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 of the CSP has a small number of forbidden local configurations, our algorithm is accurate and efficient: it outputs a *perfect* uniform random solution and its expected running time is *quasilinear* in the number of variables. 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 has two components:

- A simple perfect sampling algorithm using *bounding chains* (Huber, STOC'98; Haggstrom and Nelander, Scandinavian Journal of Statistics'99).
  This sampler is efficient if each variable domain is small.
- A simple but powerful *state tensorization* trick to reduce large domains to smaller ones.
  This trick is a generalization of *state compression* (Feng, He, and Yin, STOC'21).

The crux of our analysis is a simple *information percolation* argument which allows us to achieve bounds even beyond current best approximate samplers (Jain, Pham, and Vuong, ArXiv'21).

Previous related works either use intricate algorithms or need 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 is a collection of constraints defined on a set of variables, and a solution to the CSP is an assignment of variables that satisfies all the constraints. For any given CSP, it is natural to ask the following questions:

- DECISION. Can we decide efficiently if the CSP *has* a solution?

- SEARCH. If the CSP 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 on 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`

The Decision Problem. A fundamental criterion for the existence of 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 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 solution 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 solution from all solutions 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 Lovász Local Lemma

To state the long list of results on sampling LLL, we need the following notations. Given a CSP, let n be the number of variables, k be the maximum number of variables in each constraint, Q be the maximum number of values that each variable can take, Δ be the maximum constraint degree, p be the maximum violation probability of a constraint, and N be the maximum number of forbidden local configurations for each constraint. A constraint is called *atomic* if it only has one forbidden local configuration. A CSP is called atomic if all of its constraints are atomic, i.e., $N = 1$.

For example in Boolean k-CNF formula, each constraint will depend on exactly k Boolean variables and thus $Q = 2, p = 2^{-k}, N = 1$. For hypergraph coloring, each vertex is allowed to choose a color from $\{1, 2, \ldots, Q\}$ and each edge contains exactly k vertices. The constraints require no edge is monochromatic. Therefore $N = Q$, $p = Q^{1-k}$, and Δ equals the maximum edge degree in the hypergraph. For general CSPs, each constraint may depend on different number of variables and each variable may have different domain size.

The sampling LLL turns out to be computationally more challenging than the algorithmic LLL. For example, for k-CNF the Moser-Tardos algorithm can efficiently find a solution if $\Delta \lesssim 2^k$ where $\gtrsim$ informally hides lower order terms. However, it is intractable to approximately sample a uniform solution if $\Delta \gtrsim 2^{k/2}$, 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 $n^{\mathsf{poly}(k\Delta)}$ if $\Delta \lesssim 2^{k/60}$, where they novelly use the algorithmic LLL to *mark/unmark* variables and then convert the problem into solving linear programs of size $n^{\mathsf{poly}(k\Delta)}$. We remark that this algorithm is deterministic if we only need a multiplicative approximation of the number of solutions, 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][1].

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 $\widetilde{O}\left(n^{1.001}\right)$ together with an improved regime $\Delta \lesssim 2^{k/20}$, where $\widetilde{O}$ hides $\mathsf{poly}(N, k, \Delta, Q, \log(n))$. 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 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 $\widetilde{O}\left(n^{1.001}\right)$ if $p^{1/350} \cdot \Delta \lesssim 1/N$. This algorithm is limited to the special cases 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 $p^{1/7} \cdot \Delta \lesssim 1$. They revisit Moitra's mark/unmark framework and use it in an *adaptive* way. This is the first polynomial time algorithm (assuming $\Delta, k, Q = O(1)$) for general CSPs in local lemma regime. By a highly sophisticated information percolation argument, they [JPV21] also prove that the sampling algorithm in [FHY20] runs in time $\widetilde{O}\left(n^{1.001}\right)$ if $p^{0.142} \cdot \Delta \lesssim 1/N$.

Method	k-CNF	Hypergraph Coloring	General CSPs	Time
[Moi19]	$\Delta \lesssim 2^{k/60}$			$n^{\mathsf{poly}(k\Delta)}$
[GLLZ19]		$\Delta \lesssim Q^{k/16}$		$n^{\mathsf{poly}(k\Delta \log(Q))}$
[FGYZ20]	$\Delta \lesssim 2^{k/20}$			$\widetilde{O}\left(n^{1.001}\right)$
[FHY20]	$\Delta \lesssim 2^{k/13}$	$\Delta \lesssim Q^{k/9}$	$p^{1/350} \cdot \Delta \lesssim 1/N$	$\widetilde{O}\left(n^{1.001}\right)$
[JPV20]	$\Delta \lesssim 2^{k/7}$	$\Delta \lesssim Q^{k/7}$	$p^{1/7} \cdot \Delta \lesssim 1$	$n^{\mathsf{poly}(k\Delta \log(Q))}$
[JPV21]	$\Delta \lesssim 2^{0.175k}$	$\Delta \lesssim Q^{k/3}$	$p^{0.142} \cdot \Delta \lesssim 1/N$	$\widetilde{O}\left(n^{1.001}\right)$

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, ŠVV09a], also imply efficient algorithms for (random) approximate counting, which estimates the number of solutions within 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. Though there are known reductions from approximate sampling/counting to perfect sampling [JVV86b], it is unclear how to adopt them here considering the local lemma conditions.

---

[1][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 follows from standard reductions and noticing fixing *bad variables* (defined in [GGGY20]) does not influence their (deterministic) algorithm.

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 is never in question. In contrast, some solution may never be outputted by an approximate sampler. Consider the following simple example: Let $\mathcal{D}_1$ and $\mathcal{D}_2$ be two distributions 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 $1/\sqrt{n} = o(1)$. Thus $\mathcal{D}_2$ is considered a good approximation of $\mathcal{D}_1$ while the last $\sqrt{n}$ items are never sampled. This is indeed the case for [FGYZ20, FHY20, JPV21], and is undesirable if the CSP is used for addressing social problems and the missing solutions are contributed by the minority or the underrepresented. Besides the potential drawback in social fairness, this also leads to the following reasonable worry: is it possible that some solution is inherently harder to find than others. Fortunately our work shows this is not the case.

Perfect sampling is also advantageous for practical purposes and heuristic algorithms. To perform a sampling task, Markov chain is arguably the most common approach. By the convergence theorem for Markov chains [LP17], it is usually easy to show the chain mixes to the desired distribution almost surely. However to provide a good bound for the mixing time is in general a difficult task. On the other hand, if the mixing time is unknown or poorly analyzed, it is not sure when to stop the chain so the output distribution is close enough to the desired one. However given the Markov chain, there are known techniques, like *coupling from the past* [PW96], to convert it into a perfect sampler, which always gives desired distribution when it stops even if we may not know any bounds on its expected running time.

1.2   Our Results

In this paper, we provide perfect samplers for solutions of atomic CSPs in the local lemma regime. Though in previous paragraphs we only focus on sampling perfect uniform solution, our algorithm in fact works for general underlying distribution.

Let Φ be an atomic CSP with variable set V and $|V| = n$ where each $v \in V$ is endowed with a distribution $\mathcal{D}_v$ supported on finite domain Ω_v.

- Let p be the maximum violation probability of a constraint under distribution $\prod_{v \in V} \mathcal{D}_v$.
- Let Δ be the maximum constraint degree.
- Let Q be the maximum size of a variable domain Ω_v.
- Let k be the maximum number of variables that a constraint depends on.

Let μ be the distribution of solutions of Φ under $\prod_{v \in V} \mathcal{D}_v$, i.e.,

$$\mu(\sigma) = \Pr_{\sigma' \sim \prod_{v \in V} \mathcal{D}_v} \left[\sigma' = \sigma \mid \sigma' \text{ is a solution of } \Phi \right] \quad \text{for each } \sigma \in \prod_{v \in V} \Omega_v.$$

The original Lovász local lemma [EL75] states if $p \cdot \Delta \lesssim 1$ then Φ has a solution, i.e., μ is well-defined. Then the algorithmic LLL [MT10] shows one can efficiently find a solution under the same condition. Our main theorem shows one can efficiently sample a solution distributed as μ under a similar condition.

Theorem 1.1 (Theorem 5.11, Informal). *If $p^\gamma \cdot \Delta \lesssim 1/c$ where*

$$\gamma = \frac{3 + \ln(c+1) - \sqrt{\ln^2(c+1) + 6\ln(c+1)}}{9} \quad and \quad c = \max\left\{ 2, \max_{v \in V} \max_{q,q' \in \Omega_v} \frac{\mathcal{D}_v(q)}{\mathcal{D}_v(q')} \right\},$$

then our algorithm runs in expected time $\mathsf{poly}(k, Q, \Delta) \cdot n\log(n)$ *and outputs a random solution distributed perfectly as* μ.

We remark that for the uniform case (i.e., $\mathcal{D}_v$ is the uniform distribution), we have $c = 2$ and $\gamma > 0.145$, which already beats $1/7$ from [JPV20] and 0.142 from [JPV21].

Theorem 1.1 is proved by a black-box reduction, using our state tensorization trick, to the perfect sampling algorithm on small variable domains. It is possible to get improved bounds for specific underlying distributions by a finer analysis. We take the uniform distribution as a starting example and improve 0.145 to 0.175 for general atomic CSPs.

Theorem 1.2 (Corollary 5.18, Informal). *If* $p^{0.175} \cdot \Delta \lesssim 1$ *and each* $\mathcal{D}_v$ *is the uniform distribution, then our algorithm runs in expected time* $\mathsf{poly}(k, Q, \Delta) \cdot n\log(n)$ *and outputs a perfect uniform random solution.*

We remark that this 0.175 also matches previous best bound for approximately uniform sampling solutions of k-CNF formula [JPV21]. In fact, in our analysis binary domains are the worst case for general atomic CSPs: the bound on k-CNF formula is the bottleneck for the bound on general atomic CSPs.

Indeed, Theorem 1.2 can be further improved if variable domains are large. We use hypergraph coloring as an illustrating example, the bound of which matches the current best bound of approximate samplers [JPV21].

Theorem 1.3 (Theorem 5.14, Informal). *Let* H *be a hypergraph on* n *vertices where each edge contains exactly* k *vertices. Let* Δ *be the edge degree of* H. *Assume each vertex can choose a color from* $\{1, 2, \ldots, Q\}$, *and a coloring of vertices is* proper *if it does not produce monochromatic edge.*

If $\Delta \lesssim Q^{k/3}$, *then our algorithm runs in expected time* $\mathsf{poly}(k, Q, \Delta) \cdot n\log(n)$ *and outputs a perfect uniform random proper coloring of* H.

Finally we briefly discuss the connection between our result and other topics.

- **Non-atomic CSPs.** For a non-atomic CSP, let N be the maximum number of forbidden local configurations. We can convert it into an atomic CSP by decomposing the non-atomic constraints to atomic ones as in [FHY20], which only increases the constraint degree Δ to at most $\Delta \cdot N$. Then our perfect sampler is still efficient if N is small.

- **Approximate Sampling.** Our perfect sampler is a Las Vegas algorithm with quasilinear expected running time T. It is well-known that terminating the algorithm after T/ε steps gives an ε-approximate sampler under total variation distance. In particular, the local lemma condition of our approximate sampler is the same as our perfect sampler, which breaks the current best record for atomic CSPs by [JPV20, JPV21].

  We remark that one can obtain a better bound by analyzing the moments of the running time of our perfect sampler. We do not make the effort here since this is not our focus and this may require stronger local lemma conditions.

- **Approximate Counting.** One way to reduce counting to sampling is to start from a CSP with no constraint, then add clauses one by one and use the self-reducibility [JVV86a]. Another strategy is to use the simulated annealing approach developed in [BSVV08, SVV09b, FGYZ20]. Both reductions produce efficient randomized approximate counting algorithms. We refer interested readers to their paper for details.

  Similarly as in the approximate sampling case, the local lemma condition of our approximate counting algorithm is the same as our perfect sampler, which breaks the current best record for atomic CSPs by [JPV20, JPV21].

1.3 Proof Overview

To illustrate the idea, we first focus on sampling a uniform solution of k-CNF formula:

- There are n Boolean variables. Each variable is endowed with the uniform distribution over $\{0,1\}$, and appears in at most d constraints.
- Each constraint is a clause depending on exactly k variables and has exactly one forbidden local assignment.

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 3-CNF formula where $n = 6, m = 3$ and $k = 3, d = 2$.

Similar as previous works to deal with the connectivity issue of Glauber dynamics [Wig19], the first step of our algorithm is to mark variables so every clause has a certain amount of marked and unmarked variables. Let V be the set of variables and $\mathcal{M} \subseteq V$ be the set of marked variables. Then by the local lemma (Theorem 2.3), for any $\sigma \in \{0,1\}^{\mathcal{M}}$ and any $v \in \mathcal{M}$ 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 solution conditioning on $\sigma'(\mathcal{M} \setminus \{v\}) = \sigma(\mathcal{M} \setminus \{v\})$.

We call this *local uniformity*.

To sample a solution *approximately*, previous works [FGYZ20, FHY20, JPV21] simulate an idealized Glauber dynamics P_{Glauber} (Algorithm 7) on the assignments of the marked variables as follows:

(A) Initialize $\sigma(v) \sim \{0,1\}$ uniformly and independently for each $v \in \mathcal{M}$.

(B) Going forward from time 0 to $T \to +\infty$, let v_t be the variable selected at time $t \geq 0$. Iteratively find all clauses that are not yet satisfied by $\sigma(\mathcal{M} \setminus \{v_t\})$ and are connected to v_t (Algorithm 1). Let Φ' be this sub-k-CNF.

  Then update $\sigma(v_t) \leftarrow \sigma'(v_t)$, where $\sigma' \in \{0,1\}^V$ is a uniform random solution of Φ' conditioning on $\sigma'(\mathcal{M} \setminus \{v_t\}) = \sigma(\mathcal{M} \setminus \{v_t\})$. Algorithmically, σ' is provided via rejection sampling (Algorithm 2) on Φ'.

(C) After Step (A) (B), extend σ to unmarked variables $V \setminus \mathcal{M}$ by sampling a uniform random solution conditioning on $\sigma(\mathcal{M})$.

To sample a solution *perfectly*, we simulate *bounding chains* P_{BChains} (Algorithm 5) of P_{Glauber} as follows: The algorithm guarantees at any point, each variable $v \in \mathcal{M}$ is assigned with a value in $\{0, 1, \star\}$ where $\star$ represents uncertainty.

(1) Initialize $\sigma(v) = \star$ for each $v \in \mathcal{M}$.

(2) Going forward from time $-T$ to -1, let v_t be the variable selected at time $-T \leq t < 0$. Iteratively find all clauses that are not yet satisfied by $\sigma(\mathcal{M} \setminus \{v_t\})$ and are connected with v_t (Algorithm 1). Let Φ' be this sub-CSP.

  - If all marked variables connected to v_t in Φ' have value 0 or 1, we say v_t is *coupled*. Then we update $\sigma(v_t)$ by rejection sampling on Φ' and $\sigma(v_t)$ 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 small probability (SafeSampling subroutine in Algorithm 5).

6

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

    – if some marked variable has value $\star$, then we double T and re-run Step (1) (2);[2]

    – otherwise we stop and extend σ to unmarked variables $V \setminus \mathcal{M}$ by sampling a uniform random solution conditioning on $\sigma(\mathcal{M})$.

To simplify the analysis of the algorithm, we use *systematic scan* for P_{Glauber} and P_{BChains} rather than random scan [HDSMR16]. Specifically, at time $t \in \mathbb{Z}$ the algorithm always updates the variable with index $(t \mod m)$ (Algorithm 7) where $m = |\mathcal{M}|$.

Let $\mu^{\mathcal{M}}$ be the distribution of a uniform random solution projected on the marked variables $\mathcal{M}$. Our goal is to prove the following claims for P_{BChains}:

- CORRECTNESS. When we stop in Step (3), $\sigma(\mathcal{M})$ has distribution $\mu^{\mathcal{M}}$ (Subsection 4.2).

- EFFICIENCY. In expectation, each update in Step (2) is efficient. (Subsection 4.1.1)

- COALESCENCE. In expectation, we stop with $T = O(n \log(n))$ (Subsection 4.1.2).

Proof of Correctness. Firstly we show P_{Glauber} converges to $\mu_{\mathcal{M}}$ 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 $|\mathcal{M}|$ consecutive updates as one step. Then it is easy to check P' is aperiodic and irreducible with unique stationary distribution $\mu^{\mathcal{M}}$. After that, we unpack P' to show P_{Glauber} also converges to $\mu^{\mathcal{M}}$ (Lemma 4.22).

Next, 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 $\mu^{\mathcal{M}}$ if we run P_{Glauber} from time $-\infty$ to -1.

- BOUNDING CHAINS. For each $t \in \mathbb{Z}$, if $\sigma(\mathcal{M})$ in P_{BChains} has no $\star$, then the update process is exactly P_{Glauber}. This means P_{BChains} is a *coupling* of P_{Glauber} (Proposition 4.25). Note that we use $\star$ to denote uncertainty which includes all possible assignments that we need to couple. Thus when P_{BChains} stops at time T with $\widehat{\sigma} \in \{0,1\}^{\mathcal{M}}$ at Step (3), any assignment, going through the updates from time $-T$ to -1, converges to $\widehat{\sigma}$.

Combining the two observations above, we know $\widehat{\sigma}$ is distributed exactly as $\mu^{\mathcal{M}}$.

Proof of Efficiency. We first remark that only the rejection sampling is time consuming, and its running time is a geometric distribution with expectation controlled by the local uniformity (Proposition 3.3). Thus to bound its expectation, it suffices to bound the size of Φ' (Proposition 4.11). This uses the same 2-tree argument as in previous works and we briefly explain here.

Since k-CNF has bounded degree, if Φ' is large then we can find a large independent set S of clauses in Φ'. Note that clauses in Φ' are connected, thus we can further assume S is a 2-tree — S will be connected if we link any two clauses in S that are at distance 2. Intuitively, a 2-tree is an independent set that is not very spread out. Then it suffices to union bound the probability that some large 2-tree growing out of v_t survives after previous updates.

One potential pitfall is the total running time of P_{BChains} depends on both T and the running time of each update, where they can be arbitrarily correlated. Thus we need to calculate the second

---

[2] We remark that the randomness is reused. That is, the randomness used for time $t < 0$ is the same one regardless of the starting time $-T$.

moment of the update time (Subsection 4.1.1) and apply Cauchy-Schwarz inequality to break the correlation (Subsection 4.4).

Proof of Coalescence. To upper bound the round T, we employ the *information percolation* argument similarly used in [LS16, HSZ19, JPV21]. For the sake of analysis, we assume Step (2) in P_{BChains} also includes unmarked variables though it does nothing for the update, i.e., v_t is the variable with index $(t \mod n)$ where $n = |V|$.

The crucial observation is the following. If $\sigma(v_{t_0})$ is updated to $\star$ at time t_0, then at that point there must be some variable $u \neq v_{t_0}$ with value $\star$ 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 $u' \neq v_{t_1}$ with value $\star$ 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 $v_{t_{i+1}}$ has value $\star$.

To express constraints through time, we define the extended constraint (e, C) (Definition 4.12), where C is a clause and $e = \{t'_1, \ldots, t'_k\} \subseteq \{-T, \ldots, -1\}$ is a time sequence such that

- $v_{t'_1}, \ldots, v_{t'_k}$ are the variables C depends on,
- $t'_1, \ldots, t'_k$ 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 $t_i \in e^i_1$ and $t_{i+1} \in e^i_{s_i}$ and $e^i_1, \ldots, e^i_{s_i}$ are connected over $\{-T, \ldots, -1\}$. Thus all the edges $\{e^i_j\}_{i,j}$ form a connected sub-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 $n = |V|$, i.e., $\max_{t_1, t_2 \in e} |t_1 - t_2| < n$. Thus the longest path $\mathcal{P}$ in H has length $|\mathcal{P}| = \Omega(T/n)$. On the other hand, Step (2) of P_{BChains} only finds clauses that are *not* satisfied by marked variables, which means each fixed extended constraint (e, C) appears in H with extremely low (roughly 2^{-k}) probability.

Putting everything together, if we do not stop at round T, we should find a path $\mathcal{P}$ of extended constraints of length $|\mathcal{P}| = \Omega(T/n)$. 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 $\mathcal{P}$ form an independent set. Moreover, it is easy to see each extended constraint overlaps with $O(k^2 d)$ many other extended constraints, which provides an upper bound $O(k^2 d)^{|\mathcal{P}|}$ for the number of possible $\mathcal{P}$. By union bound, the probability that we do not stop at round T is roughly

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

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

We remark that to deal with general atomic CSPs, we need to be more careful with the union bound (Subsection 4.1.2). This is because in general atomic CSPs constraints may depend on different number of variables. Nevertheless, our analysis is much simpler than the one in [JPV21]. 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 4.12), is also much simpler than the discrepancy checks used in their argument.

State Tensorization. The marking process is only efficient when the variable domains are small; otherwise it cannot guarantee useful local uniformity. Similarly for $\star$, it compresses too much information when the domain is large. Therefore to deal with large domains, we introduce a simple *state tensorization* trick to perform the reduction.

For intuition, let's consider the following concrete atomic CSP Φ:

- The variables are u, v where u is endowed with distribution $\mathcal{D}_u$ by $\mathcal{D}_u(a) = \mathcal{D}_u(b) = \mathcal{D}_u(c) = 1/3$; and v is endowed with $\mathcal{D}_v$ by $\mathcal{D}_v(A) = \mathcal{D}_v(B) = 1/4, \mathcal{D}_v(C) = 1/3, \mathcal{D}_v(D) = 1/6$.

- The constraints are C_1, C_2 where $C_1 = \mathsf{False}$ iff $u = a$; and $C_2 = \mathsf{False}$ iff $u = c, v = B$.

Then we describe one possible state tensorization as follows (See Figure 1):

- Define variables u_1, u_2 where u_1 is endowed with $\mathcal{D}_{u_1}$ by $\mathcal{D}_{u_1}(0) = 2/3, \mathcal{D}_{u_1}(1) = 1/3$; and u_2 is endowed with $\mathcal{D}_{u_2}(0) = \mathcal{D}_{u_2}(1) = 1/2$.

  We interpret $u = a$ if $(u_1, u_2) = (0,0)$; $u = b$ if $(u_1, u_2) = (0,1)$; and $u = c$ if $u_1 = 1$.

- Define variables v_1, v_2, v_3 where v_1 is endowed with $\mathcal{D}_{v_1}$ by $\mathcal{D}_{v_1}(0) = \mathcal{D}_{v_1}(1) = 1/2$; and v_2 is endowed with $\mathcal{D}_{v_2}(0) = \mathcal{D}_{v_2}(1) = 1/2$; and v_3 is endowed with $\mathcal{D}_{v_3}(0) = 2/3, \mathcal{D}_{v_3}(1) = 1/3$.

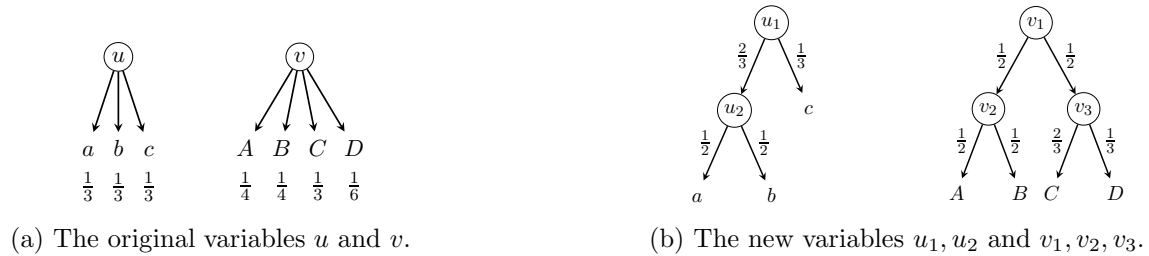 We interpret $v = A$ if $(v_1, v_2) = (0,0)$; $u = C$ if $(v_1, v_3) = (1,0)$, etc.

(a) The original variables u and v.   (b) The new variables u_1, u_2 and v_1, v_2, v_3.

Figure 1: An example for state tensorization.

Hence after the state tensorization, $C_1 = \mathsf{False}$ iff $u_1 = 0, u_2 = 0$; and $C_2 = \mathsf{False}$ iff $u_1 = 1, v_1 = 0, v_2 = 1$. Moreover, sampling the value of u, v from $\mathcal{D}_u \times \mathcal{D}_v$ is equivalent to first sampling the value of u_1, u_2, v_1, v_2, v_3 from $\mathcal{D}_{u_1} \times \mathcal{D}_{u_2} \times \mathcal{D}_{v_1} \times \mathcal{D}_{v_2} \times \mathcal{D}_{v_3}$ and then interpret the value of u, v from them. Therefore, to obtain a random solution under distribution $\mathcal{D}_u \times \mathcal{D}_v$, it suffices to first obtain a random solution under the product distribution of u_1, u_2, v_1, v_2, v_3 and then interpret them back.

Most importantly, this reduction does not change the violation probability of any individual constraint, nor change the dependency relation among constraints. These essentially guarantees that the desired local lemma condition does not deteriorate after the reduction. The formal description of the reduction can be found in Subsection 5.2.

We remark that state tensorization, combined with the marking $\mathcal{M}$, generalizes the state compression technique in [FHY20]. On the other hand, state tensorization is similar to standard gadget reduction in the study of complexity theory. For example by encoding large alphabets using binary bits, one can show Boolean CSPs are no easier to solve than CSPs with large variable domains for polynomial time algorithms. However we are not aware of such simple idea being used for perform sampling tasks.

Organization. We give formal definitions in Section 2. Useful subroutines are provided in Section 3 and then we describe and analyze our main algorithm in Section 4. We discuss our result for different applications in Section 5.

2   Preliminaries

We use $\mathbf{e} \approx 2.71828$ to denote the *natural base*. We use $\log(\cdot)$ and $\ln(\cdot)$ to denote the logarithm with base 2 and $\mathbf{e}$ respectively. We use $[N]$ to denote $\{1, 2, \dots, N\}$; 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 $\mathcal{X}$ and a distribution $\mathcal{D}$ over $\mathcal{X}$, we use $x \sim \mathcal{D}$ to denote that x is a random variable sampled from $\mathcal{X}$ according to distribution $\mathcal{D}$. For two events $\mathcal{E}_1, \mathcal{E}_2$ with $\mathbf{Pr}[\mathcal{E}_2] = 0$, we define the conditional probability $\mathbf{Pr}[\mathcal{E}_1(x) \,|\, \mathcal{E}_2(x)] = 0$. 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 $\sigma_{\mathsf{False}}^C \subseteq \prod_{v \in \mathsf{vbl}(C)} \Omega_v$ (resp., $\sigma_{\mathsf{True}}^C \subseteq \prod_{v \in \mathsf{vbl}(C)} \Omega_v$) to denote the set of falsifying (resp., satisfying) assignments of C. More generally, for $\mathcal{C}$ being a set of constraints, we use $\sigma_{\mathsf{False}}^{\mathcal{C}}$ (resp., $\sigma_{\mathsf{True}}^{\mathcal{C}}$) to denote the set of falsifying (resp., satisfying) assignments of $\mathcal{C}$, i.e., $C(\sigma) = \mathsf{False}$ for all $\sigma \in \sigma_{\mathsf{False}}^{\mathcal{C}}$ and *some* $C \in \mathcal{C}$ (resp., $C(\sigma) = \mathsf{True}$ for all $\sigma \in \sigma_{\mathsf{True}}^{\mathcal{C}}$ and *all* $C \in \mathcal{C}$).

