_{L,R}(a) = f_R^{r_R}\left(f_{R-1}^{r_{R-1}}\left(\cdots f_L^{r_L}(a) \cdots\right)\right) \quad \text{for all } a \in \Lambda.$$

Let $M \ge 1$ be the smallest integer such that $F_{-M,-1}$ is a constant function. Let $A = F_{-M,-1}(\Lambda)$ be the corresponding constant. Then $F_{-M',-1}(\Lambda) \equiv A$ for any $M' > M$, and A is distributed as $\mu^{\Phi|\boldsymbol{\pi}}$ if $M < +\infty$ almost surely.

Proof. Since $\mathsf{P}_i = \mathsf{P}_{i+n}$, for any integer $\ell \ge 1$ and all $a, b \in \Lambda$ we have

$$\mathbf{Pr}\left[F_{-\ell \cdot n,-1}(a,b)\right] = 1_a \left(\mathsf{P}_{-n}\mathsf{P}_{-n+1} \cdots \mathsf{P}_{-1}\right)^{\ell} 1_b^{\top} = 1_a \left(\mathsf{P}_0\mathsf{P}_1 \cdots \mathsf{P}_{n-1}\right)^{\ell} 1_b^{\top} = \mathbf{Pr}\left[F_{0,\ell \cdot n-1}(a,b)\right].$$

Thus for any $a, b \in \Lambda$ by Lemma 3.26,

$$\lim_{\ell \to +\infty} \mathbf{Pr}\left[F_{-\ell \cdot n,-1}(a) = b\right] = \lim_{\ell \to +\infty} \mathbf{Pr}\left[F_{0,\ell \cdot n-1}(a) = b\right] = \mu^{\Phi|\boldsymbol{\pi}}(b).$$

On the other hand for $M' > M$, $F_{-M',-1}(\Lambda) = F_{-M,-1}\left(F_{-M',-M-1}(\Lambda)\right) = A$. Then by $M < +\infty$ almost surely, we have

$$\mathbf{Pr}\left[A = b\right] = \lim_{\ell \to +\infty} \mathbf{Pr}\left[F_{-\ell \cdot n,-1}(a) = b\right] = \mu^{\Phi|\boldsymbol{\pi}}(b). \qquad \square$$

Therefore to obtain a perfect sample from $\mu^{\Phi|\boldsymbol{\pi}}$ we only need to (1) design a coupling, then (2) sample and fix random $(r_t)_{t \ge 0}$, and lastly (3) find some $M' \ge 1$ such that $F_{-M',-1}$ is a constant function and output the constant. Thus we now show our Algorithm 4 provides a coupling for (1) and an efficient way to check (3).

Let $\Lambda^* = \prod_{v\in V}\left(\{\star\}\cup Q_v\right)$. Define $g_i\colon \Lambda^*\times[0,1]\to\Lambda^*$ for each $i\in\{0,\dots,n-1\}$ as follows.

Algorithm 7: A coupling $g_i\colon \Lambda^*\times[0,1]\to\Lambda^*$ for each $i\in\{0,\dots,n-1\}$

Input: a partial assignment $\sigma_{\mathsf{in}}\in\Lambda^*$, and randomness $r\in[0,1]$.
Output: a partial assignment $\sigma_{\mathsf{out}}\in\Lambda^*$.
Data: an atomic CSP $\Phi = \left(V, (\Omega_v)_{v\in V}, \mathcal{C}\right)$, projections $\boldsymbol{\pi} = (\pi_v)_{v\in V}, \pi_v\colon \Omega_v\to Q_v$, and an index $i\in\{0,\dots,n-1\}$.

1 Initialize $\sigma_{\mathsf{out}}\leftarrow\sigma_{\mathsf{in}}$ `/* Assume` $V=\{x_0,\dots,x_{n-1}\}$. `*/`
2 Define distribution $\mathcal{D}$ over $Q_{x_i}\cup\{\star\}$ by

$$\mathcal{D}(q) = \begin{cases} \max\left\{0, \frac{\left|\pi_{x_i}^{-1}(q)\right|}{|\Omega_{x_i}|} - \gamma_{x_i}\right\} & q\in Q_{x_i}, \\ 1-\sum_{q\in Q_{x_i}}\max\left\{0, \frac{\left|\pi_{x_i}^{-1}(q)\right|}{|\Omega_{x_i}|} - \gamma_{x_i}\right\} & q=\star. \end{cases}$$

3 Sample c from $\mathcal{D}$
4 **if** $c\neq\star$ **then** Update $\sigma_{\mathsf{out}}(x_i)\leftarrow c$ and **return** σ_{out}
5 **else** $\sigma_{\mathsf{out}}(x_i)\leftarrow\star$
6 $(\Phi', \mathsf{Token})\leftarrow \mathtt{Component}(\Phi,\boldsymbol{\pi},\sigma_{\mathsf{out}},x_i)$ where $\Phi' = \left(V', \left(\pi_v^{-1}(\sigma_{\mathsf{out}}(v))\right)_{v\in V'}, \mathcal{C}'\right)$
7 **if** $\mathsf{Token}=\mathsf{False}$ **then return** σ_{out}
8 Define distribution $\mathcal{D}^*$ over Q_{x_i} by

$$\mathcal{D}^*(q) = \frac{\left|\left\{\sigma\in\sigma_{\mathsf{True}}^{\mathcal{C}'}\;\middle|\;\pi_{x_i}(\sigma(x_i))=q\right\}\right|}{\left|\sigma_{\mathsf{True}}^{\mathcal{C}'}\right|}\quad\text{for } q\in Q_{x_i}.$$

9 Define distribution $\mathcal{D}'$ over Q_{x_i} by $\mathcal{D}'(q) = \frac{\mathcal{D}^*(q)-\mathcal{D}(q)}{\mathcal{D}(\star)}$ for $q\in Q_{x_i}$
10 Sample c' from $\mathcal{D}'$ and update $\sigma_{\mathsf{out}}(x_i)\leftarrow c'$ and **return** σ_{out}

We say $\sigma_1\Subset\sigma_2$ for some $\sigma_1,\sigma_2\in\Lambda^*$ if $\sigma_2(v)$ equals $\star$ or $\sigma_1(v)$ for all $v\in V$.

Observation 3.28. When $\sigma_1,\sigma_2\in\Lambda$, $\sigma_1\Subset\sigma_2$ iff $\sigma_1=\sigma_2$.

Now we verify the following properties of Algorithm 7.

Lemma 3.29. *If $e\cdot\alpha\cdot\Delta\le 1$ then the following holds for $g_i\colon\Lambda^*\times[0,1]\to\Lambda^*$ from Algorithm 7.*

(1) $\mathcal{D}^*$ and $\mathcal{D}'$ on Line 8-9 are valid distributions when $\mathsf{Token}=\mathsf{True}$.

(2) $g_i(\sigma_1,r)\Subset g_i(\sigma_2,r)$ holds for any $r\in[0,1]$ and $\sigma_1,\sigma_2\in\Lambda^*$ with $\sigma_1\Subset\sigma_2$.

(3) If $\sigma_{\mathsf{in}}\in\Lambda$, then $\mathcal{D}^*(q)=\mathsf{P}_i(\sigma_{\mathsf{in}},\sigma_{\mathsf{in}}^q)$ where σ_{in}^q is σ_{in} with the x_i-th entry replaced by q.

(4) g_i restricted on Λ is a coupling of P_i.

(5) For any $t\equiv i \mod n$, g_i is the same update procedure as the t-th for iteration in Algorithm 4.