For sampling LLL, we also need to specify the underlying distribution. Assume each $v \in V$ has some distribution $\mathcal{D}_v$ supported on Ω_v.[3] Define $\mu_{\mathsf{True}}^{\mathcal{C}}$ as the distribution of solutions of $\mathcal{C}$ induced by $(\mathcal{D}_v)_{v \in V}$, i.e.,

$$\mu_{\mathsf{True}}^{\mathcal{C}}(\sigma) = \Pr_{\sigma' \sim \prod_{v \in V} \mathcal{D}_v} \left[\sigma' = \sigma \,\middle|\, \sigma' \in \sigma_{\mathsf{True}}^{\mathcal{C}} \right] \quad \text{for each } \sigma \in \prod_{v \in V} \Omega_v.$$

Definition 2.1 ((Atomic) Constraint Satisfaction Problem)**.** A *constraint satisfaction problem* is specified by $\Phi = \left(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C} \right)$ where $\mathcal{C}$ is a set of constraints on V and each $\mathcal{D}_v$ is a distribution supported on Ω_v.

We say Φ is *atomic* if $\left| \sigma_{\mathsf{False}}^C \right| = 1$ for all $C \in \mathcal{C}$. In this case, we abuse the notation to define $\sigma_{\mathsf{False}}^C$ as the unique falsifying assignment of C.

In addition, we define the following measures for Φ:

- the *width* is $k = k(\Phi) = \max_{C \in \mathcal{C}} |\mathsf{vbl}(C)|$;

- the *variable degree* is $d = d(\Phi) = \max_{v \in V} |\{C \in \mathcal{C} \,|\, v \in \mathsf{vbl}(C)\}|$;

- the *constraint degree* is $\Delta = \Delta(\Phi) = \max_{C \in \mathcal{C}} |\{C' \in \mathcal{C} \,|\, \mathsf{vbl}(C) \cap \mathsf{vbl}(C') \neq \emptyset\}|$;[4]

- the *domain size* is $Q = Q(\Phi) = \max_{v \in V} |\Omega_v|$;

- the *maximal individual falsifying probability* is

$$p = p(\Phi) = \max_{C \in \mathcal{C}} \Pr_{\sigma \sim \prod_{v \in \mathsf{vbl}(C)} \mathcal{D}_v} [C(\sigma) = \mathsf{False}] = \max_{C \in \mathcal{C}} \sum_{\sigma \in \sigma_{\mathsf{False}}^C} \prod_{v \in \mathsf{vbl}(C)} \mathcal{D}_v(\sigma(v)).$$

---

[3]This means $\mathcal{D}_v(U) > 0$ for all $U \in \Omega_v$. One natural choice is the uniform distribution.

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

We will simply use k, d, Δ, Q, 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.

Lovász Local Lemma. The Lovász local lemma provides sufficient conditions to guarantee the existence of a solution of CSPs.

Theorem 2.2 ([EL75]). *Let* $\Phi = \big(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\big)$ *be a CSP. If* $\mathbf{e}p\Delta \leq 1$, *then*

$$\Pr_{\sigma \sim \prod_{v \in V} \mathcal{D}_v} \big[\sigma \in \sigma^{\mathcal{C}}_{\mathsf{True}}\big] \geq (1 - \mathbf{e}p)^{|\mathcal{C}|} > 0.$$

The following more general version, first stated in [HSS11], can be proved with minor modification of the original proof of the Lovász local lemma [EL75].

Theorem 2.3 ([HSS11, Theorem 2.1]). *Let* $\Phi = \big(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\big)$ *be a CSP. If* $\mathbf{e}p\Delta \leq 1$, *then* $\sigma^{\mathcal{C}}_{\mathsf{True}} \neq \emptyset$ *and for any constraint B (not necessarily from $\mathcal{C}$) we have*

$$\Pr_{\sigma \sim \mu^{\mathcal{C}}_{\mathsf{True}}} [B(\sigma) = \mathsf{True}] \leq (1 - \mathbf{e}p)^{-|\Gamma(B)|} \Pr_{\sigma \sim \prod_{v \in V} \mathcal{D}_v} [B(\sigma) = \mathsf{True}],$$

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

Hypergraphs. Here we give definitions related with hypergraphs. All the definitions directly translate to graphs when every edge in the hypergraph contains two 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 $\Phi = \big(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\big)$ we naturally view it as a hypergraph $H(\Phi)$ where

$$V(H(\Phi)) = V \quad \text{and} \quad E(H(\Phi)) = \{\mathsf{vbl}(C)\}_{C \in \mathcal{C}}. \tag{1}$$

Similar as the measures of CSPs, we define the following measures for 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 2.4. *Assume H is a connected hypergraph. Then for any $e, e' \in E(H)$, there exists a sequence of edges $e_1, e_2, \ldots, e_\ell$ 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.

Marking. Apart from the CSP $\Phi = \left(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\right)$ itself, our algorithms will also need a subset of V which we call *marking*. Both the correctness and efficiency of our algorithms rely on the marking.

Assume Φ is atomic and recall our measures for Φ from Definition 2.1. We define the following constants given marking $\mathcal{M}$, the meaning of which will be clear as we proceed to the next section:

- The *maximal conditional falsifying probability* of $(\Phi, \mathcal{M})$ is

$$\alpha = \alpha(\Phi, \mathcal{M}) = \max_{C \in \mathcal{C}} \prod_{v \in \mathsf{vbl}(C) \setminus \mathcal{M}} \mathcal{D}_v(\sigma^C_{\mathsf{False}}(v)). \tag{2}$$

- When $\mathbf{e}\alpha \leq 1$, define

  - the *multiplicative bias* of $(\Phi, \mathcal{M})$ as

  $$\beta = \beta(\Phi, \mathcal{M}) = (1 - \mathbf{e}\alpha)^{-d}; \tag{3}$$

  - the *maximal multiplicative-biased falsifying probability* of $(\Phi, \mathcal{M})$ as

  $$\rho = \rho(\Phi, \mathcal{M}) = \max_{C \in \mathcal{C}} \prod_{v \in \mathsf{vbl}(C) \cap \mathcal{M}} \beta \cdot \mathcal{D}_v(\sigma^C_{\mathsf{False}}(v)); \tag{4}$$

  - the *maximal multiplicative-biased unpercolated probability* of $(\Phi, \mathcal{M})$ as

  $$\lambda = \lambda(\Phi, \mathcal{M}) = \max_{C \in \mathcal{C}} |\mathsf{vbl}(C)|^2 \prod_{v \in \mathsf{vbl}(C) \cap \mathcal{M}} \left(\beta \cdot \mathcal{D}_v(\sigma^C_{\mathsf{False}}(v)) + (\beta - 1)\left(|\Omega_v| - 2\right)\right). \tag{5}$$

When context is clear, we will just use $\alpha, \beta, \rho, \lambda$.

Wildcard. We will reserve $\star$ as a wildcard symbol. Our algorithms will use $\star$ to represent all the possibilities in some domain.

Let $\Phi = \left(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\right)$ be a CSP. For each $v \in V$ we assume $\star \notin \Omega_v$ and define $\Omega_v^\star = \Omega_v \cup \{\star\}$. For any $C \in \mathcal{C}$ and $\sigma \in \prod_{v \in V} \Omega_v^\star$ we abuse notation to define

$$C(\sigma) = \begin{cases} \mathsf{False} & \exists \sigma' \in \prod_{v \in V} \Omega_v \text{ such that } C(\sigma') = \mathsf{False} \text{ and } \sigma(v) \in \{\sigma'(v), \star\} \text{ for all } v \in V, \\ \mathsf{True} & \text{otherwise.} \end{cases} \tag{6}$$

Here we define CSPs projected on assignments with $\star$ coordinates. Intuitively this assignment fixes (and only fixes) the non-$\star$ variables for the CSP. But pedantically we provide the following definition.

Definition 2.5 (Projected Constraint Satisfaction Problem)**.** For any $\sigma \in \prod_{v \in V} \Omega_v^\star$, we define the *projected constraint satisfaction problem* $\Phi|_\sigma = \left(V, (\Omega_v|_\sigma, \mathcal{D}_v|_\sigma)_{v \in V}, \mathcal{C}|_\sigma\right)$ by setting

$$(\Omega_v|_\sigma, \mathcal{D}_v|_\sigma) = \begin{cases} (\Omega_v, \mathcal{D}_v) & \sigma(v) = \star, \\ (\{\sigma(v)\}, \text{point distribution}) & \text{otherwise} \end{cases} \quad \text{for all } v \in V$$

and $\mathcal{C}|_\sigma = \{C|_\sigma \mid C \in \mathcal{C}, C(\sigma) = \mathsf{False}\}$ where $C|_\sigma$ has the same evaluation rule as $C \in \mathcal{C}$ but depends on possibly fewer variables, i.e., $\mathsf{vbl}(C|_\sigma) = \{v \in \mathsf{vbl}(C) \mid \sigma(v) = \star\} \subseteq \mathsf{vbl}(C)$. Similarly we sometimes view $C|_\sigma$ as constraint only on $\mathsf{vbl}(C|_\sigma)$.

Recall the measures defined in Definition 2.1. We note the following simple fact.

Fact 2.6. $k(\Phi|_\sigma) \leq k(\Phi)$, $d(\Phi|_\sigma) \leq d(\Phi|_\sigma)$, $\Delta(\Phi|_\sigma) \leq \Delta(\Phi)$, and $Q(\Phi|_\sigma) \leq Q(\Phi)$. Moreover, if Φ is an atomic CSP, then $\Phi|_\sigma$ is an atomic CSP.

3 Useful Subroutines

In this section, we provide some useful subroutines for later reference in our main algorithm.

3.1 A Component Subroutine

Recall notations defined in Definition 2.5. We first set up the following $\texttt{Component}(\Phi, \mathcal{M}, \sigma, u)$ subroutine, which uses $u \in V$ and current assignment $\sigma \in \prod_{v \in V} \Omega_v^\star$ to (hopefully) decompose projected CSP $\Phi|_\sigma$ into two disjoint parts: One containing u and one isolated from u. For our purpose, the input will guarantee that Φ is atomic, $\sigma(u) = \star$, and $\sigma(v) \equiv \star$ for all $v \in V \setminus \mathcal{M}$.

Algorithm 1: The Component subroutine

Input: an atomic CSP $\Phi = \big(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\big)$, a marking $\mathcal{M} \subseteq V$, an assignment
$\quad\quad \sigma \in \big(\prod_{v \in \mathcal{M}} \Omega_v^\star\big) \times \{\star\}^{V \setminus \mathcal{M}}$, and $u \in V$ with $\sigma(u) = \star$
Output: (Φ', Token) where $\Phi' = \big(V', (\Omega_v|_\sigma, \mathcal{D}_v|_\sigma)_{v \in V'}, \mathcal{C}'\big)$ and $\mathsf{Token} \in \{\mathsf{True}, \mathsf{False}\}$

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

Here we note the following observation regarding Algorithm 1.

Proposition 3.1. *The following holds for* $\texttt{Component}(\Phi, \mathcal{M}, \sigma, u)$.

(1) *It runs in time* $O\left(\Delta k |\mathcal{C}'| + dk\right)$.

(2) $u \in V' \subseteq V$, $\mathcal{C}' \subseteq \mathcal{C}|_\sigma$, *and* $\sigma(v) = \star$ *for all* $v \in V'$. *Moreover,* $\sigma(v) = \sigma_{\mathsf{False}}^C(v)$ *holds for any* $C|_\sigma \in \mathcal{C}'$ *and* $v \in (\mathsf{vbl}(C) \cap \mathcal{M}) \setminus \{u\}$.

(3) Φ' *is an atomic CSP and hypergraph* $H(\Phi')$ *(Defined in (1)) is connected.*

(4) *If* $\mathsf{Token} = \mathsf{True}$, *let* $V'' = V \setminus V'$ *and* $\mathcal{C}'' = \mathcal{C}|_\sigma \setminus \mathcal{C}'$. *Then* $\Phi'' = \big(V'', (\Omega_v|_\sigma, \mathcal{D}_v|_\sigma)_{v \in V''}, \mathcal{C}''\big)$ *is an atomic CSP. Moreover* $\sigma_{\mathsf{True}}^{\mathcal{C}|_\sigma} = \sigma_{\mathsf{True}}^{\mathcal{C}'} \times \sigma_{\mathsf{True}}^{\mathcal{C}''}$ *and* $\mu_{\mathsf{True}}^{\mathcal{C}|_\sigma} = \mu_{\mathsf{True}}^{\mathcal{C}'} \times \mu_{\mathsf{True}}^{\mathcal{C}''}$.

Proof. Item (2) is evident from the algorithm, Φ being atomic, and Definition 2.5.

For Item (3), note that each time we add a constraint $C|_\sigma$ into $\mathcal{C}'$, we add $\mathsf{vbl}(C|_\sigma)$ into V'. Thus Φ' is a CSP. Since Φ is atomic, by Fact 2.6 Φ' is also atomic. In addition, we only consider $C|_\sigma$ with $\mathsf{vbl}(C|_\sigma) \cap V' \neq \emptyset$ at Line 2, hence $H(\Phi')$ is connected.

For Item (1), the algorithm can be executed by first checking all (at most d) constraints related with u, then iteratively checking (at most $\Delta|\mathcal{C}'|$ in total) constraints related with the newly added constraints in $\mathcal{C}'$. Thus the total running time is $O(k) \cdot (\Delta|\mathcal{C}'| + d)$.

Now we focus on Item (4) when $\mathsf{Token} = \mathsf{True}$. The condition from Line 2 implies for any $C|_\sigma \in \mathcal{C}''$, $\mathsf{vbl}(C|_\sigma) \cap V' = \emptyset$ and thus $\mathsf{vbl}(C|_\sigma) \subseteq V''$. Therefore Φ'' is a CSP. Since Φ is atomic, by Fact 2.6 Φ'' is also atomic. Then the "moreover" part follows from V (resp., $\mathcal{C}|_\sigma$) being a disjoint union of V' and V'' (resp., $\mathcal{C}'$ and $\mathcal{C}''$). $\qquad\square$

3.2 A RejectionSampling Subroutine

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

Algorithm 2: The RejectionSampling algorithm

    Input: a CSP $\Phi = \left(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\right)$ and a randomness tape r
    Output: an assignment $\sigma \in \mu_{\mathsf{True}}^{\mathcal{C}}$
1 while True **do**
2   Sample $\sigma \sim \prod_{v \in V} \mathcal{D}_v$ with fresh randomness from r
3   **if** $C(\sigma) = \mathsf{True}$ *for all* $C \in \mathcal{C}$ **then return** σ
4 end

We have the following result on [Algorithm 2](#) by basic facts of geometric distributions.

Fact 3.2. *The following holds for* RejectionSampling(Φ, r) *over random* r *if* $\sigma_{\mathsf{True}}^{\mathcal{C}} \neq \emptyset$.

- *It halts almost surely, and outputs* $\sigma \sim \mu_{\mathsf{True}}^{\mathcal{C}}$ *when it halts.*

- *Let* T *be the number of* **while** *iterations it takes before it halts. Then*

$$\mathbb{E}[T] = \frac{1}{\mathbf{Pr}_{\sigma \sim \prod_{v \in V} \mathcal{D}_v}\left[\sigma \in \sigma_{\mathsf{True}}^{\mathcal{C}}\right]} \quad and \quad \mathbb{E}\left[T^2\right] = 2 \cdot (\mathbb{E}[T])^2 - \mathbb{E}[T].$$

- *Let* X *be its total running time.*[5] *Then*

$$\mathbb{E}[X] = O\left(\mathbb{E}[T] \cdot (k|\mathcal{C}| + Q|V|)\right) \quad and \quad \mathbb{E}\left[X^2\right] = O\left(\left(\mathbb{E}[T] \cdot (k|\mathcal{C}| + Q|V|)\right)^2\right).$$

The following result is useful when we perform rejection sampling on a projected CSP, say Φ' from [Algorithm 1](#).

Proposition 3.3. *Let* $\Phi = \left(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\right)$ *be an atomic CSP and* $\mathcal{M} \subseteq V$ *be a marking. Let* $k = k(\Phi)$, $\Delta = \Delta(\Phi)$, $Q = Q(\Phi)$, $\alpha = \alpha(\Phi, \mathcal{M})$, *and* $\beta = \beta(\Phi, \mathcal{M})$. *Let* $\sigma \in \left(\prod_{v \in \mathcal{M}} \Omega_v^{\star}\right) \times \{\star\}^{V \setminus \mathcal{M}}$ *be an arbitrary assignment and* $\Phi' = \left(V', (\Omega_v|_\sigma, \mathcal{D}_v|_\sigma)_{v \in V'}, \mathcal{C}'\right)$ *be an arbitrary sub-CSP of* $\Phi|_\sigma$ *where* $V' \subseteq V$ *and* $\mathcal{C}' \subseteq \mathcal{C}|_\sigma$.

If $\mathbf{e}\alpha\Delta \leq 1$, *then the following holds for* RejectionSampling(Φ', r) *over random* r.

(1) *Let* X *be its running time. Then* $X < +\infty$ *almost surely and, if hypergraph* $H(\Phi')$ *(Defined in [(1)](#)) is connected, we have*

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

(2) *Let* σ' *be its output. Then* $\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}'}$. *Moreover for any* $v \in V'$ *and any* $q \in \Omega_v|_\sigma$, *we have*

$$\mathbf{Pr}\left[\sigma'(v) = q\right] \leq \beta \cdot \mathcal{D}_v|_\sigma(q).$$

---

[5]Each **while** iteration can be performed in time $O(k|\mathcal{C}| + Q|V|)$ where $O(Q) \cdot |V|$ is for Line 2 and $O(k) \cdot |\mathcal{C}|$ is for Line 3.

Proof. Firstly by Fact 2.6, $\Delta(\Phi') \leq \Delta$. Then for any $C|_\sigma \in \mathcal{C}'$, we have

$$
\underset{\widetilde{\sigma} \sim \prod_{v \in \mathsf{vbl}(C|_\sigma)} \mathcal{D}_v|_\sigma}{\mathbf{Pr}} [C|_\sigma(\widetilde{\sigma}) = \mathsf{False}]
$$

$$
= \prod_{v \in \mathsf{vbl}(C|_\sigma)} \mathcal{D}_v(\sigma_{\mathsf{False}}^C(v)) \qquad \text{(since } \Phi \text{ is atomic and by Definition 2.5)}
$$

$$
\leq \prod_{v \in \mathsf{vbl}(C) \setminus \mathcal{M}} \mathcal{D}_v(\sigma_{\mathsf{False}}^C(v)) \qquad \text{(since } \sigma(V \setminus \mathcal{M}) = \star^{V \setminus \mathcal{M}} \text{ and thus } \mathsf{vbl}(C|_\sigma) \subseteq V \setminus \mathcal{M})
$$

$$
\leq \alpha. \qquad \qquad \text{(by (2))}
$$

Thus $p(\Phi') \leq \alpha$ (Recall p from Definition 2.1) and we apply Theorem 2.2 to obtain

$$
\underset{\widetilde{\sigma} \sim \prod_{v \in V'} \mathcal{D}_v|_\sigma}{\mathbf{Pr}} \left[\widetilde{\sigma} \in \sigma_{\mathsf{True}}^{\mathcal{C}'} \right] \geq (1 - \mathbf{e}\alpha)^{|\mathcal{C}'|} > 0.
$$

By Fact 2.6, $k(\Phi') \leq k$. Then Item (1) follows immediately from Fact 3.2 by noticing $|V'| \leq k|\mathcal{C}'| + 1^6$ when $H(\Phi')$ is connected. $\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}'}$ in Item (2) follows from Fact 3.2 as well.

Note that $\beta \geq 1$. Thus by Definition 2.5 we may safely assume $\sigma(v) = \star$ in Item (2). Let $B(\sigma')$ be the event (i.e., constraint) "$\sigma'(v) = q$". Then $\mathsf{vbl}(B) = \{v\}$ and

$$
\underset{\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}'}}{\mathbf{Pr}} \left[B(\sigma') \right] = \underset{\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}'}}{\mathbf{Pr}} \left[\sigma'(v) = q \right] \quad \text{and} \quad \underset{\sigma' \sim \prod_{v \in V'} \mathcal{D}_v|_\sigma}{\mathbf{Pr}} \left[B(\sigma') \right] = \mathcal{D}_v|_\sigma(q).
$$

By Fact 2.6, $d(\Phi') \leq d$. Thus Item (2) follows naturally from Theorem 2.3 by the definition of β and noticing

$$
\left| \left\{ C|_\sigma \in \mathcal{C}' \,\middle|\, \mathsf{vbl}(B) \cap \mathsf{vbl}(C|_\sigma) \neq \emptyset \right\} \right| = \left| \left\{ C|_\sigma \in \mathcal{C}' \,\middle|\, v \in \mathsf{vbl}(C|_\sigma) \right\} \right| \leq d(\Phi') \leq d. \qquad \square
$$

3.3   A SafeSampling Subroutine

The following simple SafeSampling complements RejectionSampling when there is uncertainty to update $u \in \mathcal{M}$. Details will be clear in Subsection 4.2.2.

Algorithm 3: The SafeSampling algorithm

Input: an atomic CSP $\Phi = \big(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\big)$, a marking $\mathcal{M} \subseteq V$, some $u \in \mathcal{M}$, and a randomness tape r

Output: some value $q \in \Omega_u^\star$

1 Recall $\beta = \beta(\Phi, \mathcal{M})$ from (3)

2 Sample c from $\mathcal{D}_u^\star$ using r where $\mathcal{D}_u^\star$ is a distribution over $\Omega_u^\star$ by

$$
\mathcal{D}_u^\star(q) = \begin{cases} \max\left\{0, 1 - \beta \cdot (1 - \mathcal{D}_u(q))\right\} & q \in \Omega_u, \\ 1 - \sum_{q' \in \Omega_u} \mathcal{D}_u^\star(q') & q = \star. \end{cases}
$$

3 return c

We note the following observation regarding SafeSampling.

Proposition 3.4. *The following holds for* $\mathtt{SafeSampling}(\Phi, \mathcal{M}, u, r)$ *over random r if* $\mathbf{e}\alpha\Delta \leq 1$.

---

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

(1) *It runs in time $O(Q)$ where $Q = Q(\Phi)$ from Definition 2.1 and $\mathcal{D}_u^\star$ from Line 2 is a distribution.*

(2) *For any $q \in \Omega_u$, $\mathcal{D}_u^\star(q) \leq \mathcal{D}_u(q)$ and $\mathcal{D}_u^\star(\{q, \star\}) \leq \beta \cdot \mathcal{D}_u(q) + (\beta - 1)(|\Omega_u| - 2)$.*

(3) *Let $\sigma \in \left(\prod_{v \in \mathcal{M}} \Omega_v^\star\right) \times \{\star\}^{V \setminus \mathcal{M}}$ be an arbitrary assignment with $\sigma(u) = \star$. Let $\Phi' = \left(V', (\Omega_v|_\sigma, \mathcal{D}_v|_\sigma)_{v \in V'}, \mathcal{C}'\right)$ be an arbitrary sub-CSP of $\Phi|_\sigma$ where $V' \subseteq V$ and $\mathcal{C}' \subseteq \mathcal{C}|_\sigma$. Let $\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}'}$.[7] Then for any $q \in \Omega_u|_\sigma = \Omega_u$, we have $\mathcal{D}_u^\star|_\sigma(q) \leq \mathbf{Pr}\left[\sigma'(u) = q\right]$.*

Proof. First note that $\beta \geq 1$. For Item (2), observe that

$$\mathcal{D}_u^\star(q) = \max\left\{0, 1 - \beta \cdot (1 - \mathcal{D}_u(q))\right\} \leq \max\left\{0, 1 - 1 \cdot (1 - \mathcal{D}_u(q))\right\} = \mathcal{D}_u(q)$$

and

$$
\begin{aligned}
\mathcal{D}_u^\star(\{q, \star\}) &= 1 - \sum_{q' \in \Omega_u \setminus \{q\}} \mathcal{D}_u^\star(q') \leq 1 - \sum_{q' \in \Omega_u \setminus \{q\}} \left(1 - \beta \cdot (1 - \mathcal{D}_u(q'))\right) \\
&= 1 + (\beta - 1)(|\Omega_u| - 1) - \beta \cdot \sum_{q' \in \Omega_u \setminus \{q\}} \mathcal{D}_u(q') \\
&= 1 - (\beta - 1)(|\Omega_u| - 1) - \beta \cdot (1 - \mathcal{D}_u(q)) = \beta \cdot \mathcal{D}_u(q) + (\beta - 1)(|\Omega_u| - 2).
\end{aligned}
$$

For Item (1), it suffices to observe, using Item (2), that $\mathcal{D}_u^\star(\star) \geq 1 - \sum_{q' \in \Omega_u} \mathcal{D}_u(q') = 0$. Finally for Item (3), we simply have

$$
\begin{aligned}
\mathbf{Pr}\left[\sigma'(u) = q\right] &= 1 - \sum_{q' \in \Omega_u \setminus \{q\}} \mathbf{Pr}\left[\sigma'(u) = q'\right] \\
&\geq 1 - \beta \cdot \sum_{q' \in \Omega_u \setminus \{q\}} \mathcal{D}_u(q') \qquad \text{(by Item (2) of Proposition 3.3 and } \mathcal{D}_u|_\sigma = \mathcal{D}_u) \\
&= 1 - \beta \cdot (1 - \mathcal{D}_u(q)) \\
&\geq \mathcal{D}_u(q) \geq \mathcal{D}_u^\star(q). \qquad\qquad\qquad\qquad\quad \text{(since } \beta \geq 1 \text{ and by Item (2))}
\end{aligned}
$$

$\square$

4   The **AtomicCSPSampling** Algorithm

We now formally describe our main algorithm AtomicCSPSampling in Algorithm 4. The missing subroutines will be provided as we prove the correctness and efficiency of Algorithm 4.

Intuitively, the $\widetilde{\sigma}$ after the while iterations will be a random assignment over $\mathcal{M}$ (i.e., $\widetilde{\sigma} \in \left(\prod_{v \in \mathcal{M}} \Omega_v\right) \times \{\star\}^{V \setminus \mathcal{M}}$) with certain distribution; and the final output σ simply extends the assignment of $\widetilde{\sigma}$ to $V \setminus \mathcal{M}$. Putting them together, we will prove σ is distributed as $\mu_{\mathsf{True}}^{\mathcal{C}}$.

Recall measures $\alpha, \rho, \lambda, k, d, \Delta, Q$ defined in (2), (4), (5), and Definition 2.1. Now we present our main theorem, the proof of which is the focus of the rest of the section.

Theorem 4.1. *Let $\Phi = \left(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\right)$ be an atomic CSP. Let $\mathcal{M} \subseteq V$ be a marking. If $\mathbf{e}\alpha\Delta \leq 1$, $\mathbf{e}\Delta^2\rho \leq 1/32$, and $\Delta^2\lambda \leq 1/16$, then the following holds for* AtomicCSPSampling$(\Phi, \mathcal{M})$.

- CORRECTNESS. *It halts almost surely and outputs $\sigma \sim \mu_{\mathsf{True}}^{\mathcal{C}}$ when it halts.*

- EFFICIENCY. *Its expected total running time is $O\left(kQ\Delta^3|V|\log(|V|)\right)$.*

---

[7] $\mu_{\mathsf{True}}^{\mathcal{C}'}$ is well-defined guaranteed by Item (2) of Proposition 3.3.

Algorithm 4: The AtomicCSPSampling algorithm

Input: an atomic CSP $\Phi = \left(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\right)$ and a marking $\mathcal{M} \subseteq V$
Output: an assignment $\sigma \in \sigma_{\mathsf{True}}^{\mathcal{C}}$

1 Assign infinitely long randomness r_i independently for each $i \in \mathbb{Z}$
2 Initialize $T \leftarrow 1$
3 **while** True **do**
4 $\quad$ $\widetilde{\sigma} \leftarrow$ BoundingChain$(\Phi, \mathcal{M}, -T, r_{-T}, \ldots, r_{-1})$ $\qquad$ /* $\widetilde{\sigma} \in \left(\prod_{v \in \mathcal{M}} \Omega_v^\star\right) \times \{\star\}^{V \setminus \mathcal{M}}$ */
5 $\quad$ **if** $\widetilde{\sigma}(v) \neq \star$ *for all* $v \in \mathcal{M}$ **then** break
6 $\quad$ **else** Update $T \leftarrow 2 \cdot T$
7 **end**
8 $\sigma \leftarrow$ FinalSampling$(\Phi, \mathcal{M}, \widetilde{\sigma})$
9 **return** σ

Remark 4.2. We remark that $\lambda \geq \rho$ since $\beta \geq 1$ and domains have at least 2 elements[8]. Thus we can use, say, $\Delta^2 \lambda \leq 1/100$ to dominate both $\mathbf{e}\Delta^2 \rho \leq 1/32$ and $\Delta^2 \lambda \leq 1/16$ and thus simplify the conditions in Theorem 4.1. This indeed only loses minor factors. However, we choose to present in the current format to make the proofs cleaner as $\mathbf{e}\Delta^2 \rho$ and $\Delta^2 \lambda$ comes from different places.

4.1 The **BoundingChain** Subroutine

Recall our SafeSampling subroutine from Subsection 3.3. We present the BoundingChain subroutine.

Algorithm 5: The BoundingChain subroutine

Input: an atomic CSP $\Phi = \left(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\right)$, a marking $\mathcal{M} \subseteq V$, a starting time $-T$,
$\qquad\quad$ and randomness tapes $r_{-T}, \ldots, r_{-1}$
Output: an assignment $\sigma \in \prod_{v \in V} \Omega_v^\star$

1 Initialize $\sigma(v) \leftarrow \star$ for all $v \in V$ $\qquad\qquad\qquad\qquad$ /* Assume $V = \{v_0, \ldots, v_{n-1}\}$ */
2 **for** $t = -T$ **to** -1 **do**
3 $\quad$ $i_t \leftarrow t \mod n$, and $\sigma(v_{i_t}) \leftarrow \star$ $\qquad\qquad$ /* Update $\sigma(v_{i_t})$ in this round */
4 $\quad$ $(\Phi_t, \mathsf{Token}_t) \leftarrow$ Component$(\Phi, \mathcal{M}, \sigma, v_{i_t})$ where $\Phi_t = \left(V_t, (\Omega_v|_\sigma, \mathcal{D}_v|_\sigma)_{v \in V_t}, \mathcal{C}_t\right)$
5 $\quad$ **if** $v_{i_t} \in V \setminus \mathcal{M}$ **then** Continue
6 $\quad$ **else if** $(v_{i_t} \in \mathcal{M}) \wedge (\mathsf{Token}_t = \mathsf{True})$ **then**
7 $\quad\quad$ $\sigma' \leftarrow$ RejectionSampling(Φ_t, r_t)
8 $\quad\quad$ Update $\sigma(v_{i_t}) \leftarrow \sigma'(v_{i_t})$
9 $\quad$ **else** $\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ /* $(v_{i_t} \in \mathcal{M}) \wedge (\mathsf{Token}_t = \mathsf{False})$ */
10 $\quad\quad$ Update $\sigma(v_{i_t}) \leftarrow$ SafeSampling$(\Phi, \mathcal{M}, v_{i_t}, r_t)$
11 $\quad$ **end**
12 **end**
13 **return** σ

Remark 4.3. Algorithm 5 also works if we ignore variables outside $\mathcal{M}$. This is because $\sigma(V \setminus \mathcal{M})$ is always kept $\star^{V \setminus \mathcal{M}}$. However the current version is more convenient for our analysis in Subsection 4.1.2 and does not influence the running time much.

---

[8]If some domain has only 1 element, then the corresponding distribution must be a point distribution. Thus we can simply fix the variable to this value and simplify the CSP.

We first note the following fact regarding each round of update.

Lemma 4.4. *Let $t \in \{-T, \ldots, -1\}$ be an arbitrary time with $v_{i_t} \in \mathcal{M}$. Let $\sigma \in \left(\prod_{v \in \mathcal{M}} \Omega_v^\star\right) \times \{\star\}^{V \setminus \mathcal{M}}$ be an arbitrary assignment. Let $q \in \Omega_{v_{i_t}}^\star$ be the update of $\sigma(v_{i_t})$ in the t-th* for *iteration of Algorithm 5 over random r_t. If $e\alpha\Delta \leq 1$, then*

(1) q is well-defined almost surely;

(2) for any $q' \in \Omega_{v_{i_t}}$ we have

$$\mathbf{Pr}\left[q = q' \,\middle|\, \sigma, t\right] \leq \beta \cdot \mathcal{D}_{v_{i_t}}(q') \quad and \quad \mathbf{Pr}\left[q \in \{q', \star\} \,\middle|\, \sigma, t\right] \leq \beta \cdot \mathcal{D}_{v_{i_t}}(q') + (\beta - 1)\left(\left|\Omega_{v_{i_t}}\right| - 2\right).$$

Proof. Note that σ uniquely determines Token_t, i.e., whether we update $\sigma(v_{i_t})$ by RejectionSampling or SafeSampling. Thus we only need to consider two possibilities separately.

- RejectionSampling. By Item (3) of Proposition 3.1 and Item (1) of Proposition 3.3, q is well-defined almost surely. Since $\sigma(v_{i_t})$ is set to $\star$ right before the update and q is never $\star$ in RejectionSampling, we have $\mathcal{D}_{v_{i_t}}|_\sigma = \mathcal{D}_{v_{i_t}}$ and, by Item (2) of Proposition 3.3,

$$\mathbf{Pr}\left[q \in \{q', \star\} \,\middle|\, \sigma, t\right] = \mathbf{Pr}\left[q = q' \,\middle|\, \sigma, t\right] \leq \beta \cdot \mathcal{D}_{v_{i_t}}(q').$$

- SafeSampling. By Item (1) of Proposition 3.4, q is always well-defined. Then the two bounds follow from Item (2) of Proposition 3.4. $\qquad\square$

We will show the following result for *one single call* of Algorithm 5.

Proposition 4.5. *If $e\alpha\Delta \leq 1$, $e\Delta^2\rho \leq 1/32$, and $\Delta^2\lambda \leq 1/16$, then the following holds over random $r_{-T}, \ldots, r_{-1}$ for* $\mathtt{BoundingChain}(\Phi, \mathcal{M}, -T, r_{-T}, \ldots, r_{-1})$.

- EFFICIENCY. *Let X_t be the running time of the t-th* for *iteration. Then $X_t < +\infty$ almost surely and $\mathbb{E}\left[X_t^2\right] = O\left(dk^2\Delta^5Q^2\right)$.*

- COALESCENCE. *Let $\mathcal{E}$ be the event "in the returned assignment σ, there exists some $u \in \mathcal{M}$ such that $\sigma(u) = \star$". If $T \geq 2|V| - 1$, we have $\mathbf{Pr}\left[\mathcal{E}\right] \leq 4|V| \cdot 2^{-T/|V|}$.*

4.1.1 Moment Bounds on the Running Time

To establish the efficiency part of Proposition 4.5, we need to control the size of Φ_t in each for iteration. This requires some additional definitions.

Definition 4.6 (2-tree)**.** Let G be an undirected graph. A set of vertices $S \subseteq V(G)$ is a *2-tree* if the following holds.

- $\mathsf{dist}_G(u, v) \geq 2$ 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.[9]

- 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 bound the number of 2-trees and show how to extract a large 2-tree from any connected subgraph.

---

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

Lemma 4.7 ([FGYZ20, Corollary 5.7]). *Let G be a graph with maximum degree d. Then for any $v \in V(G)$ and integer $\ell \geq 1$, the number of 2-trees in G of size ℓ containing v is at most $\left(\mathbf{e}d^2\right)^{\ell-1}/2$.*

Lemma 4.8 ([JPV21, Lemma 4.5]). *Let G be a graph with maximum degree d. Let G' be a connected subgraph of G. Then for any $v \in V(G')$, there exists a 2-tree $S \subseteq V(G')$ with $v \in S$ and size $|S| \geq |V(G')|/(d+1)$.*

Lemma 4.9 ([FGYZ20, Observation 5.5]). *If a graph G has a 2-tree of size $\ell > 1$ containing $v \in V(G)$, then G also has a 2-tree of size $\ell - 1$ containing v.*

The following result is an immediate corollary of Lemma 4.8 and Lemma 4.9.

Corollary 4.10. *Let G be a graph with maximum degree d. Let G' be a connected subgraph of G. Then for any $v \in V(G')$ and any integer $\ell \leq \lceil |V(G')|/(d+1)\rceil$, there exists a 2-tree $S \subseteq V(G')$ with $v \in S$ and size $|S| = \ell$.*

Now we show the following concentration bound.

Proposition 4.11. *Let $d = d(\Phi)$, $\Delta = \Delta(\Phi)$, $\alpha = \alpha(\Phi, \mathcal{M})$, and $\rho = \rho(\Phi, \mathcal{M})$. Assume $\mathbf{e}\alpha\Delta \leq 1$. For any $t \in \{-T, \ldots, -1\}$, recall $\mathcal{C}_t$ from Line 4 of Algorithm 5. Then we have*

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

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

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

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

Let σ and v_{i_t} be the assignment and variable to update at time t respectively. Let G be the subgraph of $\mathsf{Lin}(\Phi)$ induced by vertex set $V(G) = \{C \in \mathcal{C} \mid C|_\sigma \in \mathcal{C}_t\}$. Then by Item (3) of Proposition 3.1, G is a connected subgraph of $\mathsf{Lin}(\Phi)$.[10] For any $C|_\sigma \in \mathcal{C}_t$ with $v_{i_t} \in \mathsf{vbl}(C)$, by Corollary 4.10 there exists a 2-tree $\widetilde{S} \subseteq V(G)$ with $C \in \widetilde{S}$ and size $|\widetilde{S}| = \ell$ provided $\ell \leq \lceil |\mathcal{C}_t|/\Delta\rceil$. Define

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

Then by Lemma 4.7 and noticing there are at most d choices of C, we have

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

By Item (2) of Proposition 3.1, $\sigma(v) = \sigma_{\mathsf{False}}^C(v)$ for any $C|_\sigma \in \mathcal{C}_t$ and $v \in (\mathsf{vbl}(C) \cap \mathcal{M}) \setminus \{v_{i_t}\}$. Note that $\sigma(v)$ is initialized as $\star$. Thus $\sigma(v)$ must be updated before time t. Let $\mathsf{UpdTime}(v, t)$ be the last update time of v before the t-th for iteration in Algorithm 5, and let $\mathcal{E}_{v,C}$ be the event "$\sigma(v)$ is updated to $\sigma_{\mathsf{False}}^C(v)$ in the $\mathsf{UpdTime}(v, t)$-th for iteration". Recall the definition of ρ from (4), then we have[11]

$$\mathbf{Pr}\left[|\mathcal{C}_t| \geq \ell \cdot \Delta\right] \leq \mathbf{Pr}\left[\mathcal{E}_{v,C}, \forall C|_\sigma \in \mathcal{C}_t, \forall v \in (\mathsf{vbl}(C) \cap \mathcal{M}) \setminus \{v_{i_t}\}\right]$$

---

[10]Actually Item (3) of Proposition 3.1 says the subgraph of $\mathsf{Lin}(\Phi|_\sigma)$ induced by vertex set $\mathcal{C}_t$ is connected, which implies G is a connected subgraph of $\mathsf{Lin}(\Phi)$ as $\mathsf{vbl}(C|_\sigma) \subseteq \mathsf{vbl}(C)$.

[11]We remark that for the fourth inequality below, we do *not* assume any independence between $\mathcal{E}_{v,C}$. We simply use the chain rule of conditional probability in the order of time. For example, if $\mathsf{UpdTime}(v_1, t) < \mathsf{UpdTime}(v_2, t)$, then $\mathbf{Pr}\left[\mathcal{E}_{v_1,C_1} \wedge \mathcal{E}_{v_2,C_2}\right] = \mathbf{Pr}\left[\mathcal{E}_{v_1,C_1}\right] \cdot \mathbf{Pr}\left[\mathcal{E}_{v_2,C_2} \mid \mathcal{E}_{v_1,C_1}\right]$ and then we apply Item (2) of Lemma 4.4 twice.

$$\leq \mathbf{Pr}\left[\mathcal{E}_{v,C}, \forall C \in \widetilde{S}, \forall v \in (\mathsf{vbl}(C) \cap \mathcal{M}) \setminus \{v_{i_t}\}\right]$$

$$\leq \sum_{S \in \mathcal{S}} \mathbf{Pr}\left[\mathcal{E}_{v,C}, \forall C \in S, \forall v \in (\mathsf{vbl}(C) \cap \mathcal{M}) \setminus \{v_{i_t}\}\right] \qquad \text{(by union bound)}$$

$$\leq \sum_{S \in \mathcal{S}} \prod_{C \in S} \prod_{v \in (\mathsf{vbl}(C) \cap \mathcal{M}) \setminus \{v_{i_t}\}} \min\left\{1, \beta \cdot \mathcal{D}_v(\sigma_{\mathsf{False}}^C(v))\right\}$$

$$\text{(since } (\mathsf{vbl}(C))_{C \in S} \text{ are pairwise disjoint and by Item (2) of Lemma 4.4)}$$

$$\leq \sum_{S \in \mathcal{S}} \prod_{C \in S, v_{i_t} \notin \mathsf{vbl}(C)} \prod_{v \in \mathsf{vbl}(C) \cap \mathcal{M}} \beta \cdot \mathcal{D}_v(\sigma_{\mathsf{False}}^C(v))$$

$$\leq \sum_{S \in \mathcal{S}} \rho^{|S|-1} \leq \frac{d}{2} \cdot \left(\mathbf{e}\Delta^2 \rho\right)^{\ell-1}. \qquad \qquad \square$$

Now we obtain moment bounds for the running time of each for iteration in Algorithm 5.

Proof of the Efficiency Part of Proposition 4.5. Let Y_t and Z_t be the running time of Line 4 and Line 5-11 respectively. By Item (1) of Proposition 3.1, we have $\mathbb{E}\left[Y_t^2 \mid \mathcal{C}_t\right] = O\left((\Delta k |\mathcal{C}_t| + dk)^2\right)$. By Item (3) of Proposition 3.1, Item (1) of Proposition 3.3, and Item (1) of Proposition 3.4, we also have

$$\mathbb{E}\left[Z_t^2 \mid \mathcal{C}_t\right] = O\left(\max\left\{1, Q^2, \frac{(kQ|\mathcal{C}_t| + Q)^2}{(1 - \mathbf{e}\alpha)^{2 \cdot |\mathcal{C}_t|}}\right\}\right) = O\left(\frac{(kQ|\mathcal{C}_t| + Q)^2}{(1 - \mathbf{e}\alpha)^{2 \cdot |\mathcal{C}_t|}}\right).$$

By Proposition 4.11, we have

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

Since $X_t = Y_t + Z_t + O(1)$ and $(a + b + c)^2 \leq 4 \cdot (a^2 + b^2 + c^2)$, we have

$$\mathbb{E}\left[X_t^2\right] = O\left(1 + \mathbb{E}\left[Y_t^2\right] + \mathbb{E}\left[Z_t^2\right]\right) = O\left(1 + \sum_{L=0}^{+\infty} \mathbf{Pr}\left[|\mathcal{C}_t| = L\right] \mathbb{E}\left[Y_t^2 + Z_t^2 \mid \mathcal{C}_t = L\right]\right)$$

$$\leq O\left(d^2 k^2\right) + \sum_{\ell=0}^{+\infty} \mathbf{Pr}\left[|\mathcal{C}_t| \geq \ell \cdot \Delta\right] \cdot \Delta \cdot O\left(\Delta^2 k^2 \left(\Delta(\ell+1)\right)^2 + \frac{(Qk\Delta(\ell+1))^2}{(1 - \mathbf{e}\alpha)^{2(\ell+1)\Delta}}\right)$$

$$\text{(bucketing } L \in [\ell \cdot \Delta, (\ell+1) \cdot \Delta))$$

$$\leq O\left(d^2 k^2\right) + \sum_{\ell=0}^{+\infty} \mathbf{Pr}\left[|\mathcal{C}_t| \geq \ell \cdot \Delta\right] \cdot \Delta \cdot O\left(\Delta^2 k^2 \left(\Delta(\ell+1)\right)^2 + (Qk\Delta(\ell+1))^2 \cdot 16^{\ell+1}\right)$$

$$\text{(since } \mathbf{e}\alpha \geq 1/\Delta \text{ and } \Delta \geq 2, \text{ we have } (1 - \mathbf{e}\alpha)^\Delta \geq 1/4)$$

$$\leq O\left(d^2 k^2\right) + O\left(d\Delta\right) \sum_{\ell=1}^{+\infty} \left(\frac{1}{32}\right)^{\ell-1} \left(\Delta^2 k^2 \left(\Delta(\ell+1)\right)^2 + (Qk\Delta(\ell+1))^2 \cdot 16^{\ell+1}\right)$$

$$= O\left(d^2 k^2 + dk^2\Delta^5 + dk^2\Delta^3 Q^2\right) = O\left(dk^2\Delta^5 Q^2\right). \qquad \text{(since } d \leq \Delta)$$

$$\square$$

4.1.2 Concentration Bounds for the Coalescence

Here we analyze the coalescence part of Proposition 4.5, which is also the stopping condition for the while iterations in Algorithm 4. We use *information percolation* argument and need additional setup.

We follow the notation convention in Subsection 4.1.1: $V = \{v_0, \ldots, v_{n-1}\}$ and $i_t = t \mod n$ for $t \in \{-T, \ldots, -1\}$. $\mathsf{UpdTime}(v, t)$ is the last update time of v *before* time t, i.e.,

$$\mathsf{UpdTime}(v, t) = \max\left\{-T - 1, \max\left\{t' < t \,\middle|\, v_{i_{t'}} = v\right\}\right\}. \tag{7}$$

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

Definition 4.12 (Extended Constraints). For any $C \in \mathcal{C}$ and $e = \{t_1, \ldots, t_m\} \subseteq \{-T, \ldots, -1\}$, we say (e, C) is an *extended constraint* if the following holds.

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

Fact 4.13. *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

  (2a) $0 \leq t_{\mathsf{max}}(e) - t_{\mathsf{min}}(e) < n$, where $t_{\mathsf{min}}(e) = \min_{t' \in e} t'$;
  (2b) $e = \{\mathsf{UpdTime}(v, t') \mid v \in \mathsf{vbl}(C)\}$ for any $t_{\mathsf{max}} + 1 \leq t' \leq t_{\mathsf{min}} + n$;
  (2c) for any $C' \in \mathcal{C}$, we have

$$\left|\left\{e' \,\middle|\, (e', C') \text{ is an extended constraint with } e \cap e' \neq \emptyset\right\}\right| < 2 \cdot \left|\mathsf{vbl}(C')\right|.$$

Proof. Item (1) is evident from Item (1) of Definition 4.12.

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

$$S = \{-T, \ldots, -1\} \cap \{a_i - j \cdot n \mid i \in [m], j \in \mathbb{Z}\} = \{b_1, b_2, \ldots, b_{T(C)}\}$$

where $-T \leq b_1 < \cdots < b_{T(C)} \leq -1$. Note that $b_i \equiv b_{i+m} \mod n$ for all $i \in [T(C) - m]$. If (e, C) is an extended constraint, then by Item (2) of Definition 4.12, e consists of a consecutive interval of S, i.e., $e = \{b_o, b_{o+1}, \ldots, b_{o+m-1}\}$ for some $o \in [T(C) - m + 1]$. Thus $t_{\mathsf{max}}(e) - t_{\mathsf{min}}(e) = b_{o+m-1} - b_o < n$ (since $b_i \equiv b_{i+m} \mod n$) which verifies Item (2a). Since either $b_o + n = b_{o+m}$ or $b_o + n \geq 0$, we know $\mathsf{UpdTime}(v_{a_i}, t') = \mathsf{UpdTime}(v_{a_i}, b_{o+m-1} + 1)$ for all $i \in [m]$ and $b_{o+m-1} + 1 \leq t' \leq b_o + n$ which verifies Item (2b).

Now we prove Item (2c). By Item (1), assume without loss of generality $\mathsf{vbl}(C) \cap \mathsf{vbl}(C') \neq \emptyset$. Let $m' = |\mathsf{vbl}(C')|$ and $\mathsf{vbl}(C') = \{v_{a'_1}, \ldots, v_{a'_{m'}}\}$ and define

$$S' = \{-T, \ldots, -1\} \cap \{a'_i - j \cdot n \mid i \in [m'], j \in \mathbb{Z}\} = \{b'_1, \ldots, b'_{T(C')}\}$$

where $-T \leq b'_1 < \cdots < b'_{T(C')} \leq -1$. Then similarly, we have $b'_i \equiv b'_{i+m'} \mod n$ for all $i \in [T(C') - m']$ and $e' = \{b'_{o'}, \ldots, b'_{o'+m'-1}\}$ for some $o' \in [T(C') - m' + 1]$. Let $i_{\mathsf{min}} = \min\{i \in [T(C')] \mid b'_i \in e\}$ and $i_{\mathsf{max}} = \max\{i \in [T(C')] \mid b'_i \in e\}$. Then $e \cap e' \neq \emptyset$ iff $i_{\mathsf{min}} \leq o' + m' - 1$ and $i_{\mathsf{max}} \geq o'$. Therefore there are at most $i_{\mathsf{max}} - i_{\mathsf{min}} + m'$ choices of o'. Since $b'_{i_{\mathsf{min}}}, b'_{i_{\mathsf{max}}} \in e$, by Item (2a) we know $b'_{i_{\mathsf{max}}} - b'_{i_{\mathsf{min}}} < n$. Hence $i_{\mathsf{max}} < i_{\mathsf{min}} + m'$. In all, there are at most $2m' - 1$ choices of e'. $\square$

Definition 4.14 (Extended Hypergraph). *Extended hypergraph* $H^{\text{ext}} = (V^{\text{ext}}, E^{\text{ext}})$ has vertex set $V^{\text{ext}} = \{-T, \ldots, -1\}$ and extended constraints as edges:

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

Moreover, we label each edge 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 labels are different.

Define σ_0 as the final returned assignment in Algorithm 5. For each $t \in \{-T, \ldots, -1\}$, let σ_t be the assignment at Line 4 of the t-th for iteration in Algorithm 5. In particular $\sigma_t(v_{i_t}) = \star$ due to Line 3.

Now we present the following algorithm informally described in Subsection 1.3 to sequentially find constraints that are not satisfied during the BoundingChain process. We remark that this algorithm is only for our analysis, and we do *not* run it during AtomicCSPSampling.

Algorithm 6: Find failed constraints during the BoundingChain process

Input: assignments $(\sigma_t)_{t \in \{-T, \ldots, 0\}}$ defined above and some $u \in \mathcal{M}$ with $\sigma_0(u) = \star$
Output: $H' = (V', E')$ where $V' \subseteq V^{\text{ext}}$ and $E' \subseteq E^{\text{ext}}$

1   Set $t_0 \leftarrow \mathsf{UpdTime}(u, 0)$ and initialize $V' \leftarrow \{t_0\}, E' \leftarrow \emptyset$
2   $\mathtt{FailedConstraints}(t_0)$
3   **return** (V', E')
4   **Procedure** $\mathtt{FailedConstraints}(t)$:
5   **if** $t < -T + n - 1$ **then return**   /* $(v_{i_t} \in \mathcal{M}) \wedge (\mathsf{Token}_t = \mathsf{False})$ */
6   Initialize $V_t \leftarrow \{v_{i_t}\}$ and $\mathcal{C}_t \leftarrow \emptyset$
7   **while** $\exists C|_{\sigma_t} \in \mathcal{C}|_{\sigma_t} \setminus \mathcal{C}_t$ with $\mathsf{vbl}(C|_{\sigma_t}) \cap V_t \neq \emptyset$ **do**
8       $e \leftarrow \{\mathsf{UpdTime}(v, t+1) \,|\, v \in \mathsf{vbl}(C)\}$ /* (e, C) is an extended constraint */
9   Update $\mathcal{C}_t \leftarrow \mathcal{C}_t \cup \{C|_{\sigma_t}\}$ and $V_t \leftarrow V_t \cup \mathsf{vbl}(C|_{\sigma_t})$
10   Update $E' \leftarrow E' \cup \{e\}$ and $V' \leftarrow V' \cup e$ /* e is labeled by C */
11   **end**
12   **foreach** $v \in (V_t \cap \mathcal{M}) \setminus \{v_{i_t}\}$ **do**
13      $\mathtt{FailedConstraints}(\mathsf{UpdTime}(v, t))$
14   **end**
15 **end**

We have the following observation regarding Algorithm 6.