Proof. First we prove Item (1). By Lemma 3.4, $\mathcal{D}^*$ is a valid distribution. For $\mathcal{D}'$, note that $\sum_{q\in Q_{x_i}}(\mathcal{D}^*(q)-\mathcal{D}(q)) = 1-(1-\mathcal{D}(\star)) = \mathcal{D}(\star)$. Thus it suffices to show $\mathcal{D}^*(q)\ge\mathcal{D}(q)$ for all $q\in Q_{x_i}$. If $\mathcal{D}(q)=0$ then it is trivially true. Otherwise $\frac{\left|\pi_{x_i}^{-1}(q)\right|}{|\Omega_{x_i}|}\ge\gamma_{x_i}$, then

$$\mathcal{D}^*(q)-\mathcal{D}(q) = 1-\mathcal{D}(q)-\sum_{q'\in Q_v\setminus\{q\}}\mathcal{D}^*(q')$$

$$\geq 1 - \left(\frac{\left|\pi_{x_i}^{-1}(q)\right|}{|\Omega_{x_i}|} - \gamma_{x_i}\right) - \beta \sum_{q' \in Q_v \setminus \{q\}} \frac{\left|\pi_{x_i}^{-1}(q')\right|}{|\Omega_{x_i}|} \qquad \text{(by Lemma 3.11)}$$

$$= \gamma_{x_i} - (\beta - 1)\left(1 - \frac{\left|\pi_{x_i}^{-1}(q)\right|}{|\Omega_{x_i}|}\right)$$

$$\geq 0. \qquad \text{(by unpacking } \gamma_{x_i}\text{)}$$

For Item (2), we fall into one of the following cases, all of which satisfies $g_i(\sigma_1, r) \in g_i(\sigma_2, r)$.

- $\boxed{c \neq \star.}$ Then both $\sigma_1(x_i)$ and $\sigma_2(x_i)$ is updated to c.
- $\boxed{c = \star \text{ and } \mathsf{Token} = \mathsf{False} \text{ for } \sigma_2.}$ Then $\sigma_2(x_i)$ is updated to $\star$.
- $\boxed{c = \star \text{ and } \mathsf{Token} = \mathsf{True} \text{ for } \sigma_2.}$ Since $\sigma_1 \in \sigma_2$, Token also equals True for σ_1. Moreover they get the same CSP from Line 6. Thus $\sigma_1(x_i)$ and $\sigma_2(x_i)$ are updated to the same value c'.

Item (3) follows from Item (4) of Lemma 3.3 and Definition 3.22. Then Item (4) follows from $\mathcal{D}^*(q) = \mathcal{D}'(q) \cdot \mathcal{D}(\star) + \mathcal{D}(q)$ where $\mathcal{D}(q)$ is from Line 3-4 and $\mathcal{D}'(q) \cdot \mathcal{D}(\star)$ is from Line 9-10.

Finally we prove Item (5). To obtain the pseudocode in Algorithm 4, we reorganize Algorithm 7 by first call Component, then based on the value of Token we either (A) sample c only, or (B) sample both c and c'. The former is SafeSampling, and the latter, executed jointly, is exactly RejectionSampling as we analyzed for Item (3) and (4). □

One more ingredient we need is the following well-known Borel-Cantelli theorem.

Theorem 3.30 (Borel-Cantelli Theorem, e.g., [GS01, Section 7.3])**.** *Let T be a non-negative random variable. If $\sum_{i=0}^{+\infty} \mathbf{Pr}[T > i] < +\infty$ then $T < +\infty$ almost surely.*

Now we are ready to prove Proposition 3.19.

Proof of Proposition 3.19. Assume the while iterations in Algorithm 2 stop at $T = T_{\mathsf{Final}}$ or $T_{\mathsf{Final}} = +\infty$ if iterations never end. Since each iteration halts almost surely by Proposition 3.5, T_{Final} is well-defined almost surely.

Let $V = (x_0, \ldots, x_{n-1})$ and define $i_t = t \mod n$ for all $t \in \mathbb{Z}$ as in Algorithm 4. Define random functions $G_{L,R} \colon \Lambda^* \to \Lambda^*$ over random $(r_t)_{t \in \mathbb{Z}}$ for $-\infty < L \leq R < +\infty$ as

$$G_{L,R}(a) = g_R^{r_R}\left(g_{R-1}^{r_{R-1}}\left(\cdots g_L^{r_L}(a) \cdots\right)\right) \quad \text{for all } a \in \Lambda^*,$$

where $g_t = g_{i_t} \colon \Lambda^* \times [0,1] \to \Lambda^*$ comes from Algorithm 7 and $g_t^r(\cdot) = g_t(\cdot, r)$.

Let $M \geq 1$ be the smallest integer such that $G_{-M,-1}$ is a constant function on Λ, i.e., $G_{-M,-1}(\Lambda) \equiv A$. By Item (4) of Lemma 3.29 and Lemma 3.27, $G_{-M_1',-1}(\Lambda) \equiv A$ for all $M_1' \geq M_1$, and A is distributed as $\mu^{\Phi|\boldsymbol{\pi}}$ if $M_1 < +\infty$ almost surely.

Let $M_2 \geq 1$ be the smallest integer such that $G_{-M_2,-1}(\star^n) \in \Lambda$. Iteratively applying Item (2) of Lemma 3.29, we know $G_{-M_2,-1}(\Lambda) \in G_{-M_2,-1}(\star^n)$. Then by Observation 3.28 $G_{-M_2,-1}(\Lambda)$ is constant. Thus $M_2 \geq M_1$ and $G_{-M_2,-1}(\star^n) = A$.

Now by Item (5) of Lemma 3.29, $\texttt{BoundingChain}(\Phi, \boldsymbol{\pi}, -T, r_{-T}, \ldots, r_{-1})$ equals $G_{-T,-1}(\star^n)$, which means $T_{\mathsf{Final}} \geq M_2$. Thus the final partial assignment $\sigma_{\mathsf{Partial}}$ equals A, which has distribution $\mu^{\Phi|\boldsymbol{\pi}}$ provided $T_{\mathsf{Final}} < +\infty$ almost surely.

Now we only need to show $T_{\mathsf{Final}} < +\infty$ almost surely. Note that either $T_{\mathsf{Final}} = 1$ or, by Algorithm 2, $\texttt{BoundingChain}(\Phi, \boldsymbol{\pi}, -T_{\mathsf{Final}}/2, r_{-T_{\mathsf{Final}}/2}, \ldots, r_{-1}) = G_{-T_{\mathsf{Final}}/2,-1}(\star^n) \notin \Lambda$. Thus

$T_{\mathsf{Final}} \le 2 \cdot M_2$, which means it suffices to show $M_2 < +\infty$ almost surely. By Item (4) of Lemma 3.29 and the analysis above, $G_{-i,-1}(\star^n) = A \in \Lambda$ for all $i \ge M_2$; thus

$$
\begin{aligned}
\sum_{i=0}^{+\infty} \mathbf{Pr}\left[M_2 > i\right] &\le 2n - 1 + \sum_{i=2n-1}^{+\infty} \mathbf{Pr}\left[G_{-i,-1}(\star^n) \notin \Lambda\right] \\
&= 2n - 1 + \sum_{i=2n-1}^{+\infty} \mathbf{Pr}\left[\texttt{BoundingChain}(\Phi, \boldsymbol{\pi}, -i, r_{-i}, \ldots, r_{-1}) \notin \Lambda\right] \\
&\le 2n - 1 + \sum_{i=2n-1}^{+\infty} dkn \cdot \left(d^2 k^4 \lambda\right)^{\frac{i}{2(n-1)} - 1} && \text{(by Proposition 3.5)} \\
&< +\infty, && \text{(since } d^2 k^4 \lambda < 1\text{)}
\end{aligned}
$$

as desired by Theorem 3.30. $\square$

3.4 The FinalSampling Subroutine

Finally we give the missing $\texttt{FinalSampling}(\Phi, \boldsymbol{\pi}, \sigma_{\mathsf{Partial}})$ subroutine, which uniformly completes a partial assignment $\sigma_{\mathsf{Partial}}$.