Lemma 4.15. *Algorithm 6 halts always. Furthermore, if $T \geq 2n - 1$ then*

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

    (1a) it is an extended constraint,
    (1b) for each $v \in \mathsf{vbl}(C)$, the assignment on v is updated to $\star$ or $\sigma_{\mathsf{False}}^C(v)$ in the $\mathsf{UpdTime}(v, t+1)$-th for iteration in Algorithm 5;

(2) each time we call $\mathtt{FailedConstraints}(t)$, 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 $t_{\mathsf{min}}(e_1) < -T + n - 1$.

22

Proof. Since $\mathsf{UpdTime}(v,t) < t$ for all $v \in V$ and $t \in \{-T, \ldots, 0\}$, Algorithm 6 always halts.

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

- **Item (1a) for t_0.** Note that $0 > t_0 \geq 0 - n \geq -T + n - 1$. Then $\mathsf{UpdTime}(v, t_0 + 1) = t_0 \geq -T$ for $v = v_{i_{t_0}}$; and $\mathsf{UpdTime}(v, t_0 + 1) \geq -T$ for all $v \neq v_{i_{t_0}}$. This means $-T - 1 \notin e$ and thus (e, C) is an extended constraint.

- **Item (1b) for t_0.** Since $C|_{\sigma_{t_0}} \in \mathcal{C}|_{\sigma_{t_0}}$ at Line 7, we know $C(\sigma_{t_0}) = \mathsf{False}$ and, by (6), $\sigma_{t_0}(v) \in \{\sigma_{\mathsf{False}}^C(v), \star\}$ for each $v \in \mathsf{vbl}(C)$. This means, if $v \neq v_{i_{t_0}}$, 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 5. Meanwhile if $v = v_{i_{t_0}}$, then the assignment on v is updated to $\star$ in this $t_0 = \mathsf{UpdTime}(v, t_0 + 1)$-th for iteration, resulting in $\sigma_0(v) = \star$.

To complete the induction, we note that each later call of $\mathtt{FailedConstraints}$ relies on some v from Line 13 when we execute some $\mathtt{FailedConstraints}(\mathsf{UpdTime}(v, t))$. This means $v \in \mathcal{M} \setminus \{v_{i_t}\}$ and $\sigma_t(v) = \star$ and $t \geq -T + n - 1$. Thus the assignment on v is updated to $\star$ in the $-T \leq \mathsf{UpdTime}(v, t)$-th for iteration in Algorithm 5. Then the argument above also goes through with almost no change.

For Item (2), note that V' is initialized as $\{t_0\}$. Upon Line 13, we have $\mathsf{UpdTime}(v, t) = \mathsf{UpdTime}(v, t + 1)$ since $v \neq v_{i_t}$, which has been added into V' by Line 10 earlier.

Now we turn to Item (3). By Item (1), $E' \subseteq E^{\mathsf{ext}}$. Meanwhile by Line 10, E' are edges 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 $\mathtt{FailedConstraints}(t)$ is connected to t. This can be proved by induction: When we find $C|_{\sigma_t}$ satisfying the condition at Line 7, fix some $v' \in \mathsf{vbl}(C|_{\sigma_t}) \cap V_t$. Then for the edge e constructed in the next step, each $\mathsf{UpdTime}(v, t + 1) \in e$ is connected to $\mathsf{UpdTime}(v', t + 1) \in e$. Note that either (A) $v' = v_{i_t}$ and thus $\mathsf{UpdTime}(v', t + 1) = t$, or (B) $\mathsf{UpdTime}(v', t + 1)$ was added into V' in an earlier time and connected to t. Hence each $\mathsf{UpdTime}(v, t + 1) \in e$ is connected to t as desired.

Finally we prove Item (4). Each time we call $\mathtt{FailedConstraints}(t)$, it implies the assignment on $v_{i_t} \in \mathcal{M}$ is updated to $\star$ in the t-th for iteration in Algorithm 5. This means $\mathsf{Token}_t = \mathsf{False}$ and $\mathsf{SafeSampling}$ is performed in Algorithm 5. By comparing Algorithm 1 and $\mathsf{FailedConstraints}$, we know v_{i_t} is connected by falsified constraints to some $v \in (V_t \cap \mathcal{M}) \setminus \{v_{i_t}\}$ that $\sigma_t(v) = \star$. This implies at least one round of Line 13 here will be executed. Thus the recursion of Algorithm 6 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 $t_0 \geq -n$. Hence by Item (3), there exists $e_0, e_1 \in E'$ such that $t_0 \in e_0$ and $t_1 \in e_1$; and $t_{\mathsf{max}}(e_0) \geq t_0 \geq -n$, $t_{\mathsf{min}}(e_1) \leq t_1 < -T + n - 1$. $\qquad\square$

Finally we complete the proof of Proposition 4.5.

Proof of the Coalescence Part of Proposition 4.5. Since σ_0 is defined as the final returned assignment, we know event $\mathcal{E}$ (Defined in Proposition 4.5) is "there exists some $u \in \mathcal{M}$ such that $\sigma_0(u) = \star$". Using Algorithm 6 we obtain $H' = (V', E')$. By Fact 2.4 and Item (3) (4) of Lemma 4.15, there exists a path $\widetilde{e}_1, \widetilde{e}_2, \ldots, \widetilde{e}_\ell \in E' \subseteq E^{\mathsf{ext}}$ with labels $\widetilde{C}_1, \widetilde{C}_2, \ldots, \widetilde{C}_\ell$ such that

(a) $\widetilde{e}_i \cap \widetilde{e}_{i+1} \neq \emptyset$ and thus $t_{\mathsf{min}}(\widetilde{e}_i) \leq t_{\mathsf{max}}(\widetilde{e}_{i+1})$ for all $i \in [\ell - 1]$;

(b) $\widetilde{e}_i \cap \widetilde{e}_j = \emptyset$ for all $i, j \in [\ell]$ with $|i - j| > 1$;

(c) $-n \leq t_{\mathsf{max}}(\widetilde{e}_1) < 0$;

(d) $-T \leq t_{\mathsf{min}}(\widetilde{e}_\ell) < -T + n - 1$.

Meanwhile

$$t_{\mathsf{max}}(\widetilde{e}_1) = t_{\mathsf{max}}(\widetilde{e}_\ell) + \sum_{i=1}^{\ell-1} (t_{\mathsf{max}}(\widetilde{e}_i) - t_{\mathsf{max}}(\widetilde{e}_{i+1}))$$

$$\leq t_{\mathsf{min}}(\widetilde{e}_\ell) + n - 1 + \sum_{i=1}^{\ell-1} (t_{\mathsf{min}}(\widetilde{e}_i) + n - 1 - t_{\mathsf{max}}(\widetilde{e}_{i+1})) \qquad \text{(by Item (2a) of Fact 4.13)}$$

$$\leq (-T + n - 2) + \ell \cdot (n - 1). \qquad \text{(by Item (a) (d) above)}$$

Thus by Item (c) above, we have $\ell \geq -2 + \lceil T/(n-1) \rceil$. For convenience we truncate the tail of the path so that ℓ is the largest even number no more than $-2 + \lceil T/(n-1) \rceil$. Thus $\ell \geq -3 + \lceil T/(n-1) \rceil$. We remark that since the tail is truncated, Item (d) above is not necessarily true now.

Now for any fixed path $e_1, \ldots, e_\ell \in E^{\mathsf{ext}}$ with labels $C_1, \ldots, C_\ell$ satisfying Item (a) (b) (c) above, we bound the probability that this path, denoted by P, appears in H'. Assume

$$\prod_{i=1}^{\ell/2} |\mathsf{vbl}(C_{2i-1})| \leq \prod_{i=1}^{\ell/2} |\mathsf{vbl}(C_{2i})| \tag{8}$$

and the other case can be analyzed similarly. For each $t \in \{-T, \ldots, -1\}$ and $C \in \mathcal{C}$ with $v_{i_t} \in \mathsf{vbl}(C)$ we define $\mathcal{E}_{t,C}$ as the event " the assignment on v_{i_t} is updated to $\star$ or $\sigma^C_{\mathsf{False}}(v)$ in the t-th for iteration in Algorithm 5". By Item (1b) of Lemma 4.15, e_i, labeled by C_i, appearing in H' implies $\bigwedge_{t \in e_i} \mathcal{E}_{t,C_i}$ happens. Recall the definition of λ from (5), then we have[12]

$$\mathbf{Pr}\left[P \text{ appears in } H'\right] \leq \mathbf{Pr}\left[e_{2i} \text{ appears in } H' \text{ with label } C_{2i} \text{ for all } i \in [\ell/2]\right]$$

$$\leq \mathbf{Pr}\left[\bigwedge_{i=1}^{\ell/2} \bigwedge_{t \in e_{2i}} \mathcal{E}_{t,C_{2i}}\right] = \mathbf{Pr}\left[\bigwedge_{i=1}^{\ell/2} \bigwedge_{t \in e_{2i}, v_{i_t} \in \mathcal{M}} \mathcal{E}_{t,C_{2i}}\right]$$

$$\leq \prod_{i=1}^{\ell/2} \prod_{v \in \mathsf{vbl}(C_{2i}) \cap \mathcal{M}} \left(\beta \cdot \mathcal{D}_v(\sigma^C_{\mathsf{False}}(v)) + (\beta - 1)(|\Omega_v| - 2)\right)$$

(since $(e_{2i})_{i \in [\ell/2]}$ are pairwise disjoint and by Item (2) of Lemma 4.4)

$$\leq \prod_{i=1}^{\ell/2} (4\Delta |\mathsf{vbl}(C_{2i})|)^{-2} \leq \prod_{i=1}^{\ell} (4\Delta |\mathsf{vbl}(C_i)|)^{-1}.$$

(since $\Delta^2 \lambda \leq 1/16$ and by (8))

Now it suffices to union bound over all possible paths, i.e., $\mathbf{Pr}[\mathcal{E}] \leq \sum_P \mathbf{Pr}[P \text{ appears in } H']$ where P is some fixed path $e_1, \ldots, e_\ell \in E^{\mathsf{ext}}$ with labels $C_1, \ldots, C_\ell$ satisfying Item (a) (b) (c) above. First by Item (c), there are at most n possible $t_{\mathsf{max}}(e_1)$. Then by Item (1) of Fact 4.13 there are at most d possible C_1 given $t_{\mathsf{max}}(e_1)$. By Item (2) of Definition 4.12 this determines (e_1, C_1). Now given (e_1, C_1), we enumerate the rest of the path by a rooted tree $\mathcal{T}(e_1, C_1)$ constructed as follows:

- $\mathcal{T}(e_1, C_1)$ has depth $2(\ell - 1)$ and the root is labeled with (e_1, C_1).
- Given a node z with label $(e_i, C_i), i \in [\ell - 1]$, we construct the next two layers differently.
    - For each $C_{i+1} \in \mathcal{C}$ with $\mathsf{vbl}(C_i) \cap \mathsf{vbl}(C_{i+1}) \neq \emptyset$, we create a child node z' with label C_{i+1} and link to z.

---

[12]We remark that for the third inequality below, we do *not* assume any independence between $\mathcal{E}_{t,C}$. For example, if $t_1 < t_2$, then $\mathbf{Pr}[\mathcal{E}_{t_1,C_4} \wedge \mathcal{E}_{t_2,C_2}] = \mathbf{Pr}[\mathcal{E}_{t_1,C_4}] \cdot \mathbf{Pr}[\mathcal{E}_{t_2,C_2} \mid \mathcal{E}_{t_1,C_4}]$ and then we apply Item (2) of Lemma 4.4 twice.

    – For each z', assume its label is C_{i+1}. We find some e_{i+1} such that $e_i \cap e_{i+1} \neq \emptyset$ and (e_{i+1}, C_{i+1}) is an extended constraint, and then create a child node z'' with label (e_{i+1}, C_{i+1}) and link to z'.

                (z'', given z', has at most $2 \cdot |\mathsf{vbl}(C_{i+1})|$ possibilities by Item (2c) of Fact 4.13.)

    – We move to each z'' and repeat the construction.

Each leaf of $\mathcal{T}(e_1, C_1)$ represents a path P which satisfies Item (a) (c) already, and

- either, P does not satisfy Item (b) and thus does not contribute to the union bound;
- or, P is valid and $\mathbf{Pr}\left[P \text{ appears in } H'\right] \leq \prod_{i=1}^{\ell} (4\Delta\, |\mathsf{vbl}(C_i)|)^{-1}$ as above.

Now we put weight on each internal node z of $\mathcal{T}(e_1, C_1)$ as follows:

- If z has label (e, C), then its weight is $w(z) = (\sqrt{2}\Delta)^{-1}$.
- Otherwise z has label C, then its weight is $w(z) = \left(2\sqrt{2} \cdot |\mathsf{vbl}(C)|\right)^{-1}$.

This means $\mathbf{Pr}\left[P \text{ appears in } H'\right] \leq (4\Delta)^{-1} \prod_{\text{internal node } z \text{ in } P} w(z)$ for each valid P in $\mathcal{T}(e_1, C_1)$ where the $(4\Delta)^{-1}$ factor is because there are only $\ell - 1$ internal nodes for each case along the path. Thus

$$
\sum_{\text{valid } P \text{ in } \mathcal{T}(e_1,C_1)} \mathbf{Pr}\left[P \text{ appears in } H'\right] \leq \sum_{\text{valid } P \text{ in } \mathcal{T}(e_1,C_1)} (4\Delta)^{-1} \prod_{\text{internal node } z \text{ in } P} w(z)
$$
$$
\leq (4\Delta)^{-1} \sum_{P \text{ in } \mathcal{T}(e_1,C_1)} \prod_{\text{internal node } z \text{ in } P} w(z)
$$
$$
\leq (\sqrt{2})^{-2(\ell-1)}/(4\Delta) = 2^{-\ell}/(2\Delta),
$$

where the last inequality can be proved by induction on the depth and noticing $(\sqrt{2} \cdot w(z))^{-1}$ is at least the number of child nodes of z for each internal node $z \in \mathcal{T}(e_1, C_1)$.

    Putting everything together, we have

$$
\mathbf{Pr}\left[\mathcal{E}\right] \leq \sum_{\text{valid } (e_1,C_1)} \sum_{\text{valid } P \text{ in } \mathcal{T}(e_1,C_1)} \mathbf{Pr}\left[P \text{ appears in } H'\right]
$$
$$
\leq nd \cdot 2^{-\ell}/(2\Delta) \qquad\qquad ((e_1, C_1) \text{ has at most } nd \text{ choices})
$$
$$
\leq n \cdot 2^{2 - \frac{T}{n-1}} \leq 4n \cdot 2^{-\frac{T}{n}}. \qquad (\text{since } \ell \geq -3 + \lceil T/(n-1) \rceil \text{ and } d \leq \Delta)
$$

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ $\square$

4.2 The Distribution after **BoundingChain** Subroutines

Recall in `AtomicCSPSampling`$(\Phi, \mathcal{M})$ (Algorithm 4), we keep doubling T and performing the corresponding `BoundingChain`$(\Phi, \mathcal{M}, -T, r_{-T}, \dots, r_{-1})$ until the returned assignment has no $\star$ on $\mathcal{M}$. Thus before we present the `FinalSampling` subroutine, we pause for now to analyze these BoundingChain calls *in a whole*.

Definition 4.16 (Projected Distribution). Given an atomic CSP $\Phi = \left(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\right)$ and a marking $\mathcal{M} \subseteq V$, let $\Lambda = \left(\prod_{v \in \mathcal{M}} \Omega_v\right) \times \{\star\}^{V \setminus \mathcal{M}}$ and define the *projected distribution* $\mu^{\mathcal{M}} \in \mathbb{R}^{\Lambda}$ by

$$
\mu^{\mathcal{M}}(\sigma) = \Pr_{\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}}} \left[\sigma'(\mathcal{M}) = \sigma(\mathcal{M})\right] \quad \text{for all } \sigma \in \Lambda.
$$

We will show the distribution after all BoundingChain subroutines is exactly $\mu^{\mathcal{M}}$.

Proposition 4.17. *If* $\mathrm{e}\alpha\Delta \leq 1$, $\mathrm{e}\Delta^2\rho \leq 1/32$, *and* $\Delta^2\lambda \leq 1/16$, *then the* while *iterations in* AtomicCSPSampling$(\Phi, \mathcal{M})$ *halts almost surely and the final assignment* $\widetilde{\sigma}$ *has distribution* $\mu^{\mathcal{M}}$.

We introduce and analyze the following SystematicScan$(\Phi, \mathcal{M}, \sigma_{\mathsf{in}}, L, R, r_L, \ldots, r_R)$ algorithm, then show it couples with BoundingChain.

Algorithm 7: The SystematicScan algorithm

    Input: an atomic CSP $\Phi = \big(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\big)$, a marking $\mathcal{M} \subseteq V$, an assignment
        $\sigma_{\mathsf{in}} \in \big(\prod_{v \in \mathcal{M}} \Omega_v\big) \times \{\star\}^{V \setminus \mathcal{M}}$, a starting time L, a stopping time R, and
        randomness tapes $r_L, \ldots, r_R$.
    Output: an assignment $\sigma \in \big(\prod_{v \in \mathcal{M}} \Omega_v\big) \times \{\star\}^{V \setminus \mathcal{M}}$.

1   Initialize $\sigma \leftarrow \sigma_{\mathsf{in}}$
2   **for** $t = L$ **to** R **do**   `/* Assume V = {v₀,…,vₙ₋₁} */`
3      $i_t \leftarrow t \mod n$, and $\sigma(v_{i_t}) \leftarrow \star$ `/* Update σ(vᵢₜ) in this round */`
4      $\Phi_t = \big(V_t, (\Omega_v|_\sigma, \mathcal{D}_v|_\sigma)_{v \in V_t}, \mathcal{C}_t\big) \leftarrow$ Component$(\Phi, \mathcal{M}, \sigma, v_{i_t})$
                      `/* Ignore the returned Token since it is always True here */`
5   **if** $v_{i_t} \in V \setminus \mathcal{M}$ **then** Continue
6   **else**
7          $\sigma' \leftarrow$ RejectionSampling(Φ_t, r_t)
8   Update $\sigma \leftarrow \sigma'(v_{i_t})$
9   **end**
10 **end**
11 **return** σ

4.2.1   Convergence of SystematicScan

We first show SystematicScan converges to $\mu^{\mathcal{M}}$. We set up basic notations for Markov chains.

Let Λ be a finite state space; for our purpose it will be $\big(\prod_{v \in \mathcal{M}} \Omega_v\big) \times \{\star\}^{V \setminus \mathcal{M}}$. We view any distribution μ over Λ as a *horizontal* vector in $\mathbb{R}^\Lambda$ where $\sum_{a \in \Lambda} \mu(a) = 1$ and $\mu(a) \geq 0$ holds for all $a \in \Lambda$. We denote $\mathbb{1}_a \in \mathbb{R}^\Lambda$ as the point distribution of $a \in \Lambda$, i.e., $\mathbb{1}_a(b) = 1$ iff $a = b$.

A *Markov chain* $(X_t)_{t \geq 0}$ over Λ is given by its transition matrices $(\mathsf{P}_t)_{t \geq 0}$ 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 $t \geq 0$, then $(X_t)_{t \geq 0}$ is a *time homogeneous Markov chain* given by P;
- if $(\mathsf{P}_t)_{t \geq 0}$ are possibly different, then $(X_t)_{t \geq 0}$ is a *time inhomogeneous Markov chain*.

Assume $(X_t)_{t \geq 0}$ is a time homogeneous Markov chain over Λ given by transition matrix P. We say P is

- *irreducible* if for any $a, b \in \Lambda$, there exists some integer $t \geq 0$ such that $\mathsf{P}^t(a, b) > 0$;
- *aperiodic* if for any $a \in \Lambda$, $\gcd\big\{\text{integer } t > 0 \,\big|\, \mathsf{P}^t(a, a) > 0\big\} = 1$;[13]
- *stationary with respect to distribution* μ if $\mu\mathsf{P} = \mu$;

---

[13] gcd stands for greatest common divisor.

- *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.

Fact 4.18 (e.g., [LP17, Proposition 1.20]). *If P is reversible with respect to μ, then P is also stationary with respect to μ.*

Theorem 4.19 (The Convergence Theorem, e.g., [LP17, Theorem 4.9]). *Suppose $(X_t)_{t\geq 0}$ 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 $\lim_{t \to +\infty} X_t = \mu$.[14]*

Now we turn to SystematicScan and follow the notation convention in Algorithm 7: $V = \{v_0, \ldots, v_{n-1}\}$ and $i_t = t \mod n$. Recall we also fix $\Lambda = \left(\prod_{v \in \mathcal{M}} \Omega_v\right) \times \{\star\}^{V \setminus \mathcal{M}}$.

Definition 4.20 (One-step Transition Matrix). *For any $i \in \{0, \ldots, n-1\}$, define the one-step transition matrix on $v_i \in V$ as $\mathsf{P}_i \in \mathbb{R}^{\Lambda \times \Lambda}$ where*

$$\mathsf{P}_i(\sigma_1, \sigma_2) = \Pr_{\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}}} \left[\sigma'(\mathcal{M}) = \sigma_2(\mathcal{M}) \,\middle|\, \sigma'(\mathcal{M} \setminus \{v_i\}) = \sigma_1(\mathcal{M} \setminus \{v_i\}) \right].$$

We first prove some useful facts and also connect one-step transition matrices to SystematicScan.

Proposition 4.21. *If $\mathrm{e}\alpha\Delta \leq 1$, then the following holds.*

(1) *Each P_i is well-defined.*

(2) *For any $i \in \{0, \ldots, n-1\}$ and $\sigma_1, \sigma_2 \in \Lambda$, we have $\mathsf{P}_i(\sigma_1, \sigma_2) \equiv 0$ if $\sigma_1(\mathcal{M} \setminus \{v_i\}) \neq \sigma_2(\mathcal{M} \setminus \{v_i\})$; and $\mathsf{P}_i(\sigma_1, \sigma_2) > 0$ if otherwise.*

(3) *$\mu^{\mathcal{M}} \mathsf{P}_{i_1} \cdots \mathsf{P}_{i_m} = \mu^{\mathcal{M}}$ holds for any sequence $i_1, i_2, \ldots, i_m \in \{0, \ldots, n-1\}$ of finite length m.*

(4) *For any $L \leq R$ and σ_{in}, SystematicScan$(\Phi, \mathcal{M}, \sigma_{\mathsf{in}}, L, R, r_L, \ldots, r_R)$ halts almost surely over random $r_L, \ldots, r_R$ and its output distribution is $\mu = \mathbb{1}_{\sigma_{\mathsf{in}}} \mathsf{P}_{i_L} \mathsf{P}_{i_{L+1}} \cdots \mathsf{P}_{i_R}$.*

Proof. First we prove Item (1) (2). For any fixed $i \in \{0, \ldots, n-1\}$ and $\sigma_1 \in \Lambda$, define assignment σ by setting $\sigma(V \setminus \{v_i\}) = \sigma_1(V \setminus \{v_i\})$ and $\sigma(v_i) = \star$. Then for any $\sigma_2 \in \Lambda$ we have

$$\mathsf{P}_i(\sigma_1, \sigma_2) = \Pr_{\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}|\sigma}} \left[\sigma'(\mathcal{M}) = \sigma_2(\mathcal{M}) \right],$$

where $\mathcal{C}|_\sigma$ is defined in Definition 2.5. Thus Item (1) is equivalent to $\mu_{\mathsf{True}}^{\mathcal{C}|\sigma}$ being well-defined, which follows from Proposition 3.3. Since $\sigma(\mathcal{M})$ has no $\star$, $\sigma'(\mathcal{M} \setminus \{v_i\}) \equiv \sigma(\mathcal{M} \setminus \{v_i\}) \equiv \sigma_1(\mathcal{M} \setminus \{v_i\})$. Thus the first part of Item (2) follows naturally. As for the second part, if $\sigma_1(V\setminus\{v_i\}) = \sigma_2(V\setminus\{v_i\})$, then

$$\mathsf{P}_i(\sigma_1, \sigma_2) = \frac{\Pr_{\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}}} \left[\sigma'(\mathcal{M}) = \sigma_2(\mathcal{M}) \right]}{\Pr_{\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}}} \left[\sigma'(\mathcal{M} \setminus \{v_i\}) = \sigma_1(\mathcal{M} \setminus \{v_i\}) \right]}. \tag{9}$$

Thus $\mathsf{P}_i(\sigma_1, \sigma_2) > 0$ iff the enumerator is positive, which is equivalent to $\mu_{\mathsf{True}}^{\mathcal{C}|\sigma_2}$ being well-defined and is, again, guaranteed by Proposition 3.3.

Now we prove Item (3). Since $\mu^{\mathcal{M}} \mathsf{P}_{i_1} \cdots \mathsf{P}_{i_m} = \left(\mu^{\mathcal{M}} \mathsf{P}_{i_1}\right) \mathsf{P}_{i_2} \cdots \mathsf{P}_{i_m}$, it suffices to show for $m = 1$ and then apply induction. By Fact 4.18, 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.,

$$\mu^{\mathcal{M}}(\sigma_1)\mathsf{P}_i(\sigma_1, \sigma_2) = \mu^{\mathcal{M}}(\sigma_2)\mathsf{P}_i(\sigma_2, \sigma_1) \quad \text{for any } \sigma_1, \sigma_2 \in \Lambda. \tag{10}$$

---

[14]The convergence is entry-wise.

By Item (2), we may safely assume $\sigma_1(\mathcal{M} \setminus \{v_i\}) = \sigma_2(\mathcal{M} \setminus \{v_i\})$ and observe that

$$\mu^{\mathcal{M}}(\sigma_1)\mathsf{P}_i(\sigma_1, \sigma_2)$$

$$= \Pr_{\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}}} \left[\sigma'(\mathcal{M}) = \sigma_1(\mathcal{M})\right] \cdot \frac{\Pr_{\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}}} \left[\sigma'(\mathcal{M}) = \sigma_2(\mathcal{M})\right]}{\Pr_{\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}}} \left[\sigma'(\mathcal{M} \setminus \{v_i\}) = \sigma_1(\mathcal{M} \setminus \{v_i\})\right]} \qquad \text{(by (9))}$$

$$= \Pr_{\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}}} \left[\sigma'(\mathcal{M}) = \sigma_2(\mathcal{M})\right] \cdot \frac{\Pr_{\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}}} \left[\sigma'(\mathcal{M}) = \sigma_1(\mathcal{M})\right]}{\Pr_{\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}}} \left[\sigma'(\mathcal{M} \setminus \{v_i\}) = \sigma_2(\mathcal{M} \setminus \{v_i\})\right]}$$

$$= \mu^{\mathcal{M}}(\sigma_2)\mathsf{P}_i(\sigma_2, \sigma_1).$$

Finally we turn to Item (4). By induction on R, it suffices to verify for $L = R = t \in \mathbb{Z}$ that $\mathbf{1}_{\sigma_{\mathsf{in}}}\mathsf{P}_{i_t} = \mathsf{P}_{i_t}(\sigma_{\mathsf{in}}, \cdot)$ is the distribution of $\sigma_{\mathsf{out}} \leftarrow \texttt{SystematicScan}(\Phi, \mathcal{M}, \sigma_{\mathsf{in}}, t, t, r_t)$. Similarly as above, define assignment $\widetilde{\sigma}$ by setting $\widetilde{\sigma}(V \setminus \{v_{i_t}\}) = \sigma_{\mathsf{in}}(V \setminus \{v_{i_t}\})$ and $\widetilde{\sigma}(v_{i_t}) = \star$. Then for each $\widehat{\sigma} \in \Lambda$ we have

$$\mathsf{P}_{i_t}(\sigma_{\mathsf{in}}, \widehat{\sigma}) = \Pr_{\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}|\widetilde{\sigma}}} \left[\sigma'(\mathcal{M}) = \widehat{\sigma}(\mathcal{M})\right].$$

Now we consider two separate cases.