Algorithm 8: The FinalSampling subroutine

Input: an atomic CSP $\Phi = \left(V, (\Omega_v)_{v \in V}, \mathcal{C}\right)$, projections $\boldsymbol{\pi} = (\pi_v)_{v \in V}, \pi_v \colon \Omega_v \to Q_v$, and a partial assignment $\sigma_{\mathsf{Partial}} \in \prod_{v \in V} Q_v$

Output: an assignment $\sigma \in \sigma_{\mathsf{True}}^{\mathcal{C}}$

1 $\Phi_v = \left(V_v, \left(\pi_v^{-1}(\sigma_{\mathsf{Partial}}(v))\right)_{v \in V_v}, \mathcal{C}_v\right) \leftarrow \texttt{Component}(\Phi, \boldsymbol{\pi}, \sigma_{\mathsf{Partial}}, v)$ for all $v \in V$
`/* Ignore the returned Token since it is always True here. */`

2 Initialize $\widetilde{V} \leftarrow \emptyset$

3 **while** $\exists v \in V \setminus \widetilde{V}$ **do**

4 $\quad \sigma_v \leftarrow \texttt{RejectionSampling}(\Phi_v, r_v)$ `/*` r_v `is a fresh new randomness tape. */`

5 $\quad$ Assign $\sigma(u) \leftarrow \sigma_v(u)$ for all $u \in V_v$

6 $\quad$ Update $\widetilde{V} \leftarrow \widetilde{V} \cup V_v$

7 **end**

8 **return** σ

We observe the following results regarding Algorithm 8.

Lemma 3.31. *If $e \cdot \alpha \cdot \Delta \le 1$, then the following holds for* $\texttt{FinalSampling}(\Phi, \boldsymbol{\pi}, \sigma_{\mathsf{Partial}})$.

- *It halts almost surely, and outputs $\sigma \sim \left(\sigma_{\mathsf{True}}^{\mathcal{C}} \cap \left(\prod_{v \in V} \pi_v^{-1}(\sigma_{\mathsf{Partial}}(v))\right)\right)$ when it halts.*
- *Its expected total running time is at most $O\left(k|\mathcal{C}| + d|V| + k \sum_{v \in V} |\mathcal{C}_v| \left(1 - e \cdot \alpha\right)^{-|\mathcal{C}_v|}\right)$.*

Proof. We construct each Φ_v when needed by Line 4. This takes time $O\left(k|\mathcal{C}| + d|V|\right)$ by Lemma 3.3. By Lemma 3.4, each iteration of Line 4 halts almost surely and generates $\sigma_v \sim \sigma_{\mathsf{True}}^{\mathcal{C}_v}$ in expected time $O\left((k\left|\mathcal{C}_v\right| + 1) \cdot (1 - e \cdot \alpha)^{-|\mathcal{C}_v|}\right)$. Therefore FinalSampling halts almost surely and its expected total running time is

$$
O\left(k|\mathcal{C}| + d|V| + \sum_{v \text{ needed by Line 4}} (k\left|\mathcal{C}_v\right| + 1) \cdot (1 - e \cdot \alpha)^{-|\mathcal{C}_v|}\right)
$$

$$\le O\left(k|\mathcal{C}|+d|V|+k\sum_{v\in V}|\mathcal{C}_v|\left(1-e\cdot\alpha\right)^{-|\mathcal{C}_v|}\right). \qquad \text{(pulling out the case } |\mathcal{C}_v|=0\text{)}$$

Moreover when $\mathsf{FinalSampling}$ halts, by iteratively applying Item (4) of Lemma 3.3 we have

$$\sigma\sim\prod_{v\text{ needed by }\mathsf{Line\ 4}}\sigma_{\mathsf{True}}^{\mathcal{C}_v}=\sigma_{\mathsf{True}}^{\mathcal{C}}\cap\left(\prod_{v\in V}\pi_v^{-1}(\sigma_{\mathsf{Partial}}(v))\right). \qquad \square$$

Combined with Proposition 3.19, we analyze the performance of Algorithm 8 in Algorithm 2.

Proposition 3.32. *If $e\cdot\alpha\cdot\Delta\le 1$, $d^2k^4\lambda<1$, and $e\cdot\Delta^2\rho\le 1/32$, then the following holds for the $\mathsf{FinalSampling}$ in Algorithm 2.*

- *It halts almost surely, and outputs $\sigma\sim\sigma_{\mathsf{True}}^{\mathcal{C}}$ when it halts.*
- *Its expected running time is at most $O\left(dk\Delta^2|V|\right)$.*

Proof. By Lemma 3.31, it halts almost surely. Fix an arbitrary $\sigma^*\in\sigma_{\mathsf{True}}^{\mathcal{C}}$. Let $\sigma_{\mathsf{Partial}}=\sigma^*|_{\boldsymbol{\pi}}$. Combining Proposition 3.19 and Definition 3.18, we have

$$\mathbf{Pr}\left[\sigma=\sigma^*\right]=\mu^{\Phi|_{\boldsymbol{\pi}}}(\sigma_{\mathsf{Partial}})\cdot\frac{1}{\left|\sigma_{\mathsf{True}}^{\mathcal{C}}\cap\left(\prod_{v\in V}\pi_v^{-1}(\sigma_{\mathsf{Partial}}(v))\right)\right|}=\frac{1}{\left|\sigma_{\mathsf{True}}^{\mathcal{C}}\right|}$$

as desired.

Note that $\mu^{\Phi|_{\boldsymbol{\pi}}}$ is a stationary distribution for P_i by Fact 3.25. Meanwhile by Lemma 3.29, the t-th for iteration in $\mathsf{BoundingChain}$ is a coupling of P_{i_t}. Thus in $\mathsf{FinalSampling}$, upon receiving $\sigma_{\mathsf{Partial}}$ which has distribution $\mu^{\Phi|_{\boldsymbol{\pi}}}$, we can execute $|V|$ more rounds of $\mathsf{Line\ 3\text{-}11}$ in Algorithm 4 on $\sigma_{\mathsf{Partial}}$ using fresh randomness, and the resulted partial assignment still has distribution $\mu^{\Phi|_{\boldsymbol{\pi}}}$. In other words, we may safely assume the last $|V|$ rounds of update in the final $\mathsf{BoundingChain}$ procedure are all using $\mathsf{RejectionSampling}$. Thus each $|\mathcal{C}_v|$ in Lemma 3.31 also satisfies the concentration bound in Lemma 3.12. By a similar calculation in the proof of the efficiency part of Proposition 3.5 and noticing $|\mathcal{C}|\le d|V|$, the expected running time here is at most

$$O\left(k\left|\mathcal{C}\right|+d|V|+dk|V|\sum_{\ell=1}^{+\infty}\left(\frac{1}{32}\right)^{\ell-1}\cdot\Delta\cdot(\ell+1)\Delta\cdot 4^{\ell+1}\right)=O\left(dk\Delta^2|V|\right). \qquad \square$$

3.5 Putting Everything Together

Now we put everything together to prove our main theorem.

Proof of Theorem 3.2. The correctness part follows immediately from Proposition 3.32 and Proposition 3.19. Thus we focus on the efficiency part.

- Let X be the total running time of $\texttt{AtomicCSPSampling}(\Phi,\boldsymbol{\pi})$.
- Let A be the time for computing constants in Notation 3.1. Then $A=O(|V|+|\mathcal{C}|)=O(d|V|)$.
- For integer $i\ge 1$ and $j\in[i]$, let $X_{i,j}$ be the running time of the $(-j)$-th for iteration in $\texttt{BoundingChain}(\Phi,\boldsymbol{\pi},-i,r_{-i},\dots,r_{-1})$. Then by Proposition 3.5

$$\mathbb{E}\left[X_{i,j}^2\right]=O\left(k^2d^3\Delta^3\right).$$

- Let T_{Final} be the T when the while iterations stop. Then by Proposition 3.5,

$$\mathbf{Pr}\left[T_{\mathsf{Final}} \geq t\right] \leq dk|V| \cdot \left(d^2k^4\lambda\right)^{\frac{t}{2(|V|-1)}-1} \leq dk|V| \cdot 2^{-\frac{t}{4(|V|-1)}} \quad \text{for } t \geq 4(|V|-1).$$

- Let Y be the running time of the FinalSampling in the end. Then by Proposition 3.32

$$\mathbb{E}[Y] = O\left(dk\Delta^2|V|\right).$$

Therefore we have $X = A + \sum_{t=0}^{\log(T_{\mathsf{Final}})} \sum_{j=1}^{2^t} X_{2^t,j} + Y$[14] and

$$\begin{aligned}
\mathbb{E}[X] &= \mathbb{E}[A] + \mathbb{E}[Y] + \sum_{t=0}^{+\infty}\sum_{j=1}^{2^t} \mathbb{E}\left[X_{2^t,j} \cdot [T_{\mathsf{Final}} \geq 2^t]\right] \\
&\leq \mathbb{E}[A] + \mathbb{E}[Y] + \sum_{t=0}^{+\infty}\sum_{j=1}^{2^t} \sqrt{\mathbb{E}\left[X_{2^t,j}^2\right]\mathbf{Pr}\left[T_{\mathsf{Final}} \geq 2^t\right]} \qquad \text{(by Cauchy-Schwarz inequality)} \\
&\leq \mathbb{E}[A] + \mathbb{E}[Y] + \sum_{t=0}^{m}\sum_{j=1}^{2^t} \sqrt{\mathbb{E}\left[X_{2^t,j}^2\right]} + \sum_{t=m+1}^{+\infty}\sum_{j=1}^{2^t} \sqrt{\mathbb{E}\left[X_{2^t,j}^2\right]\mathbf{Pr}\left[T_{\mathsf{Final}} \geq 2^t\right]} \\
&\qquad\qquad (m \geq \lfloor \log(4(|V|-1)) \rfloor \text{ to be determined later}) \\
&= O\left(dk\Delta^2|V| + \sqrt{k^2d^3\Delta^3} \cdot \left(2^m + \sqrt{dk|V|}\sum_{t=m+1}^{+\infty} 2^t \cdot 2^{-\frac{2^t}{8(|V|-1)}}\right)\right).
\end{aligned}$$

We pick $m = \lceil \log(|V|) + \log\log(|V|) + 10 \rceil$ then

$$\begin{aligned}
&2^m + \sqrt{dk|V|}\sum_{t=m+1}^{+\infty} 2^{t-\frac{2^t}{8(|V|-1)}} \\
&\leq 2^m + \sqrt{dk|V|}\int_m^{+\infty} 2^{x-\frac{2^x}{8(|V|-1)}}\,\mathrm{d}\,x \qquad \left(2^{x-\frac{2^x}{n}} \text{ is decreasing when } 2^x \geq \tfrac{n}{\ln(2)}\right) \\
&= 2^m + \sqrt{dk|V|} \cdot \frac{8(|V|-1)}{\ln^2(2)} \cdot 2^{-\frac{2^m}{8(|V|-1)}} \qquad \left(\text{since } \left(\tfrac{-n}{\ln^2(2)} \cdot 2^{-\frac{2^x}{n}}\right)' = 2^{x-\frac{2^x}{n}}\right) \\
&= O\left(|V|\log(|V|) + \sqrt{dk}\right).
\end{aligned}$$

Since $d \leq \Delta$ and $|V| \geq k$, we have $\mathbb{E}[X] = O\left(kd^2\Delta^2|V|\log(|V|)\right)$. □

Remark 3.33. Computing higher moments of $X_{i,j}, Y$ and using possibly stronger assumption and choosing better m, one can improve the dependency on k, d, Δ in the expected running time. However we view these as constants compared with $|V|$. Thus we do not make the effort here.

4 Applications

Here we instantiate Theorem 3.2 to special CSPs. We first review the constants from Notation 3.1.

Let $\Phi = \left(V, (\Omega_v)_{v\in V}, \mathcal{C}\right)$ be an atomic CSP and $\boldsymbol{\pi} = (\pi_v)_{v\in V}$ be projections where $\pi_v \colon \Omega_v \to Q_v$. Let $\alpha = \max_{C\in\mathcal{C}} \frac{1}{\prod_{v\in\mathsf{vbl}(C)}\left|\pi_v^{-1}\left(\sigma_{\mathsf{False}}^{C|\pi}(v)\right)\right|}$. When $e \cdot \alpha < 1$, define

[14] Technically we also need to initialize the randomness in Line 1 of Algorithm 2, and check for Line 5 in Algorithm 2, and initialize the assignment in Line 1 of Algorithm 4. However these can be done on the fly and their cost will be minor compared with the parts we listed.

- $\beta = (1 - e \cdot \alpha)^{-d(\Phi)}$,
- $\rho = \max_{C \in \mathcal{C}} \prod_{v \in \mathsf{vbl}(C)} \min \left\{ 1, \beta \cdot \frac{\left|\pi_v^{-1}\left(\sigma_{\mathsf{False}}^{C|\boldsymbol{\pi}}(v)\right)\right|}{|\Omega_v|} \right\}$,
- $\gamma_v = (\beta - 1) \cdot \max_{q \in Q_v} \left(1 - \frac{\left|\pi_v^{-1}(q)\right|}{|\Omega_v|} \right)$,
- $\lambda = \max_{C \in \mathcal{C}} \prod_{v \in \mathsf{vbl}(C)} \left(\frac{\left|\pi_v^{-1}(\sigma_{\mathsf{False}}^{C|\boldsymbol{\pi}}(v))\right|}{|\Omega_v|} + (|Q_v| - 1) \cdot \gamma_v \right)$.

Theorem (Theorem 3.2 restated)**.** *Let $\Phi = \left(V, (\Omega_v)_{v \in V}, \mathcal{C}\right)$ be an atomic CSP. Let $\boldsymbol{\pi} = (\pi_v)_{v \in V}$ be projections. If $e \cdot \alpha \cdot \Delta \le 1$, $d^2 k^4 \lambda \le 1/2$, and $e \cdot \Delta^2 \rho \le 1/32$, then the following holds for* $\texttt{AtomicCSPSampling}(\Phi, \boldsymbol{\pi})$.

- *Correctness. It halts almost surely and outputs $\sigma \sim \sigma_{\mathsf{True}}^{\mathcal{C}}$ when it halts.*
- *Efficiency. Its expected total running time is $O\left(kd^2\Delta^2|V|\log(|V|)\right)$.*

4.1 q-coloring of k-uniform Hypergraphs

Definition 4.1 (q-coloring of k-uniform Hypergraphs)**.** Let q and k be positive integers. Let $H = (V, E)$ be a k-uniform hypergraph.[15] We associate it with an atomic CSP $\Phi = \Phi(H, q, k) = \left(V, [q]^V, \mathcal{C}\right)$ where $\mathcal{C} = \left\{ C_{e,i} \colon [q]^V \to \{\mathsf{True}, \mathsf{False}\} \,\middle|\, e \in E, i \in [q] \right\}$ and

$$C_{e,i}(\sigma) = \mathsf{False} \quad \text{iff} \quad \sigma(v) = i, \forall v \in e.$$

$\sigma_{\mathsf{True}}^{\mathcal{C}}$ is the set of proper colorings.

Therefore $k(\Phi) = k$, $d(\Phi) = q \cdot d(H)$, and $\Delta(\Phi) \le q \cdot \Delta(H) \le qk \cdot d(H)$. Meanwhile $p(\Phi) = q^{-k}$. Thus by Theorem 2.3, there exists a proper coloring if $e \cdot q^{1-k} \cdot \Delta(H) \le 1$. Here we obtain the following perfect sampling result.