- $\boxed{v_{i_t} \in V \setminus \mathcal{M}.}$ Then $\sigma'(\mathcal{M}) \equiv \widetilde{\sigma}(\mathcal{M})$. Thus $\mathsf{P}_{i_t}(\sigma_{\mathsf{in}}, \widehat{\sigma})$ equals 1 if $\widehat{\sigma} = \sigma_{\mathsf{in}}$; and equals 0 if otherwise. This agrees with $\sigma_{\mathsf{out}} \equiv \sigma_{\mathsf{in}}$ from the algorithm.

- $\boxed{v_i \in \mathcal{M}.}$ Let $\Phi_t = \left(V_t, (\Omega_v|_{\widehat{\sigma}}, \mathcal{D}_v|_{\widehat{\sigma}})_{v \in V_t}, \mathcal{C}_t\right)$ be from Line 4. Then by Item (2) of Proposition 3.3, we have

$$\Pr[\sigma_{\mathsf{out}} = \widehat{\sigma}] = \begin{cases} \Pr_{\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}_t}} \left[\sigma'(v_{i_t}) = \widehat{\sigma}(v_{i_t})\right] & \widehat{\sigma}(V \setminus \{v_{i_t}\}) = \widetilde{\sigma}(V \setminus \{v_{i_t}\}), \\ 0 & \text{otherwise.} \end{cases}$$

By Item (4) of Proposition 3.1, $\Pr_{\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}_t}} \left[\sigma'(v_{i_t}) = \widehat{\sigma}(v_{i_t})\right] = \Pr_{\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}|\widetilde{\sigma}}} \left[\sigma'(v_{i_t}) = \widehat{\sigma}(v_{i_t})\right]$. Thus

$$\Pr[\sigma_{\mathsf{out}} = \widehat{\sigma}] = \begin{cases} \Pr_{\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}|\widetilde{\sigma}}} \left[\sigma'(v_{i_t}) = \widehat{\sigma}(v_{i_t})\right] & \widehat{\sigma}(V \setminus \{v_{i_t}\}) = \widetilde{\sigma}(V \setminus \{v_{i_t}\}), \\ 0 & \text{otherwise} \end{cases}$$

$$= \Pr_{\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}|\widetilde{\sigma}}} \left[\sigma'(\mathcal{M}) = \widehat{\sigma}(\mathcal{M})\right] \quad (\text{since } \widehat{\sigma} \in \Lambda \text{ and } \sigma'(\mathcal{M} \setminus \{v_{i_t}\}) \equiv \widetilde{\sigma}(\mathcal{M} \setminus \{v_{i_t}\}))$$

$$= \mathsf{P}_{i_t}(\sigma_{\mathsf{in}}, \widehat{\sigma}). \qquad \qquad \square$$

Item (4) of Proposition 4.21 shows 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 4.22. *Assume* $\mathrm{e}\alpha\Delta \leq 1$. *Let* L *and* $\sigma_{\mathsf{in}} \in \Lambda$ *be arbitrary. Define* $\mu_R = \mathbf{1}_{\sigma_{\mathsf{in}}}\mathsf{P}_{i_L} \cdots \mathsf{P}_{i_R}$. *Then* $\lim_{R \to +\infty} \mu_R = \mu^{\mathcal{M}}$.

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 $\mathbf{1}_{\sigma_{\mathsf{in}}}$ has period n. Hence $\mu_R = \mathbf{1}_{\sigma_{\mathsf{in}}}\mathsf{F}^m$ if $R = L + m \cdot n - 1$ and $m \geq 1$. Let $Y_0 = \mathbf{1}_{\sigma_{\mathsf{in}}}$ and $Y_t = Y_0\mathsf{F}^i$ for $t \geq 1$, then $(Y_t)_{t \geq 0}$ is a time homogeneous Markov chain with transition matrix F. Here we verify the following properties of F.

- STATIONARY WITH RESPECT TO $\mu^{\mathcal{M}}$. This follows immediately from Item (3) of Proposition 4.21.

- APERIODIC. By Item (2) of Proposition 4.21, 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

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

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 Item (2) of Proposition 4.21. Thus $\mathsf{F}(\sigma_1, \sigma_2) \geq \mathsf{P}_{i_L}(\sigma^0, \sigma^1)\mathsf{P}_{i_{L+1}}(\sigma^1, \sigma^2) \cdots \mathsf{P}_{i_{L+n-1}}(\sigma^{n-1}, \sigma^n) > 0$.

Therefore by Theorem 4.19, $\lim_{m \to +\infty} \mu_{L+m\cdot n-1} = \lim_{t \to +\infty} Y_t = \mu^{\mathcal{M}}$. Since each P_i is stationary with respect to $\mu^{\mathcal{M}}$ by Item (3) of Proposition 4.21, for any finite integer $o \geq 0$

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

Hence $\lim_{R \to +\infty} \mu_R = \mu^{\mathcal{M}}$. $\qquad\square$

4.2.2 Coupling from the Past and the Bounding Chain

We have showed SystematicScan *converges* to distribution $\mu^{\mathcal{M}}$, but to obtain a sample distributed *exactly* according to $\mu^{\mathcal{M}}$ 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 $a, b \in \Lambda$, $\mathbf{Pr}_{r \sim [0,1]}[f(a, r) = b] = \mathsf{P}(a, b)$. 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 4.20 and $i_t = t \mod n$ from Algorithm 7.

Lemma 4.23 (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* $-\infty < L \leq R < +\infty$ *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 $M \geq 1$ *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* $\mu^{\mathcal{M}}$ *if* $M < +\infty$ *almost surely.*

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

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

Thus for any $a, b \in \Lambda$, by Lemma 4.22 we have

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

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

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

Therefore to obtain a perfect sample from $\mu^{\mathcal{M}}$ we only need to (1) design a coupling, then (2) sample randomness tapes $(r_t)_{t \geq 0}$, and lastly (3) find some $M' \geq 1$ such that $F_{-M',-1}$ is a constant function and output the corresponding constant. Now we show in the following Algorithm 8 that Algorithm 5 implicitly provides a coupling for (1) and an efficient way to check (3).

Recall $\Lambda = \left(\prod_{v \in \mathcal{M}} \Omega_v\right) \times \{\star\}^{V \setminus \mathcal{M}}$. We define $\Lambda^\star = \left(\prod_{v \in \mathcal{M}} \Omega_v^\star\right) \times \{\star\}^{V \setminus \mathcal{M}}$. Then we construct $g_i \colon \Lambda^\star \times [0,1] \to \Lambda^\star$ for each $i \in \{0, \ldots, n-1\}$ as follows. We also remark that this coupling is information-theoretic and is for analysis only.

Algorithm 8: A coupling $g_i \colon \Lambda^\star \times [0,1] \to \Lambda^\star$ for each $i \in \{0, \ldots, n-1\}$

Input: an assignment $\sigma_{\mathsf{in}} \in \Lambda^\star$ and a randomness tape $r \in [0,1]$.
Output: an assignment $\sigma_{\mathsf{out}} \in \Lambda^\star$.
Data: an atomic CSP $\Phi = \left(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\right)$, a marking $\mathcal{M} \subseteq V$, and an index
    $i \in \{0, \ldots, n-1\}$. Assume $V = \{v_0, \ldots, v_{n-1}\}$.

1 Initialize $\sigma_{\mathsf{out}} \leftarrow \sigma_{\mathsf{in}}$ and divide r into two independent parts r_1, r_2
2 if $v_i \in V \setminus \mathcal{M}$ **then return** σ_{out}
3 Recall $\beta = \beta(\Phi, \mathcal{M})$ from (3) and define distribution $\mathcal{D}_{v_i}^\star$ over $\Omega_{v_i}^\star$ by setting

$$\mathcal{D}_{v_i}^\star(q) = \begin{cases} \max\{0, 1 - \beta \cdot (1 - \mathcal{D}_{v_i}(q))\} & q \in \Omega_{v_i}, \\ 1 - \sum_{q' \in \Omega_{v_i}} \mathcal{D}_{v_i}^\star(q') & q = \star. \end{cases}$$

4 Sample $c_1 \sim \mathcal{D}_{v_i}^\star$ using r_1 and update $\sigma_{\mathsf{out}}(v_i) \leftarrow c_1$
5 if $c_1 \neq \star$ **then return** σ_{out}
6 $(\Phi', \mathsf{Token}) \leftarrow \texttt{Component}(\Phi, \mathcal{M}, \sigma_{\mathsf{out}}, v_i)$ where $\Phi' = \left(V', (\Omega_v|_{\sigma_{\mathsf{out}}}, \mathcal{D}_v|_{\sigma_{\mathsf{out}}})_{v \in V'}, \mathcal{C}'\right)$
7 if $\mathsf{Token} = \mathsf{False}$ **then return** σ_{out}
8 Define distribution $\mathcal{D}_{v_i}^\dagger$ over Ω_{v_i} by setting $\mathcal{D}_{v_i}^\dagger(q) = \mathbf{Pr}_{\sigma' \sim \mu_{\mathsf{True}}^{\mathcal{C}'}}[\sigma'(v_i) = q]$ for $q \in \Omega_{v_i}$.
9 Define distribution $\mathcal{D}_{v_i}'$ over Ω_{v_i} by setting $\mathcal{D}_{v_i}'(q) = \frac{\mathcal{D}_{v_i}^\dagger(q) - \mathcal{D}_{v_i}^\star(q)}{\mathcal{D}_{v_i}^\star(\star)}$ for $q \in \Omega_{v_i}$
10 Sample $c_2 \sim \mathcal{D}_{v_i}'$ using r_2 and update $\sigma_{\mathsf{out}}(v_i) \leftarrow c_2$
11 return σ_{out}

We say $\sigma_1 \in \sigma_2$ for some $\sigma_1, \sigma_2 \in \Lambda^\star$ if $\sigma_2(v) \in \{\sigma_1(v), \star\}$ for all $v \in V$.

Fact 4.24. *When $\sigma_1, \sigma_2 \in \Lambda$, $\sigma_1 \in \sigma_2$ iff $\sigma_1 = \sigma_2$.*

Now we verify the following properties of Algorithm 8.

Proposition 4.25. *If $\mathrm{e}\alpha\Delta \leq 1$ then the following holds for $g_i \colon \Lambda^\star \times [0,1] \to \Lambda^\star$ from Algorithm 8.*

(1) All the distributions in Algorithm 8 are well-defined.

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

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

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

Proof. First we prove Item (1). Note that $\mathcal{D}_{v_i}^\star$ is the same one as in $\texttt{SafeSampling}(\Phi, \mathcal{M}, v_i, \cdot)$. Thus it is well-defined by Item (1) of Proposition 3.4. Now we assume $\mathsf{Token} = \mathsf{True}$ to check $\mathcal{D}_{v_i}^\dagger$ and $\mathcal{D}_{v_i}'$. By Item (2) of Proposition 3.3, $\mathcal{D}_{v_i}^\dagger$ is well-defined since $\mu_{\mathsf{True}}^{\mathcal{C}'}$ is well-defined. By Item (3) of Proposition 3.4, $\mathcal{D}_{v_i}^\star(q) \leq \mathcal{D}_{v_i}^\dagger(q)$ for any $q \in \Omega_{v_i}$. Hence $\mathcal{D}_{v_i}'$ is also well-defined.

For Item (2), we have the following cases, each of which satisfies $g_i(\sigma_1, r) \in g_i(\sigma_2, r)$.

- $\boxed{c_1 \neq \star.}$ Then both $\sigma_1(v_i)$ and $\sigma_2(v_i)$ are updated to c_1.

- $\boxed{c_1 = \star \text{ and Token} = \text{False for } \sigma_2.}$ Then $\sigma_2(v_i)$ is updated to $\star$.

- $\boxed{c_1 = \star \text{ and Token} = \text{True for } \sigma_2.}$ Since $\sigma_1 \in \sigma_2$, Token also equals True for σ_1. Moreover they get the same CSP from Line 6. Thus $\sigma_1(v_i)$ and $\sigma_2(v_i)$ are updated to the same value c_2.

Item (3) is obviously true if $v_i \in V \setminus \mathcal{M}$ thus we assume $v_i \in \mathcal{M}$. Observe that $\mathcal{D}_{v_i}^{\dagger}(q) = \mathcal{D}_{v_i}'(q) \cdot \mathcal{D}_{v_i}^{\star}(\star) + \mathcal{D}_{v_i}^{\star}(q)$ for any $q \in \Omega_{v_i}$. Then Item (3) follows from Item (4) of Proposition 4.21 with $L = R = i$.

Finally we prove Item (4). Assume $v_i \in \mathcal{M}$ since otherwise it is trivial. To obtain the pseudocode in Algorithm 5, we reorganize Algorithm 8 by first set $\sigma_{\text{out}} \leftarrow \star$ and call $\texttt{Component}(\Phi, \mathcal{M}, \sigma_{\text{out}}, v_i)$, then based on the value of Token we either (A) sample c_1 only then update $\sigma_{\text{out}}(v_i) \leftarrow c_1$, or (B) sample both c_1 and c_2 then update $\sigma_{\text{out}}(v_i) \leftarrow c_1$ if $c_1 \neq \star$; and $\sigma_{\text{out}}(v_i) \leftarrow c_2$ if otherwise. The former is SafeSampling, and the latter, executed jointly, is exactly RejectionSampling as we analyzed for Item (3). □

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

Theorem 4.26 (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 4.17.

Proof of Proposition 4.17. Assume the while iterations in Algorithm 4 stop at $T = T_{\text{Final}}$ or $T_{\text{Final}} = +\infty$ if iterations never end. Since each iteration halts almost surely by Proposition 4.5, T_{Final} is well-defined almost surely.

Let $V = (v_0, \ldots, v_{n-1})$ and define $i_t = t \mod n$ for all $t \in \mathbb{Z}$ as in Algorithm 5. Define random functions $G_{L,R} \colon \Lambda^{\star} \to \Lambda^{\star}$ over random $(r_t)_{t \in \mathbb{Z}}$ for $-\infty < L \leq R < +\infty$ as

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

where $g_t = g_{i_t} \colon \Lambda^{\star} \times [0,1] \to \Lambda^{\star}$ is from Algorithm 8 and $g_t^r(\cdot) = g_t(\cdot, r)$.

Let $M_1 \geq 1$ be the smallest integer such that $G_{-M_1,-1}$ is a constant function on Λ, i.e., $G_{-M_1,-1}(\Lambda) \equiv A$. By Item (3) of Proposition 4.25 and Lemma 4.23, $G_{-M_1',-1}(\Lambda) \equiv A$ for all $M_1' \geq M_1$, and A is distributed as $\mu^{\mathcal{M}}$ if $M_1 < +\infty$ almost surely.

Let $M_2 \geq 1$ be the smallest integer such that $G_{-M_2,-1}(\star^V) \in \Lambda$. Iteratively applying Item (2) of Proposition 4.25, we know $G_{-M_2,-1}(\Lambda) \in G_{-M_2,-1}(\star^V)$. Then by Fact 4.24, $G_{-M_2,-1}(\Lambda)$ is constant. Thus $M_2 \geq M_1$ and $G_{-M_2,-1}(\star^V) = A$.

By Item (4) of Proposition 4.25, $\texttt{BoundingChain}(\Phi, \mathcal{M}, -T, r_{-T}, \ldots, r_{-1})$ equals $G_{-T,-1}(\star^V)$, which means $T_{\text{Final}} \geq M_2$. Thus the final assignment $\widetilde{\sigma}$ equals A and has distribution $\mu^{\mathcal{M}}$ provided $T_{\text{Final}} < +\infty$ almost surely.

Now we only need to show $T_{\text{Final}} < +\infty$ almost surely. Note that either $T_{\text{Final}} = 1$ or, by Algorithm 4, $\texttt{BoundingChain}(\Phi, \mathcal{M}, -T_{\text{Final}}/2, r_{-T_{\text{Final}}/2}, \ldots, r_{-1}) = G_{-T_{\text{Final}}/2,-1}(\star^V) \notin \Lambda$. Thus $T_{\text{Final}} \leq 2 \cdot M_2$, which means it suffices to show $M_2 < +\infty$ almost surely. By Item (3) of Proposition 4.25 and the analysis above, $G_{-i,-1}(\star^V) = A \in \Lambda$ for all $i \geq M_2$; thus

$$\sum_{i=0}^{+\infty} \mathbf{Pr}[M_2 > i] \leq 2n - 1 + \sum_{i=2n-1}^{+\infty} \mathbf{Pr}\left[G_{-i,-1}(\star^V) \notin \Lambda\right]$$

31

$$= 2n - 1 + \sum_{i=2n-1}^{+\infty} \mathbf{Pr}\left[\texttt{BoundingChain}(\Phi, \mathcal{M}, -i, r_{-i}, \ldots, r_{-1}) \notin \Lambda\right]$$

$$\leq 2n - 1 + \sum_{i=2n-1}^{+\infty} 4n \cdot 2^{-\frac{i}{n}} < +\infty, \qquad \text{(by Proposition 4.5)}$$

as desired by Theorem 4.26. $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ $\square$

4.3 The **FinalSampling** Subroutine

Finally we give the missing $\texttt{FinalSampling}(\Phi, \mathcal{M}, \widetilde{\sigma})$ subroutine, which simply completes the assignment on $V \setminus \mathcal{M}$ for $\widetilde{\sigma}$.

Algorithm 9: The FinalSampling subroutine

Input: an atomic CSP $\Phi = \left(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\right)$, a marking $\mathcal{M} \subseteq V$, and an assignment
$\qquad \widetilde{\sigma} \in \left(\prod_{v \in \mathcal{M}} \Omega_v\right) \times \{\star\}^{V \setminus \mathcal{M}}$
Output: an assignment $\sigma \in \sigma_{\texttt{True}}^{\mathcal{C}}$
1 $\Phi_v = \left(V_v, (\Omega_v|_{\widetilde{\sigma}}, \mathcal{D}_v|_{\widetilde{\sigma}})_{v \in V_v}, \mathcal{C}_v\right) \leftarrow \texttt{Component}(\Phi, \mathcal{M}, \widetilde{\sigma}, v)$ for all $v \in V$
$\qquad\qquad\qquad$ /* Ignore the returned Token since it is always True here */
2 Initialize $V' \leftarrow \emptyset$
3 **while** $\exists v \in V \setminus V'$ **do**
4 $\quad$ $\sigma_v \leftarrow \texttt{RejectionSampling}(\Phi_v, r_v)$ $\qquad$ /* r_v is a fresh new randomness tape */
5 $\quad$ Assign $\sigma(V_v) \leftarrow \sigma_v(V_v)$ and update $V' \leftarrow V' \cup V_v$
6 **end**
7 **return** σ

We observe the following results regarding Algorithm 9.

Lemma 4.27. *If $\mathbf{e}\alpha\Delta \leq 1$, then the following holds for $\texttt{FinalSampling}(\Phi, \mathcal{M}, \widetilde{\sigma})$.*

- *It halts almost surely, and outputs $\sigma \sim \mu_{\texttt{True}}^{\mathcal{C}|_{\widetilde{\sigma}}}$ when it halts.*

- *Its expected total running time is at most $O\left(kdQ \sum_{v \in V} (1 - \mathbf{e}\alpha)^{-|\mathcal{C}_v|}\right)$.*

Proof. All the Φ_v can be easily constructed with one pass of $\mathcal{C}$ and V which takes time $O(k|\mathcal{C}| + |V|)$.

By Proposition 3.3, each iteration of Line 4 halts almost surely and generates $\sigma_v \sim \mu_{\texttt{True}}^{\mathcal{C}_v}$ in expected time $O\left((kQ|\mathcal{C}_v| + Q) \cdot (1 - \mathbf{e}\alpha)^{-|\mathcal{C}_v|}\right)$. Therefore FinalSampling halts almost surely and its expected total running time is at most

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

$$= O\left(k|\mathcal{C}| + |V| + \sum_{v \in V} \frac{kQ|\mathcal{C}_v| + Q}{|V_v|} \cdot (1 - \mathbf{e}\alpha)^{-|\mathcal{C}_v|}\right)$$

$$\leq O\left(kdQ \sum_{v \in V} (1 - \mathbf{e}\alpha)^{-|\mathcal{C}_v|}\right). \qquad \text{(since } |\mathcal{C}_v| \leq d|V_v| \text{ and } |\mathcal{C}| \leq d|V|)$$

Moreover when FinalSampling halts, by iteratively applying Item (4) of Proposition 3.1 we have

$$\sigma \sim \prod_{v \text{ needed by Line 4}} \mu^{\mathcal{C}_v}_{\text{True}} = \mu^{\mathcal{C}|_{\widetilde{\sigma}}}_{\text{True}}. \qquad \square$$

Combining Proposition 4.17, we analyze the performance of Algorithm 9 in Algorithm 4.

Proposition 4.28. *If $\mathbf{e}\alpha\Delta \leq 1$, $\mathbf{e}\Delta^2\rho \leq 1/32$, and $\Delta^2\lambda \leq 1/16$, then the following holds for the FinalSampling in Algorithm 4.*

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

Proof. By Item (1) of Lemma 4.27, it halts almost surely. Fix an arbitrary $\sigma' \in \sigma^{\mathcal{C}}_{\text{True}}$. Define $\widetilde{\sigma}$ by setting $\widetilde{\sigma}(\mathcal{M}) = \sigma'(\mathcal{M})$ and $\widetilde{\sigma}(V \setminus \mathcal{M}) = \star^{V \setminus \mathcal{M}}$. Combining Proposition 4.17 and Definition 4.16, we have

$$\mathbf{Pr}\left[\sigma = \sigma'\right] = \mu^{\mathcal{M}}(\widetilde{\sigma}) \cdot \mu^{\mathcal{C}|_{\widetilde{\sigma}}}_{\text{True}}(\sigma') = \mathop{\mathbf{Pr}}_{\widehat{\sigma} \sim \mu^{\mathcal{C}}_{\text{True}}}\left[\widehat{\sigma}(\mathcal{M}) = \sigma'(\mathcal{M})\right] \cdot \mu^{\mathcal{C}|_{\widetilde{\sigma}}}_{\text{True}}(\sigma')$$

$$= \mathop{\mathbf{Pr}}_{\widehat{\sigma} \sim \prod_{v \in V} \mathcal{D}_v}\left[\widehat{\sigma}(\mathcal{M}) = \sigma'(\mathcal{M}) \,\big|\, \widehat{\sigma} \in \sigma^{\mathcal{C}}_{\text{True}}\right] \cdot \mathop{\mathbf{Pr}}_{\sigma'' \sim \prod_{v \in V} \mathcal{D}_v|_{\widetilde{\sigma}}}\left[\sigma'' = \sigma' \,\big|\, \sigma'' \in \sigma^{\mathcal{C}|_{\widetilde{\sigma}}}_{\text{True}}\right]$$

$$= \frac{\mathbf{Pr}_{\widehat{\sigma} \sim \prod_{v \in V} \mathcal{D}_v}\left[\widehat{\sigma}(\mathcal{M}) = \sigma'(\mathcal{M}), \widehat{\sigma} \in \sigma^{\mathcal{C}}_{\text{True}}\right]}{\mathbf{Pr}_{\widehat{\sigma} \sim \prod_{v \in V} \mathcal{D}_v}\left[\widehat{\sigma} \in \sigma^{\mathcal{C}}_{\text{True}}\right]} \cdot \frac{\mathbf{Pr}_{\sigma'' \sim \prod_{v \in V} \mathcal{D}_v|_{\widetilde{\sigma}}}\left[\sigma'' = \sigma'\right]}{\mathbf{Pr}_{\sigma'' \sim \prod_{v \in V} \mathcal{D}_v|_{\widetilde{\sigma}}}\left[\sigma'' \in \sigma^{\mathcal{C}|_{\widetilde{\sigma}}}_{\text{True}}\right]}.$$

By Definition 2.5, $\mathcal{D}_v|_{\widetilde{\sigma}}$ equals $\mathcal{D}_v$ if $v \in V \setminus \mathcal{M}$; and equals the point distribution of $\sigma'(v)$ if $v \in \mathcal{M}$. Let $\sigma_1 \sim \prod_{v \in V \setminus \mathcal{M}} \mathcal{D}_v$ and $\sigma_2 \sim \prod_{v \in \mathcal{M}} \mathcal{D}_v$ be independent. Then

$$\frac{\mathbf{Pr}_{\sigma'' \sim \prod_{v \in V} \mathcal{D}_v|_{\widetilde{\sigma}}}\left[\sigma'' = \sigma'\right]}{\mathbf{Pr}_{\sigma'' \sim \prod_{v \in V} \mathcal{D}_v|_{\widetilde{\sigma}}}\left[\sigma'' \in \sigma^{\mathcal{C}|_{\widetilde{\sigma}}}_{\text{True}}\right]} = \frac{\mathbf{Pr}_{\sigma_1}\left[\sigma_1 = \sigma'(V \setminus \mathcal{M})\right]}{\mathbf{Pr}_{\sigma_1}\left[\sigma_1 \circ \sigma'(\mathcal{M}) \in \sigma^{\mathcal{C}|_{\widetilde{\sigma}}}_{\text{True}}\right]} \quad (\circ \text{ represents vector concatenation})$$

$$= \frac{\mathbf{Pr}_{\sigma_1, \sigma_2}\left[\sigma_1 = \sigma'(V \setminus \mathcal{M}), \sigma_2 = \sigma'(\mathcal{M})\right]}{\mathbf{Pr}_{\sigma_1, \sigma_2}\left[\sigma_1 \circ \sigma'(\mathcal{M}) \in \sigma^{\mathcal{C}|_{\widetilde{\sigma}}}_{\text{True}}, \sigma_2 = \sigma'(\mathcal{M})\right]}$$

$$\text{(since } \sigma_1, \sigma_2 \text{ are independent)}$$

$$= \frac{\mathbf{Pr}_{\sigma_1, \sigma_2}\left[\sigma_1 \circ \sigma_2 = \sigma'\right]}{\mathbf{Pr}_{\sigma_1, \sigma_2}\left[\sigma_1 \circ \sigma_2 \in \sigma^{\mathcal{C}|_{\widetilde{\sigma}}}_{\text{True}}, \sigma_2 = \sigma'(\mathcal{M})\right]}$$

$$= \frac{\mathbf{Pr}_{\widehat{\sigma} \sim \prod_{v \in V} \mathcal{D}_v}\left[\widehat{\sigma} = \sigma'\right]}{\mathbf{Pr}_{\widehat{\sigma} \sim \prod_{v \in V} \mathcal{D}_v}\left[\widehat{\sigma} \in \sigma^{\mathcal{C}|_{\widetilde{\sigma}}}_{\text{True}}, \widehat{\sigma}(\mathcal{M}) = \sigma'(\mathcal{M})\right]}$$

$$= \frac{\mathbf{Pr}_{\widehat{\sigma} \sim \prod_{v \in V} \mathcal{D}_v}\left[\widehat{\sigma} = \sigma'\right]}{\mathbf{Pr}_{\widehat{\sigma} \sim \prod_{v \in V} \mathcal{D}_v}\left[\widehat{\sigma} \in \sigma^{\mathcal{C}}_{\text{True}}, \widehat{\sigma}(\mathcal{M}) = \sigma'(\mathcal{M})\right]}$$

where the last equality is because when $\widehat{\sigma}(\mathcal{M}) = \sigma'(\mathcal{M})$, we have $\widehat{\sigma}(\mathcal{M}) = \widetilde{\sigma}(\mathcal{M})$ and thus $\widehat{\sigma} \in \sigma^{\mathcal{C}|_{\widetilde{\sigma}}}_{\text{True}}$ iff $\widehat{\sigma} \in \sigma^{\mathcal{C}}_{\text{True}}$. Hence in all, we have

$$\mathbf{Pr}\left[\sigma = \sigma'\right] = \frac{\mathbf{Pr}_{\widehat{\sigma} \sim \prod_{v \in V} \mathcal{D}_v}\left[\widehat{\sigma} = \sigma'\right]}{\mathbf{Pr}_{\widehat{\sigma} \sim \prod_{v \in V} \mathcal{D}_v}\left[\widehat{\sigma} \in \sigma^{\mathcal{C}}_{\text{True}}\right]} = \mathop{\mathbf{Pr}}_{\widehat{\sigma} \sim \prod_{v \in V} \mathcal{D}_v}\left[\widehat{\sigma} = \sigma' \,\big|\, \widehat{\sigma} \in \sigma^{\mathcal{C}}_{\text{True}}\right] = \mu^{\mathcal{C}}_{\text{True}}(\sigma')$$

as desired.