Theorem 4.2. *Let $H = (V, E)$ be a k-uniform hypergraph. If $d = d(H) \le q^{(1/3 - o(1))k}$ for positive integers k and q large enough, then there exists projections $\boldsymbol{\pi} = (\pi_v)_{v \in V}$ such that $\Phi = \Phi(H, q, k)$ and $\boldsymbol{\pi}$ satisfy the conditions in Theorem 3.2 and $\boldsymbol{\pi}$ can be constructed in time $O(q|V|)$.*

Proof. We will show $d \le q^{(k-8)/3} \cdot 2^{-30} 8^{-k} = q^{(1/3 - o(1))k}$ suffices.

Let $R = \lfloor q^{2/3} \rfloor$. For each $v \in V$ let $|Q_v| = [R]$ and π_v projects $[q]$ to $[R]$ smoothly, i.e., for each $c \in Q_v$, $\pi_v^{-1}(c)$ equals $\lfloor q/R \rfloor$ or $\lceil q/R \rceil$. This can be done in time $O(q|V|)$. Now we compute the constants for $\Phi(H, q, k)$.

Since q is large enough, we have

$$\alpha \le \lfloor q/R \rfloor^{-k} \le \left(q^{1/3}/2\right)^{-k}.$$

Since $(1-x)^{-1/x} \le 4$ for $x \in [0, 1/2]$ and $4^t \le 1 + 4t$ for $t \in [0, 1/2]$, we have

$$\beta = (1 - e \cdot \alpha)^{-q \cdot d} \le 4^{eq \cdot \alpha d} \le 1 + 4eq \cdot \alpha d \le 1 + 2^{-20} q^{-5/3} 4^{-k}.$$

Thus

$$\gamma_v \le 4eq \cdot \alpha d \le 2^{-20} q^{-5/3} 4^{-k},$$

[15] A hypergraph is *k-uniform* if each hyperedge has exactly k elements.

$$\lambda \le \left(\frac{\lceil q/R\rceil}{q} + 4eq^2\cdot \alpha d\right)^k \le \left(2\cdot q^{-2/3} + 2^{-20}q^{-2/3}4^{-k}\right)^k \le 3^k q^{-2k/3},$$
$$\rho \le \left((1+4eq\cdot\alpha d)\cdot \frac{\lceil q/R\rceil}{q}\right)^k \le \left(\left(1+2^{-20}q^{-5/3}4^{-k}\right)\cdot 2\cdot q^{-2/3}\right)^k \le 3^k q^{-2k/3}.$$

Now we verify the conditions in Theorem 3.2. The first condition is

$$e\cdot\alpha\cdot\Delta(\Phi) \le eqk\cdot d\cdot\alpha \le eqk\cdot q^{(k-8)/3}2^{-30}8^{-k}\cdot q^{-k/3}2^k \le 1.$$

The second condition is

$$k^4 d(\Phi)^2\lambda \le q^2k^4\cdot d^2\cdot\lambda \le q^2k^4\cdot q^{2(k-8)/3}2^{-60}8^{-2k}\cdot 3^k q^{-2k/3} \le \frac{1}{2}.$$

The third condition is

$$e\Delta(\Phi)^2\rho \le eq^2k^2\cdot d^2\cdot\rho \le eq^2k^2\cdot q^{2(k-8)/3}2^{-60}8^{-2k}\cdot 3^k q^{-2k/3} \le \frac{1}{32}.$$ □

4.2 Boolean k-CNF Formulas

Definition 4.3 (Boolean k-CNF Formulas)**.** Let k be a positive integer. A CSP $\Phi = \Phi(k) = \left(V, \{\mathsf{True}, \mathsf{False}\}^V, \mathcal{C}\right)$ is a Boolean k-CNF formula if Φ is atomic and each $C\in\mathcal{C}$ has $|\mathsf{vbl}(C)| = k$.

Therefore $k(\Phi) = k$, $\Delta(\Phi) \le k\cdot d(\Phi)$, and $p(\Phi) = 2^{-k}$. Thus by Theorem 2.3, there exists a satisfying assignment if $ek2^{-k}\cdot d(\Phi)\le 1$. Here we obtain the following perfect sampling result.

Theorem 4.4. *Let $\Phi = \Phi(k) = \left(V, \{\mathsf{True}, \mathsf{False}\}^V, \mathcal{C}\right)$ be a Boolean k-CNF formula. If $d = d(\Phi) \le 2^{0.175k}$ for positive integer k large enough, then there exists projections $\boldsymbol{\pi} = (\pi_v)_{v\in V}$ such that Φ and $\boldsymbol{\pi}$ satisfy the conditions in Theorem 3.2 and $\boldsymbol{\pi}$ can be constructed in time $O(dk^2|V|)$ with success probability at least* 0.99*.*

Proof. Let $k_1, k_2 \in [k]$ be two parameters to be determined later. We will find $V_1 \subseteq V$ to make sure $|\mathsf{vbl}(C)\cap V_1| \ge k_1$ and $|\mathsf{vbl}(C)\setminus V_1| \ge k_2$ hold for all $C\in\mathcal{C}$. Then $\boldsymbol{\pi}$ is constructed by setting $\pi_v\colon \{\mathsf{True}, \mathsf{False}\} \to \{0,1\}$ with $\pi_v(\mathsf{True}) = 0$ and $\pi_v(\mathsf{False}) = 1$ for $v\in V_1$; and $\pi_v\colon \{\mathsf{True}, \mathsf{False}\} \to \{0\}$ with $\pi_v(\mathsf{True}) = \pi_v(\mathsf{False}) = 0$ for $v\notin V_1$.

We now show $d \le 2^{-30}k^{-2}\cdot \min\left\{2^{k_1/2}, 2^{k_2}\right\}$ suffices given such $\boldsymbol{\pi}$. Firstly

$$\alpha = \max_{C\in\mathcal{C}} 2^{-|\mathsf{vbl}(C)\setminus V_1|} \le 2^{-k_2}.$$

Since $(1-x)^{-1/x} \le 4$ for $x\in[0,1/2]$ and $4^t \le 1+4t$ for $t\in[0,1/2]$, we also have

$$\beta = (1-e\cdot\alpha)^{-d} \le 4^{e\cdot\alpha d} \le 1 + 4e\cdot\alpha d \le 1 + 2^{-20}k^{-2}.$$

Thus

$$\gamma_v = 0 \quad \text{if } v\notin V_1 \quad \text{and} \quad \gamma_v = \frac{\beta-1}{2} \le 2^{-20}k^{-2} \quad \text{if } v\in V_1,$$
$$\lambda \le \left(\frac{1}{2} + 2^{-20}k^{-2}\right)^{k_1} \le 2^{k_1}\left(1+2^{-19}k^{-2}\right)^k \le 2\cdot 2^{k_1},$$
$$\rho \le \left(\frac{\beta}{2}\right)^{k_1} \le 2^{k_1}\left(1+2^{-20}k^{-2}\right)^k \le 2\cdot 2^{k_1}.$$

We verify the conditions in Theorem 3.2. The first condition is

$$e \cdot \alpha \cdot \Delta(\Phi) \le ek \cdot d \cdot \alpha \le ek \cdot 2^{-30} k^{-2} 2^{k_2} \cdot 2^{-k_2} \le 1.$$

The second condition is

$$k^4 d(\Phi)^2 \lambda = k^4 \cdot d^2 \cdot \lambda \le k^4 \cdot 2^{-60} k^{-4} 2^{k_1} \cdot 2 \cdot 2^{k_1} \le \frac{1}{2}.$$

The third condition is

$$e\Delta(\Phi)^2 \rho \le ek^2 \cdot d^2 \cdot \rho \le ek^2 \cdot 2^{-60} k^{-4} 2^{k_1} \cdot 2 \cdot 2^{k_1} \le \frac{1}{32}.$$

Now our goal is to construct V_1 with desired properties for k_1, k_2 as large as possible. We first note the following well-known estimate for binomial coefficients.

Fact 4.5. *For any positive integer n and $i \in \{0, 1, \dots, n\}$, we have*

$$\binom{n}{i} \le 2^{n \cdot H(i/n)}, \quad \text{where } H(x) = x \log\left(\frac{1}{x}\right) + (1-x)\log\left(\frac{1}{1-x}\right).$$

Let $\eta \in (0,1)$ be a constant to be optimized later. We put each $v \in V$ into V_1 with probability η independently. For each $C \in \mathcal{C}$, let $\mathcal{E}_C$ be the event "$|\mathsf{vbl}(C) \cap V_1| < k_1$ or $|\mathsf{vbl}(C) \setminus V_1| < k_2$". Thus

$$\begin{aligned}
\mathbf{Pr}[\mathcal{E}_C] &= \sum_{(0 \le i < k_1) \vee (k - k_2 < i \le k)} \binom{k}{i} \eta^i (1-\eta)^{k-i} \\
&\le k \cdot \max_{(0 \le i \le k_1) \vee (k-k_2 \le i \le k)} \binom{k}{i} \eta^i (1-\eta)^{k-i} \\
&\le k \cdot \max_{(0 \le x \le k_1/k) \vee (1 - k_2/k \le x \le 1)} 2^{k \cdot \left(x \log\left(\frac{\eta}{x}\right) + (1-x)\log\left(\frac{1-\eta}{1-x}\right)\right)} && \text{(by Fact 4.5)} \\
&= k \cdot 2^{-k \cdot s}, && (s = \min_{(0 \le x \le k_1/k) \vee (1-k_2/k \le x \le 1)} \mathsf{KL}(x \| \eta))
\end{aligned}$$

where $\mathsf{KL}(a\|b) = a \log\left(\frac{a}{b}\right) + (1-a)\log\left(\frac{1-a}{1-b}\right)$ is the *Kullback-Leibler divergence.*

Note that each $\mathcal{E}_C$ is correlated with at most $\Delta(\Phi) \le dk$ many $\mathcal{E}_{C'}$ (including $C' = C$). Thus by Theorem 2.5, V_1 can be constructed with probability at least 0.99 in time $O(dk^2|V|)$, provided $e \cdot dk \cdot k 2^{-k \cdot s} \le 1$.

Now we set $\eta = 0.595$, $k_1 = \lfloor (0.35 + 10^{-5})k \rfloor$, and $k_2 = \lceil (0.175 + 10^{-5})k \rceil$. We also note the following well-known convexity result regarding Kullback-Leibler divergence.

Fact 4.6. *For any fixed $b \in [0,1]$, $\mathsf{KL}(a\|b)$ is a convex function in $a \in [0,1]$ with minimum $\mathsf{KL}(b\|b) = 0$. Thus $\mathsf{KL}(a\|b)$ is decreasing when $a \le b$, and increasing when $a \ge b$.*

Assuming k is large enough, we have

$$d \le 2^{0.175k} \le 2^{-30} k^{-2} \cdot \min\left\{2^{k_1/2}, 2^{k_2}\right\}$$

and

$$\begin{aligned}
s &\ge \min_{(0 \le x \le 0.35 + 10^{-5}) \vee (0.825 - 10^{-5} \le x \le 1)} \mathsf{KL}(x \| 0.595) \\
&\ge \min\left\{\mathsf{KL}(0.35 + 10^{-5} \| 0.595), \mathsf{KL}(0.825 - 10^{-5} \| 0.595)\right\} && \text{(by Fact 4.6)} \\
&\ge 0.1756.
\end{aligned}$$

Thus $e \cdot dk \cdot k2^{-k \cdot s} \le e \cdot k^2 \cdot 2^{-0.0006k} \le 1$ as desired when k is large enough. $\square$

We remark that the projections $\boldsymbol{\pi}$ constructed in Theorem 4.4 is randomized and satisfy the conditions in Theorem 3.2 with certain constant probability. On the other hand, given the projections $\boldsymbol{\pi}$ and CSP Φ, it is easy to check whether the conditions are violated. Thus we can keep finding random $\boldsymbol{\pi}$ until it works, and the expected running time is still $O(dk^2|V|)$.

4.3 General Atomic Constraint Satisfaction Problems

For the general atomic CSPs, our current algorithm does not give good bounds. This is because in Notation 3.1, the definition of $\lambda(\Phi|_{\boldsymbol{\pi}})$ has a bad dependency on $|Q_v|$. This can be improved by slightly tweaking the algorithm and using a more refined analysis, which will be included in our full version. Here we briefly sketch the idea.

First recall this $|Q_v|$ comes from $\mathsf{SafeSampling}$ where we subtract γ_v from each $\left|\pi_v^{-1}(q)\right| / |\Omega_v|$ for $q \in Q_v$. This is to make sure the distribution is oblivious and "mimics" the true distribution — the marginal distribution on v from the uniform satisfying assignments when the partial assignment on other variables is fixed to a certain value. More precisely, this distribution should couple well with the true distribution as we described in Algorithm 7.

To shave the $|Q_v|$ factor, we use a dyadic decomposition for Q_v. We view Q_v as $\{q_0, \ldots, q_{s-1}\}$ where $s = |Q_v|$. For simplicity, assume $s = 2^\ell$ is an integer power of 2. Then we use $(b_1, \ldots, b_\ell) \in \{0,1\}^\ell$ to represent each value of Q_v. For example, $b_1 = 1$ indicates $(b_1, \ldots, b_\ell) \in \left\{q_{s/2}, q_{s-1}\right\}$; and $b_1 = 0, b_2 = 1$ indicates $(b_1, \ldots, b_\ell) \in \left\{q_{s/4}, q_{s/2-1}\right\}$. Then we enrich $\star$ to $2^\ell - 1$ cases

$$b_1 \cdots b_{\ell'} \circ \star \quad \text{for all possible } \ell \in \{0, 1, \ldots, \ell - 1\} \text{ and } (b_1, \ldots, b_{\ell'}) \in \{0,1\}^{\ell'},$$

where each $b_1 \cdots b_{\ell'} \circ \star$ means "we are certain about the partial assignment on v until the $(\ell'+1)$-th bit". Then, instead of doing $\mathsf{SafeSampling}$ directly on s values, we try to nail down each bit of the value from b_1 to b_ℓ in the this new $\mathsf{DyadicSafeSampling}$. The benefit of this dyadic decomposition is, for each bit, we only need to subtract γ_v from $1/2$ for each bit in $\{0, 1\}$. When s is not a perfect power of 2, the dyadic decomposition is similar: each bit now splits s' values into two parts, where one has $\lfloor s'/2 \rfloor$ and another $\lceil s'/2 \rceil$.

The analysis in Subsection 3.2.1 has almost no change since we only do $\mathsf{RejectionSampling}$ when there is no $\star$ (other than the variable we choose to update) in the corresponding component. For the information percolation argument in Subsection 3.2.2, we will now remove the dependency on $|Q_v|$ from the probability of $\mathcal{E}_{t,C_i}$ when doing $\mathsf{DyadicSafeSampling}$. The $\mathsf{SystematicScan}$ algorithm from Subsection 3.3 does *not* change, but the coupling designed in Algorithm 7 needs to be refined in the same dyadic manner to fit for the new $\mathsf{DyadicSafeSampling}$.