Note that $\mu^{\mathcal{M}}$ is a stationary distribution for P_i by Item (3) Proposition 4.21. Meanwhile by Item (4) of Proposition 4.25, the t-th for iteration in BoundingChain is a coupling of P_{i_t}. Thus in FinalSampling, upon receiving $\widetilde{\sigma}$ which has distribution $\mu^{\mathcal{M}}$, we can execute $|V|$ more rounds of Line 3-11 in Algorithm 5 on $\widetilde{\sigma}$ using fresh randomness, and the resulted assignment still has distribution $\mu^{\mathcal{M}}$. In other words, we may safely assume the last $|V|$ rounds of update in the final BoundingChain procedure are all using RejectionSampling. Thus each $|\mathcal{C}_v|$ in Lemma 4.27 also satisfies the concentration bound in Proposition 4.11. By a similar calculation in the proof of the efficiency part of Proposition 4.5 and noticing $|\mathcal{C}| \leq d|V|$, the expected running time here is at most

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

4.4 Putting Everything Together

Now we put everything together to prove our main theorem.

Proof of Theorem 4.1. The correctness part follows immediately from Proposition 4.28 and Proposition 4.17. Thus recall measures defined in Definition 2.1 and we focus on the efficiency part.

- Let X be the total running time of `AtomicCSPSampling`(Φ, π).
- Let A be the time for computing $\beta(\Phi, \mathcal{M})$ in (3). Then $A = O(k|\mathcal{C}|) \leq O(dk|V|)$.
- For integer $i \geq 1$ and $j \in [i]$, let $X_{i,j}$ be the running time of the $(-j)$-th for iteration in `BoundingChain`$(\Phi, \mathcal{M}, -i, r_{-i}, \ldots, r_{-1})$. Then $\mathbb{E}\left[X_{i,j}^2\right] = O\left(dk^2\Delta^5 Q^2\right)$ by Proposition 4.5.
- Let T_{Final} be the T when the while iterations stop. Then by Proposition 4.5, we have $\mathbf{Pr}\left[T_{\mathsf{Final}} \geq t\right] \leq 4|V| \cdot 2^{-t/|V|}$ for $t \geq 2|V| - 1$.
- Let Y be the running time of the FinalSampling in the end. Then by Proposition 4.28 we have $\mathbb{E}[Y] = O\left(d^2 k Q \Delta |V|\right)$.

Therefore we have $X = A + \sum_{t=0}^{\log(T_{\mathsf{Final}})} \sum_{j=1}^{2^t} X_{2^t,j} + Y$[15] and

$$\mathbb{E}[X] = \mathbb{E}[A] + \mathbb{E}[Y] + \sum_{t=0}^{+\infty} \sum_{j=1}^{2^t} \mathbb{E}\left[X_{2^t,j} \cdot [T_{\mathsf{Final}} \geq 2^t]\right]$$

$$\leq \mathbb{E}[A] + \mathbb{E}[Y] + \sum_{t=0}^{+\infty} \sum_{j=1}^{2^t} \sqrt{\mathbb{E}\left[X_{2^t,j}^2\right] \mathbf{Pr}\left[T_{\mathsf{Final}} \geq 2^t\right]} \qquad \text{(by Cauchy-Schwarz inequality)}$$

$$\leq \mathbb{E}[A] + \mathbb{E}[Y] + \sum_{t=0}^{m} \sum_{j=1}^{2^t} \sqrt{\mathbb{E}\left[X_{2^t,j}^2\right]} + \sum_{t=m+1}^{+\infty} \sum_{j=1}^{2^t} \sqrt{\mathbb{E}\left[X_{2^t,j}^2\right] \mathbf{Pr}\left[T_{\mathsf{Final}} \geq 2^t\right]}$$

$$\qquad \qquad (m \geq \lfloor \log(2|V| - 1) \rfloor \text{ to be determined later})$$

$$\leq O\left(d^2 k Q \Delta |V| + \sqrt{dk^2\Delta^5 Q^2} \cdot \left(2^m + \sqrt{|V|} \sum_{t=m+1}^{+\infty} 2^t \cdot 2^{-\frac{2^t}{|V|}}\right)\right).$$

---

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

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

$$2^m + \sqrt{|V|} \sum_{t=m+1}^{+\infty} 2^{t - \frac{2^t}{|V|}} \leq 2^m + \sqrt{|V|} \int_m^{+\infty} 2^{x - \frac{2^x}{|V|}} \, dx \qquad \left(2^{x - \frac{2^x}{n}} \text{ is decreasing when } 2^x \geq \frac{n}{\ln(2)}\right)$$

$$= 2^m + \sqrt{|V|} \cdot \frac{|V|}{\ln^2(2)} \cdot 2^{-\frac{2^m}{|V|}} \qquad \left(\text{since } \left(\frac{-n}{\ln^2(2)} \cdot 2^{-\frac{2^x}{n}}\right)' = 2^{x - \frac{2^x}{n}}\right)$$

$$= O\left(|V| \log(|V|)\right).$$

Since $d \leq \Delta$, we have $\mathbb{E}[X] = O\left(kQ\Delta^3 |V| \log(|V|)\right)$. $\qquad\qquad\square$

Remark 4.29. Computing higher moments of $X_{i,j}, Y$ and using possibly stronger assumption, one can improve the dependency on k, Δ, Q in the expected running time. However we view these as constants compared with $|V|$. Thus we do not make the effort here.

5 Applications

Here we instantiate Theorem 4.1 to special CSPs. We will use the following algorithmic Lovász local lemma for constructing the marking $\mathcal{M}$.

Theorem 5.1 ([MT10]). *Let* $\Phi = \left(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\right)$ *be a CSP. If* $ep\Delta \leq 1$, *then* $\sigma^{\mathcal{C}}_{\mathsf{True}} \neq \emptyset$ *and there exists a randomized algorithm which outputs some* $\sigma \in \sigma^{\mathcal{C}}_{\mathsf{True}}$ *in time* $O(k\Delta|V|)$ *with probability at least* 0.99.

We define the *smooth parameter* of a CSP Φ by

$$\kappa = \kappa(\Phi) = \max_{v \in V} \max_{q, q' \in \Omega_v} \frac{\mathcal{D}_v(q)}{\mathcal{D}_v(q')}. \tag{11}$$

Note that $\kappa(\Phi) \geq 1$ always. When context is clear, we will simply write κ.

5.1 Binary Domains

Let $\mathcal{M} \subseteq V$ be a marking. We specialize α, β, λ when all domains are of size 2.

- $\alpha = \alpha(\Phi, \mathcal{M}) = \max_{C \in \mathcal{C}} \alpha(\Phi, \mathcal{M}, C)$ where

$$\alpha(\Phi, \mathcal{M}, C) = \prod_{v \in \mathsf{vbl}(C) \setminus \mathcal{M}} \mathcal{D}_v(\sigma^C_{\mathsf{False}}(v));$$

- When $e\alpha \leq 1$, define

  - $\beta = \beta(\Phi, \mathcal{M}) = (1 - e\alpha)^{-d} \leq (1 - e\alpha)^{-\Delta}$;
  - $\lambda = \lambda(\Phi, \mathcal{M}) = \max_{C \in \mathcal{C}} \lambda(\Phi, \mathcal{M}, C)$ where

$$\lambda(\Phi, \mathcal{M}, C) = |\mathsf{vbl}(C)|^2 \cdot \beta^{|\mathsf{vbl}(C) \cap \mathcal{M}|} \cdot \prod_{v \in \mathsf{vbl}(C) \cap \mathcal{M}} \mathcal{D}_v(\sigma^C_{\mathsf{False}}(v)).$$

By Remark 4.2, we will use the following version of Theorem 4.1.

Theorem 5.2 (Theorem 4.1, Binary Domains). *Let* $\Phi = \left(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\right)$ *be an atomic CSP such that* $|\Omega_v| = 2$ *for all* $v \in V$. *Let* $\mathcal{M} \subseteq V$ *be a marking. If*

$$\mathbf{e} \cdot \alpha(\Phi, \mathcal{M}, C) \cdot \Delta \leq 1 \quad and \quad \Delta^2 \cdot \lambda(\Phi, \mathcal{M}, C) \leq 1/100 \quad for \ all \ C \in \mathcal{C},$$

then the following holds for `AtomicCSPSampling`$(\Phi, \mathcal{M})$.

- CORRECTNESS. *It halts almost surely and outputs* $\sigma \sim \mu_{\mathsf{True}}^{\mathcal{C}}$ *when it halts.*
- EFFICIENCY. *Its expected total running time is* $O\left(k\Delta^3 |V| \log(|V|)\right)$.

We now construct a valid marking when the underlying distributions are arbitrary.

Lemma 5.3. *Let* $\Phi = \left(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\right)$ *be an atomic CSP such that* $|\Omega_v| = 2$ *for all* $v \in V$. *For any* $\zeta \in (0, 1)$, *if* $p^\gamma \cdot \Delta \leq 0.01 \cdot \zeta/\kappa$ *where*

$$\gamma = \frac{3 - 9\zeta + \ln(\kappa + 1) - \sqrt{\ln^2(\kappa + 1) + 6 \cdot (1 - 3\zeta) \ln(\kappa + 1)}}{9},$$

then there exists a marking $\mathcal{M} \subseteq V$ *such that*

$$\mathbf{e} \cdot \alpha(\Phi, \mathcal{M}, C) \cdot \Delta \leq 1 \quad and \quad \Delta^2 \cdot \lambda(\Phi, \mathcal{M}, C) \leq 1/100 \quad for \ all \ C \in \mathcal{C}.$$

Moreover $\mathcal{M}$ *can be constructed in time* $O(k\Delta|V|)$ *with success probability at least* 0.99.

Proof. For each $C \in \mathcal{C}$, define $p_C = \prod_{v \in \mathsf{vbl}(C)} \mathcal{D}_v(\sigma_{\mathsf{False}}^C(v))$. Then $p = p(\Phi) = \max_{C \in \mathcal{C}} p_C$. Meanwhile by the definition of κ, we know $(1/(\kappa + 1))^{|\mathsf{vbl}(C)|} \leq p_C \leq (\kappa/(\kappa + 1))^{|\mathsf{vbl}(C)|}$. Thus

$$|\mathsf{vbl}(C)| \leq \frac{\ln(1/p_C)}{\ln(1 + 1/\kappa)}. \tag{12}$$

Since $x^\zeta \geq \zeta \ln(x)$ holds for any $x > 0$, we also have

$$\ln(1/p_C) \leq p_C^{-\zeta}/\zeta. \tag{13}$$

Let $\eta, \tau \in (0, 1)$ be parameters and $\mathcal{M}$ be the marking to be determined later. We will ensure $1 - \eta - \tau \geq 0$ and $\eta - \tau - 3\zeta \geq 0$. For each $C \in \mathcal{C}$, let $\mathcal{E}_C$ be the event (i.e., constraint)

$$\mathcal{E}_C = \text{``} \left| \ln \left(\prod_{v \in \mathsf{vbl}(C) \cap \mathcal{M}} \mathcal{D}_v(\sigma_{\mathsf{False}}^C(v)) \right) - \eta \ln(p_C) \right| > \tau \ln(1/p_C) \text{''}.$$

Now we check $\mathbf{e} \cdot \alpha(\Phi, \mathcal{M}, C) \cdot \Delta \leq 1$ and $\Delta^2 \cdot \lambda(\Phi, \mathcal{M}, C) \leq 1/100$ assuming no $\mathcal{E}_C$ happens. Since $\mathsf{vbl}(C)$ is the disjoint union of $\mathsf{vbl}(C) \cap \mathcal{M}$ and $\mathsf{vbl}(C) \setminus \mathcal{M}$, if $\mathcal{E}_C$ does not happen, then

$$\alpha(\Phi, \mathcal{M}, C) \leq p_C^{1 - \eta - \tau} \leq p^{1 - \eta - \tau}.$$

Since $(1 - x)^{-1/x} \leq 4$ holds for any $x \in (0, 1/2]$, we have

$$\beta \leq \left(1 - \mathbf{e} \cdot p^{1 - \eta - \tau}\right)^{-\Delta} \leq 4^{\mathbf{e} \cdot p^{1 - \eta - \tau} \cdot \Delta} \leq \left(\frac{\kappa + 1}{\kappa}\right)^\zeta \quad \text{if} \quad 2\mathbf{e} \ln(2) \cdot p^{1 - \eta - \tau} \cdot \Delta \leq \zeta \cdot \ln(1 + 1/\kappa). \tag{14}$$

Note that $2\mathbf{e}\ln(2) \cdot p^{1-\eta-\tau} \cdot \Delta \leq \zeta \cdot \ln(1+1/\kappa)$ already implies $\mathbf{e} \cdot \alpha(\Phi, \mathcal{M}, C) \cdot \Delta \leq 1$. Combining (12), (13), and (14), we also have

$$\lambda(\Phi, \mathcal{M}, C) \leq |\mathsf{vbl}(C)|^2 \cdot \beta^{|\mathsf{vbl}(C)|} p_C^{\eta-\tau} \leq \frac{\ln^2(1/p_C) \cdot p_C^{\eta-\tau-\zeta}}{\ln^2(1+1/\kappa)} \leq \frac{p_C^{\eta-\tau-3\zeta}}{\zeta^2 \ln^2(1+1/\kappa)} \leq \frac{p^{\eta-\tau-3\zeta}}{\zeta^2 \ln^2(1+1/\kappa)}.$$

Therefore $\Delta^2 \cdot \lambda(\Phi, \mathcal{M}, C) \leq 1/100$ is reduced to $\Delta^2 \cdot p^{\eta-\tau-3\zeta} \leq 0.01 \cdot \zeta^2 \cdot \ln^2(1+1/\kappa)$. In all, it suffices to make sure $1 - \eta - \tau \geq 0$, $\eta - \tau - 3\zeta \geq 0$, and

$$2\mathbf{e}\ln(2) \cdot p^{1-\eta-\tau} \cdot \Delta \leq \zeta \cdot \ln(1+1/\kappa), \quad \text{and} \quad \Delta^2 \cdot p^{\eta-\tau-3\zeta} \leq 0.01 \cdot \zeta^2 \cdot \ln^2(1+1/\kappa). \tag{15}$$

Now we show how to set η, τ and construct $\mathcal{M}$ to make sure no $\mathcal{E}_C$ happens. We put each $v \in V$ into $\mathcal{M}$ independently with probability η. For each $v \in V$, let $x_v \in \{0,1\}$ be the indicator for whether v is in $\mathcal{M}$. Then

$$\mathcal{E}_C = \text{``} \left| \sum_{v \in \mathsf{vbl}(C)} x_v \ln\left(1/\mathcal{D}_v(\sigma_{\mathsf{False}}^C(v))\right) - \mathbb{E}\left[\sum_{v \in \mathsf{vbl}(C)} x_v \ln\left(1/\mathcal{D}_v(\sigma_{\mathsf{False}}^C(v))\right) \right] \right| > \tau \ln(1/p_C) \text{''}.$$

By Hoeffding's inequality [Hoe94, Theorem 2], we have

$$\begin{aligned}
\mathbf{Pr}\left[\mathcal{E}_C\right] &\leq 2\exp\left\{ \frac{-2\tau^2 \ln^2(1/p_C)}{\sum_{v \in \mathsf{vbl}(C)} \ln^2\left(1/\mathcal{D}_v(\sigma_{\mathsf{False}}^C(v))\right)} \right\} \\
&\leq 2\exp\left\{ \frac{-2\tau^2 \ln^2(1/p_C)}{\ln(\kappa+1) \cdot \sum_{v \in \mathsf{vbl}(C)} \ln\left(1/\mathcal{D}_v(\sigma_{\mathsf{False}}^C(v))\right)} \right\} && \text{(by the definition of } \kappa\text{)} \\
&= 2\exp\left\{ \frac{-2\tau^2 \ln(1/p_C)}{\ln(\kappa+1)} \right\} && \text{(since } \ln(p_C) = \sum_{v \in \mathsf{vbl}(C)} \ln\left(\mathcal{D}_v(\sigma_{\mathsf{False}}^C(v))\right)\text{)} \\
&= 2 \cdot p_C^{2\tau^2/\ln(\kappa+1)} \leq 2 \cdot p^{2\tau^2/\ln(\kappa+1)}.
\end{aligned}$$

Since $\mathsf{vbl}(\mathcal{E}_C) = \mathsf{vbl}(C)$ and thus it correlates with Δ many $\mathcal{E}_{C'}$ (including itself), by Theorem 5.1 we can construct $\mathcal{M}$ (i.e., $(x_v)_{v \in V}$) to avoid all $\mathcal{E}_C$ in time $O(k\Delta|V|)$ with probability at least 0.99 as long as

$$2\mathbf{e} \cdot p^{2\tau^2/\ln(\kappa+1)} \cdot \Delta \leq 1.$$

Now we set $\eta = (2 - \tau + 3\zeta)/3$ and

$$\tau = \frac{-\ln(\kappa+1) + \sqrt{\ln^2(\kappa+1) + 6 \cdot (1-3\zeta)\ln(\kappa+1)}}{6} < \frac{1-3\zeta}{2}$$

so that $1 - \eta - \tau$, $(\eta - \tau - 3\zeta)/2$, and $2\tau^2/\ln(\kappa+1)$ all equal

$$\gamma = \frac{1}{3} - \zeta - \frac{-\ln(\kappa+1) + \sqrt{\ln^2(\kappa+1) + 6 \cdot (1-3\zeta)\ln(\kappa+1)}}{9} > 0.$$

Then all the conditions in (15) boil down to $p^\gamma \cdot \Delta \leq 0.1 \cdot \zeta \cdot \ln(1+1/\kappa)$. Since $\kappa \geq 1$ and $\ln(1+x) \geq 0.1x$ for all $x \in [0,1]$, we can safely replace $\ln(1+1/\kappa)$ with $0.1/\kappa$ as desired in the statement. $\qquad\square$

Note that whether $\mathcal{M}$ satisfies the conditions in Theorem 5.2 can be easily checked in time $O(k|\mathcal{C}|) = O(k\Delta|V|)$ by computing $\alpha(\Phi, \mathcal{M}, C)$ and $\lambda(\Phi, \mathcal{M}, C)$ for each $C \in \mathcal{C}$. Thus we can keep performing Lemma 5.3 until the marking $\mathcal{M}$ is valid and then we run `AtomicCSPSampling`$(\Phi, \mathcal{M})$. This provides a Las Vegas algorithm as below.

Corollary 5.4. *There exists a Las Vegas algorithm which takes as input an atomic CSP* $\Phi = \left(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\right)$ *and a parameter* $\zeta \in (0, 1)$ *such that the following holds.*
If $|\Omega_v| = 2$ *holds for all* $v \in V$ *and* $p^\gamma \cdot \Delta \leq 0.01 \cdot \zeta/\kappa$ *where*

$$\gamma = \frac{3 - 9\zeta + \ln(\kappa + 1) - \sqrt{\ln^2(\kappa + 1) + 6 \cdot (1 - 3\zeta)\ln(\kappa + 1)}}{9},$$

then the algorithm outputs a random solution of Φ *distributed perfectly as* $\mu_{\mathsf{True}}^{\mathcal{C}}$ *in expected time* $O\left(k\Delta^3|V|\log(|V|)\right)$.

Remark 5.5. One natural choice of the underlying distributions is the uniform distribution. In this case $\kappa = 1$. By setting $\zeta \to 0$, we have

$$\gamma \to \frac{3 + \ln(2) - \sqrt{\ln^2(2) + 6\ln(2)}}{9} > 0.171.$$

For example we can set $\zeta = 10^{-10}$ and the local lemma condition is simply $p^{0.171} \cdot \Delta \leq 10^{-12}/\kappa$. In Lemma 5.15 we will optimize it to 0.175 by a tighter concentration bound.

5.2 Large Domains: State Tensorization

Here we formally introduce the *state tensorization* technique, generalizing *state compression* from [FHY20]. This, as emphasized in Subsection 1.3, allows us to transform a large domain into a product of binary domains.

Let Ω be a finite domain of size at least 2 and $\mathcal{D}$ be a distribution supported on Ω. A *state tensorization* for $(\Omega, \mathcal{D})$ (See Figure 2 for a concrete example) is a rooted tree $\mathcal{T}$ where

- $\mathcal{T}$ has $|\Omega|$ leaves and each internal node of $\mathcal{T}$ has at least two child nodes;
- the leaves of $\mathcal{T}$ have a one-to-one correspondence with elements in Ω.

For each node $z \in \mathcal{T}$, let $\mathsf{leafs}(z)$ be the set of leaves in the sub-tree of z. Then $\mathsf{leafs}(\mathsf{rt}) = \Omega$ for root rt. For each internal node z, we use $\mathsf{childs}(z)$ to denote its child nodes. For any $z' \in \mathsf{childs}(z)$, we use $z \to z'$ to denote the edge from z to z'. Moreover, we define the weight of $z \to z'$ as

$$W(z \to z') = \frac{\sum_{q \in \mathsf{leafs}(z')} \mathcal{D}(q)}{\sum_{q \in \mathsf{leafs}(z)} \mathcal{D}(q)}. \tag{16}$$

It is easy to see the total weight of outgoing edges of any internal node is 1.

If $|\Omega| = 1$ and thus $\mathcal{D}$ is the point distribution, then the state tensorization $\mathcal{T}$ for $(\Omega, \mathcal{D})$ has two nodes z and z' where z is the root and z' is the only leaf and $W(z \to z') = 1$.

For each $q \in \Omega$, we use $\mathsf{path}(q, \mathcal{T})$ to denote the set of internal nodes in $\mathcal{T}$ on the path from the root to the leaf z that corresponds to q. For example in Figure 2, $\mathsf{path}(b, \mathcal{T}) = \{z_0, z_1, z_3\}$.

We first observe the following fact regarding edge weights in $\mathcal{T}$.

Fact 5.6. *Let* $q \in \Omega$ *be arbitrary. Let* $\mathsf{path}(q, \mathcal{T}) = \{z_0, \ldots, z_\ell\}$ *and* $z_{\ell+1}$ *be the leaf node corresponding to* q. *Assume* $z_0, \ldots, z_{\ell+1}$ *is in the root-to-leaf order. Then* $\mathcal{D}(q) = \prod_{i=0}^{\ell} W(z_i \to z_{i+1})$.

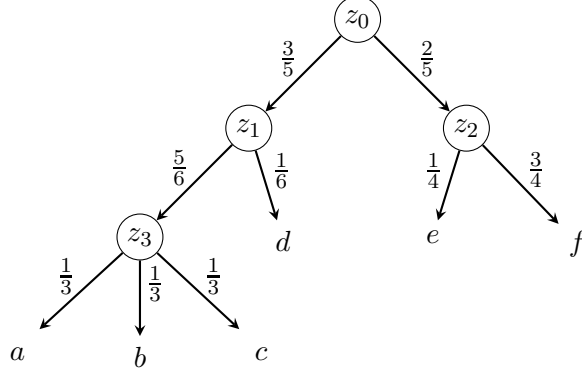State tensorization $\mathcal{T}$ for $(\Omega, \mathcal{D})$

Figure 2: One example of $\mathcal{T}$ for $(\Omega, \mathcal{D})$ where $\Omega = \{a, b, c, d, e, f\}$ and $\mathcal{D}(a) = \mathcal{D}(b) = \mathcal{D}(c) = 1/6, \mathcal{D}(d) = \mathcal{D}(e) = 1/10, \mathcal{D}(f) = 3/10$. We omit the leaf nodes.

Proof. By (16), we have

$$\prod_{i=0}^{\ell} W(z_i \to z_{i+1}) = \prod_{i=0}^{\ell} \frac{\sum_{q' \in \mathsf{leafs}(z_{i+1})} \mathcal{D}(q')}{\sum_{q' \in \mathsf{leafs}(z_i)} \mathcal{D}(q')} = \frac{\sum_{q' \in \mathsf{leafs}(z_{\ell+1})} \mathcal{D}(q')}{\sum_{q' \in \mathsf{leafs}(z_0)} \mathcal{D}(q')} = \mathcal{D}(q) \qquad \square$$

Now we move to atomic CSPs and show formally how state tensorization helps reduce domain sizes.

Definition 5.7 (Tensorized Atomic Constraint Satisfaction Problem)**.** Let $\Phi = \left(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\right)$ be an atomic CSP. Let $(\mathcal{T}_v)_{v \in V}$ be state tensorizations where each $\mathcal{T}_v$ is a state tensorization for $(\Omega_v, \mathcal{D}_v)$.[16] We construct $\Phi^{\otimes} = \left(Z, (\Omega_z, \mathcal{D}_z)_{z \in Z}, \mathcal{C}^{\otimes}\right)$ as the *tensorized atomic constraint satisfaction problem*:[17]

- Z is the set of internal nodes of all $\mathcal{T}_v$.
- For each $z \in Z$, $\Omega_z = \mathsf{childs}(z)$ and $\mathcal{D}_z$ is a distribution supported on Ω_z by setting $\mathcal{D}_z(z') = W(z \to z')$ for all $z' \in \mathsf{childs}(z)$.
- For each $C \in \mathcal{C}$, we construct $C^{\otimes} \in \mathcal{C}^{\otimes}$ by setting

$$\mathsf{vbl}(C^{\otimes}) = \bigcup_{v \in \mathsf{vbl}(C)} \mathsf{path}(\sigma_{\mathsf{False}}^C(v), \mathcal{T}_v)$$

and $C^{\otimes}(\sigma) = \mathsf{False}$ iff $\sigma(z) = z(\sigma_{\mathsf{False}}^C(v))$ for all $v \in \mathsf{vbl}(C)$ and $z \in \mathsf{path}(\sigma_{\mathsf{False}}^C(v))$ where $z(q) \in \mathsf{childs}(z)$ is the child node of z such that $q \in \mathsf{leafs}(z(q))$.[18]

For example in Figure 2, we have $\Omega_{z_0} = \{z_1, z_2\}$ and $\mathcal{D}_{z_0}(z_1) = 3/5, \mathcal{D}_{z_0}(z_2) = 2/5$. Assume constraint C is false iff d is assigned. Then $C^{\otimes}$ will have $\mathsf{vbl}(C^{\otimes}) = \mathsf{path}(d, \mathcal{T}) = \{z_0, z_1\}$ and, for $\sigma \in \Omega_{z_0} \times \Omega_{z_1} \times \Omega_{z_2} \times \Omega_{z_3}$, $C^{\otimes}(\sigma)$ iff $\sigma(z_0) = z_1$ and $\sigma(z_1)$ equals the leaf node corresponding to d.

Recall measures defined in Definition 2.1. Here we note some basic facts for Definition 5.7.

---

[16]We require each $\mathcal{T}_v$ uses different set of tree nodes. So there will be no confusion when using z without explicitly providing $\mathcal{T}_v \ni z$.

[17]Technically $\Phi^{\otimes}$ depends on $(\mathcal{T}_v)_{v \in V}$ which we omit here for simplicity. In addition we remark $\Phi^{\otimes}$ may have redundant variables that do not appear in any constraint; this is indeed consistent with our definition of CSPs as we never require them to shave redundant variables.

[18]Assume $\mathsf{path}(q, \mathcal{T}_v) = \{z_0, \ldots, z_\ell\}$ where $z_0, \ldots, z_\ell$ is in the top-down order of $\mathcal{T}_v$. Let $z_{\ell+1}$ be the leaf node corresponding to q. Then $z_i(q) = z_{i+1}$ for each $i \in \{0, \ldots, \ell\}$.

Fact 5.8. *The following holds for $\Phi^\otimes$.*

(1) $\Phi^\otimes$ is an atomic CSP where $|\mathcal{C}^\otimes| = |\mathcal{C}|$ and $|Z| = \sum_{v \in V} |\{internal\ nodes\ in\ \mathcal{T}_v\}| \le Q(\Phi)|V|$.

(2) $Q(\Phi^\otimes) = \max_{z \in Z} |\mathsf{childs}(z)|$ and $k(\Phi^\otimes) \le k(\Phi) \cdot \max_{C \in \mathcal{C}, v \in \mathsf{vbl}(C)} |\mathsf{path}(\sigma^C_{\mathsf{False}}(v), \mathcal{T}_v)|$

(3) $\Delta(\Phi^\otimes) = \Delta(\Phi)$, $d(\Phi^\otimes) = d(\Phi)$, and $p(\Phi^\otimes) = p(\Phi)$. Moreover for all $C \in \mathcal{C}$ we have

$$\mathop{\mathbf{Pr}}_{\sigma \sim \prod_{v \in \mathsf{vbl}(C)} \mathcal{D}_v} [C(\sigma) = \mathsf{False}] = \mathop{\mathbf{Pr}}_{\sigma' \sim \prod_{z \in \mathsf{vbl}(C^\otimes)} \mathcal{D}_z} [C^\otimes(\sigma') = \mathsf{False}].$$

Proof. Item (1) (2) are obvious. Now we focus on Item (3). Note that for any $C \in \mathcal{C}$ and $v \in V$, $v \in \mathsf{vbl}(C)$ iff the root of $\mathcal{T}_v$ is in $\mathsf{vbl}(C^\otimes)$. On the other hand if some internal node z is in $\mathsf{vbl}(C^\otimes)$ then all the ancestors of z are also in $\mathsf{vbl}(C^\otimes)$. Thus $\Delta(\Phi^\otimes) = \Delta(\Phi)$ and $d(\Phi^\otimes) = d(\Phi)$. The "moreover" part and $p(\Phi^\otimes) = p(\Phi)$ follow from Fact 5.6. $\qquad\square$

Now we formally describe how to translate an assignment for $\Phi^\otimes$ into an assignment for Φ. For any $\sigma \in \prod_{z \in Z} \Omega_z$, we define $\sigma^{\mathsf{Trans}} \in \prod_{v \in V} \Omega_v$ by the following process for each $v \in V$:

- We start from the root of $\mathcal{T}_v$.
- If we are at an internal node z of $\mathcal{T}_v$, then proceed to its child node $\sigma(z)$ and repeat.
- Otherwise we are at a leaf node, then set $\sigma^{\mathsf{Trans}}(v)$ by its corresponding value in Ω_v.

Finally we prove the following simple but powerful reduction result.

Proposition 5.9. *If $\sigma \sim \mu^{\mathcal{C}^\otimes}_{\mathsf{True}}$, then $\sigma^{\mathsf{Trans}} \sim \mu^{\mathcal{C}}_{\mathsf{True}}$.*

Therefore to obtain a random solution of Φ distributed perfectly as $\mu^{\mathcal{C}}_{\mathsf{True}}$, it suffices to have a random solution of $\Phi^\otimes$ distributed perfectly as $\mu^{\mathcal{C}^\otimes}_{\mathsf{True}}$ and then perform Trans operation.

Proof. Recall `RejectionSampling`$(\Phi^\otimes, \cdot)$ in Algorithm 2:

(1) Sample $\sigma \sim \prod_{z \in Z} \mathcal{D}_z$.

(2) If $C^\otimes(\sigma) = \mathsf{True}$ for all $C^\otimes \in \mathcal{C}^\otimes$, then accept σ; otherwise resample σ.

Obviously $\sigma \sim \mu^{\mathcal{C}^\otimes}_{\mathsf{True}}$. By the definition of $C^\otimes$ in Definition 5.7 and the definition of Trans above, we have $C^\otimes(\sigma) = \mathsf{True}$ iff $C(\sigma^{\mathsf{Trans}}) = \mathsf{True}$. Thus we can safely replace Step (2) by

(2a) If $C(\sigma^{\mathsf{Trans}}) = \mathsf{True}$ for all $C \in \mathcal{C}$, then accept σ; otherwise resample σ.

On the other hand for Step (1), we can first sample the σ^{Trans} part and then complete it to σ:

(1a) For each $v \in V$, we start from the root of $\mathcal{T}_v$.

  – If we are at an internal node z of $\mathcal{T}_v$, then sample $\sigma(z) \sim \mathcal{D}_z$ and proceed to $\sigma(z)$.
  – Otherwise we are at a leaf node, then move to the next variable in V.

(1b) For each $z \in Z$ that $\sigma(z)$ is not sampled in Step (1a), we complete it by $\sigma(z) \sim \mathcal{D}_z$.

By Fact 5.6 and the definition of Trans, Step (1a) is equivalent to sample $\sigma^{\mathsf{Trans}} \sim \prod_{v \in V} \mathcal{D}_v$ and σ^{Trans} does not depend on the values sampled in Step (1b). More intuitively, we express this hybrid argument on rejection sampling through the following equivalence of flow charts:

$$\text{Step (1)} \leftrightarrows \text{Step (2)} \to \sigma^{\mathsf{Trans}}$$
$$\text{equals} \quad \text{Step (1a) (1b)} \leftrightarrows \text{Step (2a)} \to \sigma^{\mathsf{Trans}}$$
$$\text{equals} \quad \text{Step (1a)} \leftrightarrows \text{Step (2a)} \to \text{Step (1b)} \to \sigma^{\mathsf{Trans}}$$
$$\text{equals} \quad \text{Step (1a)} \leftrightarrows \text{Step (2a)} \to \sigma^{\mathsf{Trans}},$$

where the last line is exactly `RejectionSampling`$(\Phi, \cdot)$. Thus $\sigma^{\mathsf{Trans}} \sim \mu^{\mathcal{C}}_{\mathsf{True}}$. $\qquad\square$

5.3 General Atomic Constraint Satisfaction Problem: Arbitrary Distribution

Now we deal with general atomic CSPs where domains may be large. Firstly we prove the following lemma which describes a balanced way to construct state tensorization.

Lemma 5.10. *There exists a deterministic algorithm such that the following holds. Let Ω be a finite domain of size at least 2 and $\mathcal{D}$ be a distribution supported on Ω. Let $\kappa = \max_{q,q' \in \Omega} \mathcal{D}(q)/\mathcal{D}(q')$. The algorithm constructs a state tensorization $\mathcal{T}$ for $(\Omega, \mathcal{D})$ where*

(1) the algorithm runs in time $O(|\Omega| \log(|\Omega|))$ and $\mathcal{T}$ is a binary tree;

(2) for any internal node $z \in \mathcal{T}$ and $\{z_1, z_2\} = \mathsf{childs}(z)$, we have $\frac{W(z \to z_1)}{W(z \to z_2)} \leq \max\{\kappa, 2\}$.

Proof. $\mathcal{T}$ is constructed like a Huffman tree as follows:

- For each $q \in \Omega$ create a node z_q with value $\mathsf{val}(z_q) = \mathcal{D}(q)$. Initialize set $S = \{z_q \mid q \in \Omega\}$.
- While $|S| \geq 2$,
  - select two distinct nodes $z_1, z_2 \in S$ with minimum value, i.e., $\mathsf{val}(z_1), \mathsf{val}(z_2)$ are the smallest among all nodes in S,
  - create a parent node z of z_1 and z_2; then set $\mathsf{val}(z) = \mathsf{val}(z_1) + \mathsf{val}(z_2)$ and update $S \leftarrow (S \cup \{z\}) \setminus \{z_1, z_2\}$. Note that $W(z \to z_1) = \frac{\mathsf{val}(z_1)}{\mathsf{val}(z)}$ and $W(z \to z_2) = \frac{\mathsf{val}(z_2)}{\mathsf{val}(z)}$.
- The final node in S when $|S| = 1$ is the root of $\mathcal{T}$.

Then Item (1) is obvious if S is implemented as a balanced binary search tree or a heap.

Now we turn to Item (2). Define $\kappa(S) = \max_{z,z' \in S} \mathsf{val}(z)/\mathsf{val}(z')$ when $|S| \geq 2$. Then each time we select $z_1, z_2 \in S$ and link them to z, we have $\frac{W(z \to z_1)}{W(z \to z_2)} \leq \kappa(S)$. Therefore it suffices to show $\kappa(S) \leq \max\{\kappa, 2\}$ throughout the construction.

- $\boxed{\text{Initialization.}}$ Then we simply have $\kappa(S) = \kappa$.
- $\boxed{\text{Afterwards.}}$ Assume $S = \{z_1, z_2, \ldots, z_\ell\}$ for $\ell \geq 2$ where $\mathsf{val}(z_1) \leq \mathsf{val}(z_2) \leq \cdots \leq \mathsf{val}(z_\ell)$. Let $S' = \{z, z_3, \ldots, z_\ell\}$ be S after the update where $\mathsf{val}(z) = \mathsf{val}(z_1) + \mathsf{val}(z_2)$. Now we have two possible cases.

  - If $\mathsf{val}(z) \leq \mathsf{val}(z_\ell)$, then
  $$\kappa(S') = \max\left\{\frac{\mathsf{val}(z_\ell)}{\mathsf{val}(z)}, \frac{\mathsf{val}(z_\ell)}{\mathsf{val}(z_3)}\right\} \leq \frac{\mathsf{val}(z_\ell)}{\mathsf{val}(z_1)} = \kappa(S) \leq \max\{\kappa, 2\}.$$

  - If $\mathsf{val}(z) > \mathsf{val}(z_\ell)$, then
  $$\kappa(S') = \frac{\mathsf{val}(z)}{\mathsf{val}(z_3)} = \frac{\mathsf{val}(z_1) + \mathsf{val}(z_2)}{\mathsf{val}(z_3)} \leq 2. \qquad \square$$

By Lemma 5.10, Proposition 5.9, and Corollary 5.4, we have the following theorem.

Theorem 5.11. *There exists a Las Vegas algorithm which takes as input an atomic CSP $\Phi = \left(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\right)$ and a parameter $\zeta \in (0, 1)$ such that the following holds.*

Let $\widetilde{\kappa} = \max\{\kappa, 2\}$ (Recall $\kappa = \kappa(\Phi)$ from (11)). If $p^\gamma \cdot \Delta \leq 0.01 \cdot \zeta/\widetilde{\kappa}$ where

$$\gamma = \frac{3 - 9\zeta + \ln(\widetilde{\kappa} + 1) - \sqrt{\ln^2(\widetilde{\kappa} + 1) + 6 \cdot (1 - 3\zeta) \ln(\widetilde{\kappa} + 1)}}{9},$$

then the algorithm outputs a random solution of Φ distributed perfectly as $\mu_{\mathsf{True}}^{\mathcal{C}}$ in expected time $O\left(k\Delta^3 Q^2 |V| \log(Q|V|)\right)$.

Proof. By fixing the variable which has domain size 1, we may safely assume $|\Omega_v| \geq 2$ for all $v \in V$.

We use Lemma 5.10 to construct $\mathcal{T}_v$ for each $(\Omega_v, \mathcal{D}_v)$. This takes time $O(Q(\Phi) \log(Q(\Phi))) \cdot |V|$. Then let $\Phi^\otimes = \left(Z, (\Omega_z, \mathcal{D}_z)_{z \in Z}, \mathcal{C}^\otimes\right)$ be the tensorized atomic CSP (Defined in Definition 5.7). By Item (2) of Lemma 5.10, we have $\kappa(\Phi^\otimes) \leq \max\{\kappa(\Phi), 2\}$. Also by Fact 5.8, we have $\Delta(\Phi^\otimes) = \Delta(\Phi)$, $p(\Phi^\otimes) = p(\Phi)$, $k(\Phi^\otimes) \leq k(\Phi) \cdot Q(\Phi)$[19], and $|Z| \leq Q(\Phi)|V|$. By Proposition 5.9, it suffices to obtain a random solution of $\Phi^\otimes$ distributed perfectly as $\mu^{\mathcal{C}^\otimes}_{\mathsf{True}}$, which, by Corollary 5.4, gives the claimed bounds. $\square$

Remark 5.12. Applying Theorem 5.11 to the uniform distributions, i.e., $\kappa = 1$, and setting $\zeta \to 0$, we have

$$\gamma \to \frac{3 + \ln(3) - \sqrt{\ln^2(3) + 6\ln(3)}}{9} > 0.145.$$

This already beats $1/7$ from [JPV20] and 0.142 from [JPV21]. In Corollary 5.18 we will further optimize it to 0.175 with a more refined construction.

5.4 Hypergraph Coloring

The previous bounds can be improved if the domains are large enough and the underlying distributions are smooth. Here we take the hypergraph coloring problem as an example.

Definition 5.13 (Hypergraph Coloring). Let Q and k be positive integers. Let $H = (V, E)$ be a k-uniform hypergraph, i.e., each edge $e \in E$ contains exactly k distinct variables. We associate it with an atomic CSP $\Phi = \Phi(H, Q) = \left(V, ([Q], \mathcal{U})^V, \mathcal{C}\right)$ where $\mathcal{U}$ is the uniform distribution over $[Q]$ and $\mathcal{C} = \left\{C_{e,i} \colon [Q]^V \to \{\mathsf{True}, \mathsf{False}\} \,\middle|\, e \in E, i \in [Q]\right\}$ and $C_{e,i}(\sigma) = \mathsf{False}$ iff $\sigma(v) = i$ for all $v \in e$. A solution to Φ is called a *proper coloring* for H.

To avoid confusion, k, d, Δ will only be referred to $k(H), d(H), \Delta(H)$. It is easy to see $k(\Phi) = k$, $d(\Phi) = Q \cdot d$, $\Delta(\Phi) = Q \cdot \Delta$, $Q(\Phi) = Q$, and $p(\Phi) = Q^{-k}$.

Theorem 5.14. *There exists a Las Vegas algorithm which takes as input a k-uniform hypergraph $H = (V, E)$ and an integer Q. If $\Delta = \Delta(H) \leq Q^{(1/3 - o_{Q,k}(1))k}$, then the algorithm outputs a perfect uniform random proper coloring for H in expected time $O\left(k\Delta^3 Q^4 \log(Q)|V| \log(Q|V|)\right)$.*

Proof. For each $v \in V$, we construct the state tensorization $\mathcal{T}_v$ for $(\Omega_v, \mathcal{D}_v) = ([Q], \mathcal{U})$ as a complete binary tree, i.e., $\mathcal{T}_v$ has depth $D = \lceil \log(Q) \rceil$ and $\mathcal{T}_v$ has 2^i nodes at level i for all $0 \leq i < \lceil \log(Q) \rceil$.

Given the state tensorizations, we obtain the tensorized atomic CSP $\Phi^\otimes = \left(Z, (\Omega_z, \mathcal{D}_z)_{z \in Z}, \mathcal{C}^\otimes\right)$ for $\Phi = \Phi(H, Q)$. By Definition 5.13 and Proposition 5.9 it suffices to obtain $\sigma \sim \mu^{\mathcal{C}^\otimes}_{\mathsf{True}}$.

Define $R = \lfloor \frac{2}{3} \log(Q) \rfloor$. We construct the marking $\mathcal{M}$ for $\Phi^\otimes$ by putting all internal nodes in $\mathcal{T}_v$ of level at least R into $\mathcal{M}$ for each $v \in V$. To apply Theorem 5.2 to $(\Phi^\otimes, \mathcal{M})$, we compute the constants α, β, λ (See in Subsection 5.1) for $(\Phi^\otimes, \mathcal{M})$.

Fix any $C^\otimes \in \mathcal{C}^\otimes$ and $v \in \mathsf{vbl}(C)$. Assume $\mathsf{path}(\sigma^C_{\mathsf{False}}(v), \mathcal{T}_v) = \{z_0, \ldots, z_\ell\}$ where $z_0, \ldots, z_\ell$ is in the top-down order of $\mathcal{T}_v$. Then $\ell \in \{D-2, D-1\}$ and $z_R, \ldots, z_\ell \in \mathcal{M}$. Since $\mathcal{T}_v$ is a complete binary tree, by (16) and the definition of $\mathcal{D}_z$ in Definition 5.7, if $\ell \geq R$ then we have

$$\prod_{i=R}^{\ell} \mathcal{D}_{z_i}(\sigma^{C^\otimes}_{\mathsf{False}}(z_i)) = \frac{1}{|\mathsf{leafs}(z_R)|} \leq 2^{-(\ell - R)} \leq 2^{-(D-2-R)} \leq 4/Q^{1/3}.$$

---

[19]It is possible to get a better bound on $k(\Phi^\otimes)$ (for example $k(\Phi^\otimes) \leq k(\Phi) \cdot \frac{\ln(\kappa(\Phi)Q(\Phi)+1)}{\ln(1+1/\kappa(\Phi))}$) by analyzing the depth of each $\mathcal{T}_v$. This is because the construction of $\mathcal{T}_v$ is "balanced" and the support of $\mathcal{D}_v$ has size $|\Omega_v| \leq Q(\Phi)$ only. However this only slightly improves the bound on the running time which is not our main focus.

Thus
$$\alpha(\Phi^{\otimes}, \mathcal{M}, C^{\otimes}) \leq \left(4/Q^{1/3}\right)^k. \tag{17}$$

By Item (3) of Fact 5.8, $\Delta(\Phi^{\otimes}) = \Delta(\Phi) = Q \cdot \Delta$. Since $(1-x)^{-1/x} \leq 4$ holds for any $x \in (0, 1/2]$, we have
$$\beta \leq 4^{\mathbf{e} \cdot \left(4/Q^{1/3}\right)^k \cdot Q\Delta} \leq 4^{\frac{1}{kD}} \quad \text{if} \quad \mathbf{e} \cdot \left(4/Q^{1/3}\right)^k \cdot Q\Delta \leq \frac{1}{kD}. \tag{18}$$

Meanwhile
$$\prod_{i=0}^{R-1} \mathcal{D}_{z_i}(\sigma_{\mathsf{False}}^{C^{\otimes}}(z_i)) = \frac{|\mathsf{leafs}(z_R)|}{Q} \leq \frac{2^{\ell-R+2}}{Q} \leq \frac{2^{D-R+1}}{Q} \leq 8/Q^{2/3}.$$

By Item (2) of Fact 5.8, $k(\Phi^{\otimes}) \leq k(\Phi) \cdot D = kD$. Thus combining (18), we have
$$\lambda(\Phi^{\otimes}, \mathcal{M}, C^{\otimes}) \leq k^2 D^2 \cdot 4 \cdot \left(8/Q^{2/3}\right)^k. \tag{19}$$

Therefore, it suffices to make sure
$$D - 2 \geq R, \quad \mathbf{e} \cdot \left(4/Q^{1/3}\right)^k \cdot Q\Delta \leq \frac{1}{kD}, \quad \text{and} \quad k^2 D^2 \cdot 4 \cdot \left(8/Q^{2/3}\right)^k \cdot (Q\Delta)^2 \leq 1/100.$$

Since $D = \lceil \log(Q) \rceil$ and $R = \lfloor \frac{2}{3} \log(Q) \rfloor$, it suffices to ensure
$$Q \geq 5 \quad \text{and} \quad \Delta \leq \left(\frac{Q^{1/3}}{4}\right)^k \cdot \frac{1}{40kQ\log(Q)} = Q^{(1/3 - o_{Q,k}(1))k}.$$

Now we compute the running time. Note that the reduction to $\Phi^{\otimes}$ and the construction of $\mathcal{M}$ only take $O(Q|V|)$ time. Since $|Z| \leq Q|V|$, $k(\Phi^{\otimes}) = O(k\log(Q))$, and $\Delta(\Phi^{\otimes}) = Q\Delta$, by Theorem 5.2 the algorithm runs in time $O\left(k\Delta^3 Q^4 \log(Q)|V|\log(Q|V|)\right)$. $\qquad\square$

5.5 General Atomic Constraint Satisfaction Problem: Uniform Distribution

The analysis in Subsection 5.3 is purely a black-box reduction from large domains to binary domains: the construction of the marking does not use the fact that the CSP after reduction is actually a tensorized atomic CSP. Here we provide a unified construction for the state tensorizations and the marking when the underlying distributions are the uniform distribution.

Let $\Phi = \left(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\right)$ be an atomic CSP such that $|\Omega_v| \geq 2$ and $\mathcal{D}_v$ is the uniform distribution for all $v \in V$. Our strategy is similar as before:

(1) Construct state tensorizations $(\mathcal{T}_v)_{v \in V}$ to obtain $\Phi^{\otimes} = \left(Z, (\Omega_z, \mathcal{D}_z)_{z \in Z}, \mathcal{C}^{\otimes}\right)$ as the tensorized atomic CSP for Φ. We will make sure each Ω_z is a binary domain, i.e., $Q(\Phi^{\otimes}) = 2$, to apply Theorem 5.2.

(2) Construct a marking $\mathcal{M} \subseteq Z$ to satisfy
$$\mathbf{e} \cdot \alpha(\Phi^{\otimes}, \mathcal{M}, C^{\otimes}) \cdot \Delta(\Phi^{\otimes}) \leq 1 \quad \text{and} \quad \Delta(\Phi^{\otimes})^2 \cdot \lambda(\Phi^{\otimes}, \mathcal{M}, C^{\otimes}) \leq 1/100 \quad \text{for all } C^{\otimes} \in \mathcal{C}^{\otimes},$$
where we recall α, β, λ from Subsection 5.1.

(3) Apply Theorem 5.2 to $\Phi^{\otimes}$ and $\mathcal{M}$. By Proposition 5.9, this provides a perfect uniform solution to Φ after performing Trans operation.

For each $C \in \mathcal{C}$, define $p_C = \prod_{v \in \mathsf{vbl}(C)} \mathcal{D}_v(\sigma_{\mathsf{False}}^C(v)) = \prod_{v \in \mathsf{vbl}(C)} |\Omega_v|^{-1}$. For each $C^\otimes \in \mathcal{C}^\otimes$, define $p_{C^\otimes} = \prod_{z \in \mathsf{vbl}(C^\otimes)} \mathcal{D}_z(\sigma_{\mathsf{False}}^{C^\otimes}(z))$. To avoid confusion and by Item (3) of Fact 5.8, we will use Δ for both $\Delta(\Phi^\otimes)$ and $\Delta(\Phi)$; use p for both $p(\Phi^\otimes)$ and $p(\Phi)$, and use p_C for both p_C and $p_{C^\otimes}$.

We construct the state tensorization $\mathcal{T}_v$ for each $(\Omega_v, \mathcal{D}_v)$ in a random and independent way. The marking $\mathcal{M} \subseteq Z$ will be constructed along with each $\mathcal{T}_v$. This will be similar as Lemma 5.3: $(\mathcal{T}_v)_{v \in V}$ and $\mathcal{M}$ are constructed with high success probability using Theorem 5.1.