Then the new parameters we need will be

- $\alpha = \max_{C \in \mathcal{C}} \frac{1}{\prod_{v \in \mathsf{vbl}(C)} \left|\pi_v^{-1}\left(\sigma_{\mathsf{False}}^{C|\pi}(v)\right)\right|}$,
- $\beta = (1 - e \cdot \alpha)^{-d(\Phi)}$,
- $\rho = \max_{C \in \mathcal{C}} \prod_{v \in \mathsf{vbl}(C)} \min\left\{1, \beta \cdot \frac{\left|\pi_v^{-1}\left(\sigma_{\mathsf{False}}^{C|\boldsymbol{\pi}}(v)\right)\right|}{|\Omega_v|}\right\}$,
- $\gamma_v \equiv \beta - 1$,
- $\lambda = \max_{C \in \mathcal{C}} \prod_{v \in \mathsf{vbl}(C)} \left(\frac{\left|\pi_v^{-1}(\sigma_{\mathsf{False}}^{C|\boldsymbol{\pi}}(v))\right|}{|\Omega_v|} + 2\gamma_v \cdot \left(1 + 2 \cdot \max_{q,q' \in Q_v} \frac{\left|\pi_v^{-1}(q)\right|}{\left|\pi_v^{-1}(q')\right|}\right)\right)$.

The conditions in Theorem 3.2 will be

$$e \cdot \alpha \cdot \Delta \le 1, \quad d^2 k^4 \cdot k\log(\Delta) \cdot \lambda \le 1/2,\text{[16]} \quad \text{and} \quad e \cdot \Delta^2 \rho \le 1/32.$$

This, however, still fails to provide any useful bound since k can be made arbitrarily large by adding extremely large constraints. To improve the dependence on k for the conditions, we need to be more careful with the information percolation argument.

Recall in our current analysis for the coalescence part of Proposition 3.5, we first enumerate all possible paths of length ℓ in the extended hypergraph. For each path, we *obliviously* take out the extended constraints indexed by odd numbers so that these are independent and each happens with probability at most λ. Then the proof completes by a union bound.

Our improvements have two parts. First we refine the enumeration. Assume we are at extended constraint (e, C). For the next step (e', C'), we first choose some $t \in e$ with $|e| = |\mathsf{vbl}(C)|$ choices; then choose the constraint C' containing x_{i_t} with at most $d(\Phi)$ choices; then use Fact 3.14 to find e' with $|e'| = |\mathsf{vbl}(C')|$ choices. After ℓ steps, we obtain a path $(e_1, C_1), \ldots, (e_\ell, C_\ell)$. Then our second improvement comes into place. For this specific path and assuming ℓ is even, we define its weight to match the selection scheme above as

$$\prod_{i=1}^{\ell-1} \left(|e_i| \cdot d(\Phi) \cdot |e_{i+1}|\right) \le d(\Phi)^{\ell-1} \prod_{i=1}^{\ell} |e_i|^2.$$

Meanwhile to upper bound the probability that this path happens, we choose extended constraints *adaptively*: we select either (A) the extended constraints indexed by odd numbers, or (B) the extended constraints indexed by even numbers, so as to maximize the product of $|e|$'s. More formally, we choose (A) iff

$$\prod_{j=1}^{j \le \ell/2} |e_{2j-1}| \ge \prod_{j=1}^{j \le \ell/2} |e_{2j}|.$$

We replace the previous $d(\Phi)^2 k(\Phi)^4 \lambda \le 1/2$ by the following fine-grained condition for each $C \in \mathcal{C}$

$$d^2 \left|\mathsf{vbl}(C)\right|^4 \cdot \underbrace{\left|\mathsf{vbl}(C)\right| \log(\Delta) \prod_{v \in \mathsf{vbl}(C)} \left(\frac{\left|\pi_v^{-1}(\sigma_{\mathsf{False}}^{C|_\pi}(v)\right|}{|\Omega_v|} + 2\gamma_v \cdot \left(1 + 2 \cdot \max_{q, q' \in Q_v} \frac{\left|\pi_v^{-1}(q)\right|}{\left|\pi_v^{-1}(q')\right|}\right)\right)}_{\lambda_C} \le \frac{1}{2}.$$

Then, upon choosing (A), we upper bound the probability by

$$\prod_{j=1}^{j \le \ell/2} \lambda_{C_{2j-1}} \le 2^{-\ell/2} \cdot d(\Phi)^{-\ell} \cdot \prod_{j=1}^{j \le \ell/2} |e_{2j-1}|^{-4} \le 2^{-\ell/2} \cdot d(\Phi)^{-\ell} \cdot \prod_{i=1}^{\ell} |e_i|^{-2},$$

which cancels out the weight of the path. A similar analysis works for the case we choose (B).

In all, the new parameters we need will be

- $\alpha = \max_{C \in \mathcal{C}} \frac{1}{\prod_{v \in \mathsf{vbl}(C)} \left|\pi_v^{-1}\left(\sigma_{\mathsf{False}}^{C|_\pi}(v)\right)\right|}$,
- $\beta = (1 - e \cdot \alpha)^{-d(\Phi)}$,

[16]The additional $k\log(\Delta)$ comes from the case where (at most) one variable in the extended constraint can be set to $b_1 \cdots b_{\ell'} \circ \star$ without falsifying the constraint. This delicate case will be elaborated in our full version.

- $\rho = \max_{C\in\mathcal{C}} \prod_{v\in\mathsf{vbl}(C)} \min\left\{1, \beta \cdot \frac{\left|\pi_v^{-1}\left(\sigma_{\mathsf{False}}^{C|\boldsymbol{\pi}}(v)\right)\right|}{|\Omega_v|}\right\}$,
- $\gamma_v \equiv \beta - 1$,
- $\lambda_C = |\mathsf{vbl}(C)| \log(\Delta) \cdot \prod_{v\in\mathsf{vbl}(C)} \left(\frac{\left|\pi_v^{-1}(\sigma_{\mathsf{False}}^{C|\boldsymbol{\pi}}(v))\right|}{|\Omega_v|} + 2\gamma_v \cdot \left(1 + 2\cdot \max_{q,q'\in Q_v} \frac{\left|\pi_v^{-1}(q)\right|}{\left|\pi_v^{-1}(q')\right|}\right)\right)$ for each $C \in \mathcal{C}$.

The conditions in Theorem 3.2 becomes

$$e\cdot\alpha\cdot\Delta \le 1, \quad e\cdot\Delta^2\rho \le 1/32, \quad \text{and } d^2\,|\mathsf{vbl}(C)|^4\,\lambda_C \le 1/2 \text{ for all } C\in\mathcal{C}.$$

Now with some similar calculation as [JPV21, Item (5) of Proposition 3.3], we are able to construct projections efficiently to satisfy these conditions provided $p(\Phi)\cdot\Delta(\Phi)^{O(1)} \le 1$.

Acknowledgement

KH wants to thank Weiming Feng for helpful discussion. KW wants to thank Chao Liao, Pinyan Lu, Jiaheng Wang, Kuan Yang, Yitong Yin, Chihao Zhang for helpful discussion on related topics in summer 2020. KW also wants to thank Kuan Yang for the help with *Mathematica*.

References

[ACG12] Ittai Abraham, Shiri Chechik, and Cyril Gavoille. Fully dynamic approximate distance oracles for planar graphs via forbidden-set distance labels. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 1199–1218. ACM, 2012. 3

[Alo91] Noga Alon. A parallel algorithmic version of the local lemma. *Random Struct. Algorithms*, 2(4):367–378, 1991. (Conference version in *FOCS*'91). doi:10.1002/rsa.3240020403. 2

[BC20] Siddharth Bhandari and Sayantan Chakraborty. Improved bounds for perfect sampling of k-colorings in graphs. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 631–642. ACM, 2020. doi:10.1145/3357713.3384244. 3

[Bec91] József Beck. An algorithmic approach to the Lovász local lemma. *Random Struct. Algorithms*, 2(4):343–365, 1991. doi:10.1002/rsa.3240020402. 2

[BGG+19] Ivona Bezáková, Andreas Galanis, Leslie A. Goldberg, Heng Guo, and Daniel Štefankovič. Approximation via correlation decay when strong spatial mixing fails. *SIAM J. Comput.*, 48(2):279–349, 2019. doi:10.1137/16M1083906. 2

[CS00] Artur Czumaj and Christian Scheideler. Coloring nonuniform hypergraphs: a new algorithmic approach to the general Lovász local lemma. *Random Struct. Algorithms*, 17(3-4):213–237, 2000. doi:10.1002/1098-2418(200010/12)17:3/4<213::AID-RSA3>3.0.CO;2-Y. 2

[EL75] Paul Erdös and László Lovász. Problems and results on 3-chromatic hypergraphs and some related questions. *Infinite and finite sets*, 11:609–627, 1975. 2, 9

[FGYZ20] Weiming Feng, Heng Guo, Yitong Yin, and Chihao Zhang. Fast sampling and counting k-SAT solutions in the local lemma regime. In *STOC*, pages 854–867. ACM, 2020. doi:10.1145/3357713.3384255. 2, 3, 4, 5, 7, 15

[FHY20] Weiming Feng, Kun He, and Yitong Yin. Sampling constraint satisfaction solutions in the local lemma regime. *arXiv preprint arXiv:2011.03915*, 2020. 2, 3, 4, 5, 7

[Fil97] James Allen Fill. An interruptible algorithm for perfect sampling via Markov chains. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing (STOC)*, pages 688–695, 1997. 3

[FMMR00] James Allen Fill, Motoya Machida, Duncan J Murdoch, and Jeffrey S Rosenthal. Extension of Fill's perfect rejection sampling algorithm to general chains. *Random Structures & Algorithms*, 17(3-4):290–316, 2000. 3

[FVY19] Weiming Feng, Nisheeth K. Vishnoi, and Yitong Yin. Dynamic sampling from graphical models. In *STOC*, pages 1070–1081. ACM, 2019. 3

[GGGY20] Andreas Galanis, Leslie Ann Goldberg, Heng Guo, and Kuan Yang. Counting solutions to random CNF formulas. In *ICALP*, volume 168 of *LIPIcs*, pages 53:1–53:14, 2020. doi:10.4230/LIPIcs.ICALP.2020.53. 2, 3, 4

[GH20] Heng Guo and Kun He. Tight bounds for popping algorithms. *Random Struct. Algorithms*, 57(2):371–392, 2020. doi:10.1002/rsa.20928. 2

[GJ19] Heng Guo and Mark Jerrum. A polynomial-time approximation algorithm for all-terminal network reliability. *SIAM J. Comput.*, 48(3):964–978, 2019. 2

[GJL19] Heng Guo, Mark Jerrum, and Jingcheng Liu. Uniform sampling through the Lovász local lemma. *J. ACM*, 66(3):18:1–18:31, 2019. (Conference version in *STOC*'17). doi:10.1145/3310131. 2, 3

[GLLZ19] Heng Guo, Chao Liao, Pinyan Lu, and Chihao Zhang. Counting hypergraph colorings in the local lemma regime. *SIAM J. Comput.*, 48(4):1397–1424, 2019. (Conference version in *STOC*'18). doi:10.1137/18M1202955. 2, 3

[GS01] Geoffrey R. Grimmett and David R. Stirzaker. *Probability and random processes*. Oxford University Press, third edition, 2001. 27

[HDSMR16] Bryan He, Christopher De Sa, Ioannis Mitliagkas, and Christopher Ré. Scan order in gibbs sampling: Models in which it matters and bounds on how much. *Advances in neural information processing systems*, 29, 2016. 5

[HN99] Olle Haggstrom and Karin Nelander. On exact simulation of markov random fields using coupling from the past. *Scandinavian Journal of Statistics*, 26(3):395–411, 1999. 3, 6, 24

[HS17] David G. Harris and Aravind Srinivasan. A constructive Lovász local lemma for permutations. *Theory Comput.*, 13:Paper No. 17, 41, 2017. (Conference version in *SODA*'14). doi:10.4086/toc.2017.v013a017. 2

[HS19] David G. Harris and Aravind Srinivasan. The Moser-Tardos framework with partial resampling. *J. ACM*, 66(5):Art. 36, 45, 2019. (Conference version in *FOCS*'13). doi:10.1145/3342222. 2

[HSS11] Bernhard Haeupler, Barna Saha, and Aravind Srinivasan. New constructive aspects of the Lovász local lemma. *J. ACM*, 58(6):28, 2011. (Conference version in *FOCS*'10). doi:10.1145/2049697.2049702. 2, 9

[Hub98] Mark Huber. Exact sampling and approximate counting techniques. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 31–40. ACM, 1998. 3, 6, 24

[Hub04] Mark Huber. Perfect sampling using bounding chains. *Ann. Appl. Probab.*, 14(2):734–753, 2004. 3

[JPV20] Vishesh Jain, Huy Tuan Pham, and Thuy Duong Vuong. Towards the sampling lovász local lemma. *CoRR*, abs/2011.12196, 2020. URL: https://arxiv.org/abs/2011.12196, arXiv:2011.12196. 2, 3

[JPV21] Vishesh Jain, Huy Tuan Pham, and Thuy Duong Vuong. On the sampling lovász local lemma for atomic constraint satisfaction problems. *CoRR*, abs/2102.08342, 2021. URL: https://arxiv.org/abs/2102.08342, arXiv:2102.08342. 2, 3, 5, 6, 7, 15, 36

[JSS20] Vishesh Jain, Ashwin Sah, and Mehtaab Sawhney. Perfectly sampling k$\geq$ (8/3 +o(1))Δ-colorings in graphs. *CoRR*, abs/2007.06360, 2020. URL: https://arxiv.org/abs/2007.06360, arXiv:2007.06360. 3

[JVV86a] Mark Jerrum, Leslie G. Valiant, and Vijay V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoretical Computer Science*, 43:169–188, 1986. 2

[JVV86b] Mark R. Jerrum, Leslie G. Valiant, and Vijay V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoret. Comput. Sci.*, 43:169–188, 1986. doi:10.1016/0304-3975(86)90174-X. 3

[KM11] Kolipaka Kashyap, Babu Rao and Szegedy Mario. Moser and Tardos meet Lovász. In *STOC*, pages 235–244, 2011. doi:10.1145/1993636.1993669. 2

[LP17] David A Levin and Yuval Peres. *Markov chains and mixing times*. American Mathematical Soc., 2017. doi:10.1090/mbk/107. 23

[LS16] Eyal Lubetzky and Allan Sly. Information percolation and cutoff for the stochastic ising model. *Journal of the American Mathematical Society*, 29(3):729–774, 2016. 6

[Moi19] Ankur Moitra. Approximate counting, the Lovász local lemma, and inference in graphical models. *J. ACM*, 66(2):10:1–10:25, 2019. (Conference version in *STOC*'17). doi:10.1145/3268930. 2, 3, 5, 7

[Mos09] Robin A. Moser. A constructive proof of the Lovász local lemma. In *STOC*, pages 343–350, 2009. doi:10.1145/1536414.1536462. 2

[MR98] Michael Molloy and Bruce Reed. Further algorithmic aspects of the local lemma. In *STOC*, pages 524–529, 1998. doi:10.1145/276698.276866. 2

[MT10] Robin A. Moser and Gábor Tardos. A constructive proof of the general Lovász local lemma. *J. ACM*, 57(2):11, 2010. doi:10.1145/1667053.1667060. 2, 9

[PW96] James G. Propp and David B. Wilson. Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random Structures Algorithms*, 9(1-2):223–252, 1996. 4, 6, 24

[She85] James B. Shearer. On a problem of Spencer. *Combinatorica*, 5(3):241–245, 1985. doi:10.1007/BF02579368. 2

[ŠVV09] Daniel Štefankovič, Santosh Vempala, and Eric Vigoda. Adaptive simulated annealing: A near-optimal connection between sampling and counting. *J. ACM*, 56(3):18, 2009. doi:10.1145/1516512.1516520. 3