Recall $\mathsf{path}(\cdot, \cdot)$ from Subsection 5.2 and $z(\cdot)$ from Definition 5.7. For each $v \in V$ and $q \in \Omega_v$, define

$$X(v, q) = \log \left(\prod_{z \in \mathsf{path}(q, \mathcal{T}_v) \cap \mathcal{M}} \mathcal{D}_z(z(q)) \right).$$

By Fact 5.6 and noticing $\mathcal{D}_v$ is uniform, we know

$$X(v, q) = \log \left(|\Omega_v|^{-1} \right) - \log \left(\prod_{z \in \mathsf{path}(q, \mathcal{T}_v) \setminus \mathcal{M}} \mathcal{D}_z(z(q)) \right).$$

Recall the definition of $C^\otimes$ from Definition 5.7, then we have

$$\log \left(\alpha(\Phi^\otimes, \mathcal{M}, C^\otimes) \right) = \log(p_C) - \sum_{v \in \mathsf{vbl}(C)} X(v, \sigma_{\mathsf{False}}^C(v)) \tag{20}$$

and $\beta = \beta(\Phi^\otimes, \mathcal{M}) \leq (1 - \mathbf{e} \cdot \max_{C^\otimes \in \mathcal{C}^\otimes} \alpha(\Phi^\otimes, \mathcal{M}, C^\otimes))^{-\Delta}$. Then we also have

$$\log \left(\lambda(\Phi^\otimes, \mathcal{M}, C^\otimes) \right) \leq \log \left(|\mathsf{vbl}(C^\otimes)|^2 \cdot \beta^{|\mathsf{vbl}(C^\otimes)|} \right) + \sum_{v \in \mathsf{vbl}(C)} X(v, \sigma_{\mathsf{False}}^C(v)).$$

Let $\eta, \tau_1, \tau_2, \zeta \in (0, 1)$ be parameters to be determined later. We will ensure $1 - \eta - \tau_1 \geq 0$ and $\eta - \tau_2 - 3\zeta \geq 0$. For each $C \in \mathcal{C}$, let $\mathcal{E}_C^{(1)}$ be the event (i.e., constraint)

$$\mathcal{E}_C^{(1)} = \text{``} \left(\sum_{v \in \mathsf{vbl}(C)} X(v, \sigma_{\mathsf{False}}^C(v)) \right) - \eta \log(p_C) < -\tau_1 \log(1/p_C) \text{''}$$

and let $\mathcal{E}_C^{(2)}$ be

$$\mathcal{E}_C^{(2)} = \text{``} \left(\sum_{v \in \mathsf{vbl}(C)} X(v, \sigma_{\mathsf{False}}^C(v)) \right) - \eta \log(p_C) > \tau_2 \log(1/p_C) \text{''}.$$

Now we check $\mathbf{e} \cdot \alpha(\Phi^\otimes, \mathcal{M}, C^\otimes) \cdot \Delta \leq 1$ and $\Delta^2 \cdot \lambda(\Phi^\otimes, \mathcal{M}, C^\otimes) \leq 1/100$ assuming no $\mathcal{E}_C^{(1)}$ or $\mathcal{E}_C^{(2)}$ happens. Then firstly $\alpha(\Phi^\otimes, \mathcal{M}, C^\otimes) \leq p_C^{1-\eta-\tau_1} \leq p^{1-\eta-\tau_1}$. Since $(1-x)^{-1/x} \leq 4$ holds for any $x \in (0, 1/2]$, we have

$$\beta \leq 4^{\mathbf{e} \cdot p^{1-\eta-\tau_1} \cdot \Delta} \leq (3/2)^\zeta \quad \text{if} \quad 2\mathbf{e} \ln(2) \cdot p^{1-\eta-\tau_1} \cdot \Delta \leq \zeta \cdot \ln(3/2).$$

Note that $2\mathbf{e} \ln(2) \cdot p^{1-\eta-\tau_1} \cdot \Delta \leq \zeta \cdot \ln(3/2)$ already implies $\mathbf{e} \cdot \alpha(\Phi^\otimes, \mathcal{M}, C^\otimes) \cdot \Delta \leq 1$.

Now we turn to λ. Since $x^\zeta \geq \zeta \ln(x)$ holds for any $x > 0$, we have $\ln(1/p_C) \leq p_C^{-\zeta}/\zeta$. Our state tensorizations will have $\kappa(\Phi^\otimes) \leq 2$ (Recall $\kappa(\cdot)$ from (11)), thus $p_C \leq (2/3)^{|\mathsf{vbl}(C^\otimes)|}$ and

$$\lambda(\Phi^\otimes, \mathcal{M}, C^\otimes) \leq |\mathsf{vbl}(C^\otimes)|^2 \cdot \beta^{|\mathsf{vbl}(C^\otimes)|} p_C^{\eta-\tau_2} \leq \frac{\ln^2(1/p_C) \cdot p_C^{\eta-\tau_2-\zeta}}{\zeta^2 \ln^2(3/2)} \leq \frac{p_C^{\eta-\tau_2-3\zeta}}{\zeta^2 \ln^2(3/2)} \leq \frac{p^{\eta-\tau_2-3\zeta}}{\zeta^2 \ln^2(3/2)}.$$

Therefore $\Delta^2 \cdot \lambda(\Phi^\otimes, \mathcal{M}, C^\otimes) \leq 1/100$ is reduced to $\Delta^2 \cdot p^{\eta - \tau_2 - 3\zeta} \leq 0.01 \cdot \zeta^2 \cdot \ln^2(3/2)$.

In all, it suffices to make sure we can construct the state tensorizations and the marking to avoid all $\mathcal{E}_C^{(1)}$ and $\mathcal{E}_C^{(2)}$ with the following additional conditions:

$$1 - \eta - \tau_1 \geq 0, \quad \eta - \tau_2 - 3\zeta \geq 0,$$
$$2e \ln(2) \cdot p^{1-\eta-\tau_1} \cdot \Delta \leq \zeta \cdot \ln(3/2), \quad \text{and} \quad \Delta^2 \cdot p^{\eta-\tau_2-3\zeta} \leq 0.01 \cdot \zeta^2 \cdot \ln^2(3/2). \tag{21}$$

We first consider the simple case where all domains are binary to verify Remark 5.5.

Lemma 5.15. *Assume $|\Omega_v| = 2$ for all $v \in V$. If $p^{0.175} \cdot \Delta \leq 10^{-7}$, then $\Phi^\otimes$ and $\mathcal{M}$ can be constructed in time $O(k(\Phi) \cdot \Delta |V|)$ with success probability at least 0.99 such that $Q(\Phi^\otimes) = 2$, $\kappa(\Phi^\otimes) = 1$, and*

$$\mathbf{e} \cdot \alpha(\Phi^\otimes, \mathcal{M}, C^\otimes) \cdot \Delta \leq 1 \quad \text{and} \quad \Delta^2 \cdot \lambda(\Phi^\otimes, \mathcal{M}, C^\otimes) \leq 1/100 \quad \text{for all } C^\otimes \in \mathcal{C}^\otimes.$$

Proof. Here we simply let each $\mathcal{T}_v$ have one root with two child nodes. Then $\Phi^\otimes$ is essentially Φ itself. We put each variable into $\mathcal{M}$ with probability η independently. Then for each $v \in V$ and $q \in \Omega_v$, we have

$$X(v, q) = \log\left(\frac{1}{2}\right) \cdot [v \in \mathcal{M}] = \begin{cases} -1 & \text{with probability } \eta, \\ 0 & \text{with probability } 1 - \eta. \end{cases}$$

Hence for any $t \in \mathbb{R}$, we have $\mathbb{E}\left[e^{t \cdot X(v,q)}\right] = 1 - \eta + \eta \cdot e^{-t}$. Note that $p_C = 2^{-|\mathsf{vbl}(C)|}$. Let $t_1, t_2 > 0$ be some parameters to be optimized soon. Then we have

$$\mathbf{Pr}\left[\mathcal{E}_C^{(1)}\right] = \mathbf{Pr}\left[-t_1 \cdot \sum_{v \in \mathsf{vbl}(C)} X(v, \sigma_{\mathsf{False}}^C(v)) > (\eta + \tau_1) \cdot t_1 \cdot |\mathsf{vbl}(C)|\right]$$

$$= \mathbf{Pr}\left[\exp\left\{-t_1 \cdot \sum_{v \in \mathsf{vbl}(C)} X(v, \sigma_{\mathsf{False}}^C(v))\right\} > \exp\left\{(\eta + \tau_1) \cdot t_1 \cdot |\mathsf{vbl}(C)|\right\}\right]$$

$$\leq \mathbb{E}\left[\exp\left\{-t_1 \cdot \sum_{v \in \mathsf{vbl}(C)} X(v, \sigma_{\mathsf{False}}^C(v))\right\}\right] \cdot e^{-(\eta+\tau_1) \cdot t_1 \cdot |\mathsf{vbl}(C)|} \quad \text{(by Markov's inequality)}$$

$$= \left(\left(1 - \eta + \eta \cdot e^{t_1}\right) \cdot e^{-(\eta+\tau_1) \cdot t_1}\right)^{|\mathsf{vbl}(C)|} \quad \text{(since } X(v,q)\text{'s are independent for different } v)$$

$$= \left(\left(\frac{1-\eta}{1-\eta-\tau_1}\right)^{1-\eta-\tau_1}\left(\frac{\eta}{\eta+\tau_1}\right)^{\eta+\tau_1}\right)^{|\mathsf{vbl}(C)|} \quad \left(\text{setting } e^{t_1} = \frac{(1-\eta)(\eta+\tau_1)}{\eta \cdot (1-\eta-\tau_1)}\right)$$

$$= 2^{-|\mathsf{vbl}(C)| \cdot \mathsf{KL}(\eta+\tau_1 \| \eta)} \leq p^{\mathsf{KL}(\eta+\tau_1 \| \eta)}, \quad \text{(since } 2^{-|\mathsf{vbl}(C)|} = p_C \leq p)$$

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*. Similarly

$$\mathbf{Pr}\left[\mathcal{E}_C^{(2)}\right] = \mathbf{Pr}\left[t_2 \cdot \sum_{v \in \mathsf{vbl}(C)} X(v, \sigma_{\mathsf{False}}^C(v)) > -(\eta - \tau_2) \cdot t_2 \cdot |\mathsf{vbl}(C)|\right]$$

$$\leq \left(\left(1 - \eta + \eta \cdot e^{-t_2}\right) \cdot e^{(\eta-\tau_2) \cdot t_2}\right)^{|\mathsf{vbl}(C)|}$$

$$= \left(\left(\frac{1-\eta}{1-\eta+\tau_2} \right)^{1-\eta+\tau_2} \left(\frac{\eta}{\eta-\tau_2} \right)^{\eta-\tau_2} \right)^{|\mathsf{vbl}(C)|} \qquad \left(\text{setting } \mathbf{e}^{-t_2} = \frac{(1-\eta)(\eta-\tau_2)}{\eta \cdot (1-\eta+\tau_2)} \right)$$

$$\leq p^{\mathsf{KL}(\eta-\tau_2\|\eta)}.$$

Let $\mathcal{E}_C = \mathcal{E}_C^{(1)} \vee \mathcal{E}_C^{(2)}$. Since each variable is put into $\mathcal{M}$ independently, $\mathcal{E}_C$ depends only on the constructions over $\mathsf{vbl}(C)$ and thus correlates with at most Δ many $\mathcal{E}_{C'}$ (including itself) and

$$\mathbf{Pr}\left[\mathcal{E}_C\right] \leq 2 \cdot p^{\min\{\mathsf{KL}(\eta+\tau_1\|\eta), \mathsf{KL}(\eta-\tau_2\|\eta)\}}.$$

To apply Theorem 5.1, it suffices to make sure

$$2\mathbf{e} \cdot p^{\min\{\mathsf{KL}(\eta+\tau_1\|\eta), \mathsf{KL}(\eta-\tau_2\|\eta)\}} \cdot \Delta \leq 1. \tag{22}$$

Combining (21) and (22), we can pick $\eta, \tau_1, \tau_2, \zeta$ satisfying $1 - \eta - \tau_1 \geq 0, \eta - \tau_2 - 3\zeta \geq 0$; then let

$$\gamma = \min \left\{ 1 - \eta - \tau_1, \frac{\eta - \tau_2 - 3\zeta}{2}, \mathsf{KL}(\eta+\tau_1\|\eta), \mathsf{KL}(\eta-\tau_2\|\eta) \right\}$$

and all the conditions boil down to $p^\gamma \cdot \Delta \leq 0.01 \cdot \zeta$.

Numerically maximizing γ, we have $\gamma = 0.175$, together with $\eta = 0.595$, $\tau_1 = 0.23$, $\tau_2 = 0.245 - 3 \cdot 10^{-5}$, and $\zeta = 10^{-5}$. □

Now we proceed to the general case. Restricted by the binary case, we cannot hope for a better bound than $p^{0.175} \cdot \Delta \lesssim 1$. Therefore our goal is to show this bound is obtainable using the numerical constants $\eta, \tau_1, \tau_2, \zeta$ determined above.

Lemma 5.16. *Assume $|\Omega_v| \geq 2$ for all $v \in V$. If $p^{0.175} \cdot \Delta \leq 10^{-7}$, then $\Phi^\otimes$ and $\mathcal{M}$ can be constructed in time $O(k(\Phi)Q(\Phi) \cdot \Delta|V|)$ with success probability at least 0.99 such that $Q(\Phi^\otimes) = 2$, $\kappa(\Phi^\otimes) \leq 2$, and*

$$\mathbf{e} \cdot \alpha(\Phi^\otimes, \mathcal{M}, C^\otimes) \cdot \Delta \leq 1 \quad \text{and} \quad \Delta^2 \cdot \lambda(\Phi^\otimes, \mathcal{M}, C^\otimes) \leq 1/100 \quad \text{for all } C^\otimes \in \mathcal{C}^\otimes.$$

Proof. Note that the choice of $\eta, \tau_1, \tau_2, \zeta$ above and the assumption $p^{0.175} \cdot \Delta \leq 10^{-7}$ already ensure (21), which guarantees $\mathbf{e} \cdot \alpha(\Phi^\otimes, \mathcal{M}, C^\otimes) \cdot \Delta \leq 1$ and $\Delta^2 \cdot \lambda(\Phi^\otimes, \mathcal{M}, C^\otimes) \leq 1/100$ for all $C^\otimes \in \mathcal{C}^\otimes$. Thus in the following we only need to show how to construct the state tensorizations (and thus $\Phi^\otimes$) and the marking $\mathcal{M}$ so that $Q(\Phi^\otimes) = 2$, $\kappa(\Phi^\otimes) \leq 2$, and no $\mathcal{E}_C^{(1)}$ or $\mathcal{E}_C^{(2)}$ happens.

For each $C \in \mathcal{C}$ and integer $m \geq 2$, define $S(C, m) = \{v \in \mathsf{vbl}(C) \,|\, |\Omega_v| = m\}$. Then $p_C = \prod_{m=2}^{+\infty} m^{-|S(C,m)|}$. Let $t_1, t_2 > 0$ be the constants used in the proof of Lemma 5.15, i.e.,

$$t_1 = \ln \left(\frac{(1-\eta)(\eta+\tau_1)}{\eta \cdot (1-\eta-\tau_1)} \right) \in [1.1659, 1.1660] \quad \text{and} \quad t_2 = \ln \left(\frac{\eta \cdot (1-\eta+\tau_2)}{(1-\eta)(\eta-\tau_2)} \right) \in [1.0035, 1.0036].$$

Our construction will satisfy the following proposition. We will provide its detail and proof soon. The key part is Item (3) which intuitively says the simple binary case is actually the worst case.

Proposition 5.17. *For each $v \in V$, the construction of the state tensorization $\mathcal{T}_v$ and the marking on internal nodes of $\mathcal{T}_v$ is randomized and satisfies the following properties.*

(1) For each $v \in V$, the construction is independent and takes $O(|\Omega_v|) = O(Q(\Phi))$ time.

(2) Each possible $\mathcal{T}_v$ is a binary tree where $\frac{W(z\to z_1)}{W(z\to z_2)} \le 2$ (Recall $W(\cdot)$ from (16)) holds for any internal node $z \in \mathcal{T}_v$ and $\{z_1, z_2\} = \mathsf{childs}(z)$.

(3) For each $v \in V$ and $q \in \Omega_v$, we have

$$\mathbb{E}\left[\mathbf{e}^{-t_1 \cdot X(v,q)-(\eta+\tau_1)\cdot t_1 \cdot \log(|\Omega_v|)}\right] \le |\Omega_v|^{-\mathsf{KL}(\eta+\tau_1\|\eta)}$$

and

$$\mathbb{E}\left[\mathbf{e}^{t_2 \cdot X(v,q)+(\eta-\tau_2)\cdot t_2 \cdot \log(|\Omega_v|)}\right] \le |\Omega_v|^{-\mathsf{KL}(\eta-\tau_2\|\eta)}.$$

We first finish the proof assuming Proposition 5.17. Note that both $Q(\Phi^{\otimes}) = 2$ and $\kappa(\Phi^{\otimes}) \le 2$ follow from Item (2) of Proposition 5.17. Thus we focus on how to make sure no $\mathcal{E}_C^{(1)}$ or $\mathcal{E}_C^{(2)}$ happens. Similarly as in the proof of Lemma 5.16, we have

$$\mathbf{Pr}\left[\mathcal{E}_C^{(1)}\right] = \mathbf{Pr}\left[-t_1 \cdot \sum_{m=2}^{+\infty}\sum_{v\in S(C,m)} X(v,\sigma_{\mathsf{False}}^C(v)) > (\eta+\tau_1)\cdot t_1 \cdot \sum_{m=2}^{+\infty} |S(C,m)| \cdot \log(m)\right]$$

$$\le \mathbb{E}\left[\exp\left\{-t_1 \cdot \sum_{m=2}^{+\infty}\sum_{v\in S(C,m)} X(v,\sigma_{\mathsf{False}}^C(v)) - (\eta+\tau_1)\cdot t_1 \cdot \sum_{m=2}^{+\infty} |S(C,m)| \cdot \log(m)\right\}\right]$$

$$= \prod_{m=2}^{+\infty}\prod_{v\in S(C,m)} \mathbb{E}\left[\mathbf{e}^{-t_1\cdot X(v,\sigma_{\mathsf{False}}^C(v))-(\eta+\tau_1)\cdot t_1 \cdot \log(m)}\right] \qquad \text{(by Item (1) of Proposition 5.17)}$$

$$\le \prod_{m=2}^{+\infty}\prod_{v\in S(C,m)} m^{-\mathsf{KL}(\eta+\tau_1\|\eta)} \qquad \text{(by Item (3) of Proposition 5.17)}$$

$$= p_C^{\mathsf{KL}(\eta+\tau_1\|\eta)} \le p^{\mathsf{KL}(\eta+\tau_1\|\eta)}. \qquad \text{(since } p_C = \prod_{m=2}^{+\infty} m^{-|S(C,m)|}\text{)}$$

We also have

$$\mathbf{Pr}\left[\mathcal{E}_C^{(2)}\right] = \mathbf{Pr}\left[t_2 \cdot \sum_{m=2}^{+\infty}\sum_{v\in S(C,m)} X(v,\sigma_{\mathsf{False}}^C(v)) > -(\eta-\tau_2)\cdot t_2 \cdot \sum_{m=2}^{+\infty} |S(C,m)| \cdot \log(m)\right]$$

$$\le \prod_{m=2}^{+\infty}\prod_{v\in S(C,m)} \mathbb{E}\left[\mathbf{e}^{t_2\cdot X(v,\sigma_{\mathsf{False}}^C(v))+(\eta-\tau_2)\cdot t_2 \cdot \log(m)}\right]$$

$$\le p^{\mathsf{KL}(\eta-\tau_2\|\eta)}.$$

Let $\mathcal{E}_C = \mathcal{E}_C^{(1)} \vee \mathcal{E}_C^{(2)}$. By Item (1) of Proposition 5.17, the constructions are independent for each variable. Therefore $\mathcal{E}_C$ depends only on the constructions over $\mathsf{vbl}(C)$ and thus correlates with at most Δ many $\mathcal{E}_{C'}$ (including itself) and

$$\mathbf{Pr}\left[\mathcal{E}_C\right] \le 2 \cdot p^{\min\{\mathsf{KL}(\eta+\tau_1\|eta),\mathsf{KL}(\eta-\tau_2\|\eta)\}} \le 2 \cdot p^{0.175}.$$

By our assumption $p^{0.175} \cdot \Delta \le 10^{-7}$, we apply Theorem 5.1 to find a construction of the state tensorizations (and thus $\Phi^{\otimes}$) and the marking $\mathcal{M}$ to avoid all $\mathcal{E}_C = \mathcal{E}_C^{(1)} \vee \mathcal{E}_C^{(2)}$. The running time is $O(k(\Phi)\Delta|V|) \cdot O(Q(\Phi))$ where $O(Q(\Phi))$ is from Item (1) of Proposition 5.17. $\square$

Putting everything together, we have the following corollary which justifies Remark 5.12.

Corollary 5.18. *There exists a Las Vegas algorithm which takes as input an atomic CSP $\phi = \left(V, (\Omega_v, \mathcal{D}_v)_{v \in V}, \mathcal{C}\right)$ such that the following holds.*

If each $\mathcal{D}_v$ is the uniform distribution and $p^{0.175} \cdot \Delta \leq 10^{-7}$, then the algorithm outputs a perfect uniform random solution of Φ in expected time $O\left(k\Delta^3 Q^2 |V| \log(Q|V|)\right)$.

Proof. By fixing the variable which has domain size 1, we may safely assume $|\Omega_v| \geq 2$ for all $v \in V$.

We keep using Lemma 5.16 to construct $\Phi^\otimes = \left(Z, (\Omega_z, \mathcal{D}_z)_{z \in Z}, \mathcal{C}^\otimes\right)$ and $\mathcal{M}$ until they satisfy the conditions in Theorem 5.2. Then we run the algorithm in Theorem 5.2 to obtain a random solution of $\Phi^\otimes$ distributed perfectly as $\mu_{\mathsf{True}}^{\mathcal{C}^\otimes}$. Then by Proposition 5.9, we perform Trans operation to obtain $\sigma^{\mathsf{Trans}} \sim \mu_{\mathsf{True}}^{\mathcal{C}}$ which is just a perfect uniform random solution of Φ.

To check the running time, by Fact 5.8 we have $\Delta(\Phi^\otimes) = \Delta(\Phi)$, $p(\Phi^\otimes) = p(\Phi)$, $k(\Phi^\otimes) \leq k(\Phi) \cdot Q(\Phi)$[20], and $|Z| \leq Q(\Phi)|V|$. Therefore by Lemma 5.16 and Theorem 5.2, it takes $O(kQ\Delta|V|)$ time in expectation to construct $\Phi^\otimes$ and $\mathcal{M}$, and $O\left(k\Delta^3 Q^2 |V| \log(Q|V|)\right)$ time in expectation to do the sampling. □

Before proving Proposition 5.17, we set up some technical lemmas.

Fact 5.19 (e.g., [Hoe94, Lemma 1 and Equation (4.16)]). *Let X be an arbitrary random variable with $a \leq X \leq b$ almost surely. Let $c = \mathbb{E}[X]$. Then for any $t \in \mathbb{R}$, we have*

$$\mathbb{E}\left[\mathbf{e}^{t \cdot X}\right] \leq \frac{b - c}{b - a} \cdot \mathbf{e}^{a \cdot t} + \frac{c - a}{b - a} \cdot \mathbf{e}^{b \cdot t} \leq \mathbf{e}^{\frac{(b-a)^2 t^2}{8}} \cdot \mathbf{e}^{c \cdot t}.$$

Fact 5.20. *Our choice of $\eta, \tau_1, \tau_2, t_1, t_2$ satisfies*

$$\frac{\log^2(3)t_1^2}{8} \leq \left(\tau_1 t_1 - \frac{\mathsf{KL}(\eta + \tau_1 \| \eta)}{\log(\mathbf{e})}\right) \log(x) \text{ and } \frac{\log^2(3)t_2^2}{8} \leq \left(\tau_2 t_2 - \frac{\mathsf{KL}(\eta - \tau_2 \| \eta)}{\log(\mathbf{e})}\right) \log(x) \text{ for all } x \geq 8,$$

and

$$\frac{t_1^2}{8} \leq \left(\tau_1 t_1 - \frac{\mathsf{KL}(\eta + \tau_1 \| \eta)}{\log(\mathbf{e})}\right) \log(x) \text{ and } \frac{t_2^2}{8} \leq \left(\tau_2 t_2 - \frac{\mathsf{KL}(\eta - \tau_2 \| \eta)}{\log(\mathbf{e})}\right) \log(x) \text{ for all } x \in \{3, 4, 6, 7\},$$

and

$$\frac{\log^2(5/2)t_1^2}{8} \leq \left(\tau_1 t_1 - \frac{\mathsf{KL}(\eta + \tau_1 \| \eta)}{\log(\mathbf{e})}\right) \log(x) \text{ and } \frac{\log^2(5/2)t_2^2}{8} \leq \left(\tau_2 t_2 - \frac{\mathsf{KL}(\eta - \tau_2 \| \eta)}{\log(\mathbf{e})}\right) \log(x) \text{ for } x = 5.$$

Proof. Since $\log(x)$ is an increasing function for $x > 0$, it only needs to verify the first two inequalities at $x = 8$ and the middle two at $x = 3$. Then all of them can be verified numerically. □

Now we give the proof of Proposition 5.17.

Proof of Proposition 5.17. Our construction will be independent for each $v \in V$ and depend only on $|\Omega_v|$. Item (1) (2) will be evident as we describe the construction.

Let $N = |\Omega_v| \geq 2$. The simplest case $N = 2$ is essentially Lemma 5.15: $\mathcal{T}_v$ has one root with two leaf nodes, and we put the root into $\mathcal{M}$ with probability η. By the choice of t_1, t_2 (See the calculation in Lemma 5.15), the two inequalities in Item (3) are actually equality.

With foresight, our construction for $N \geq 3$ will satisfy the following conditions.

(A) $\mathbb{E}[X(v, q)] = \eta \log(1/N)$ for each $q \in \Omega_v$.

---

[20]Actually we can upper bound $k(\Phi^\otimes)$ by $k(\Phi) \cdot O(\log(Q(\Phi)))$. This is because the state tensorizations here are all "balanced" binary trees of depth $O(\log(Q(\Phi)))$. However this only slightly improves the bound on the running time which is not our main focus.

(B) If $N = 5$, then $X(v, q) \in [a_N, b_N]$ always holds for all $q \in \Omega_v$ where $b_N - a_N \leq \log(5/2)$.

(C) If $N \in \{3, 4, 6, 7\}$, then $X(v, q) \in [a_N, b_N]$ always holds for all $q \in \Omega_v$ where $b_N - a_N \leq 1$.

(D) If $N \geq 8$, then $X(v, q) \in [a_N, b_N]$ always holds for all $q \in \Omega_v$ where $b_N - a_N \leq \log(3)$.

Then we verify Item (3) given Item (A) (B) (C) (D) here:

$$
\begin{aligned}
\mathbb{E}\left[\mathbf{e}^{-t_1 \cdot X(v,q) - (\eta + \tau_1) \cdot t_1 \cdot \log(N)}\right] &= \mathbb{E}\left[\mathbf{e}^{-t_1 \cdot (X(v,q) - \eta \log(1/N)) - \tau_1 \cdot t_1 \cdot \log(N)}\right] \\
&\leq \exp\left\{\frac{(b_N - a_N)^2 t_1^2}{8} - \tau_1 \cdot t_1 \cdot \log(N)\right\} \quad \text{(by Fact 5.19 and Item (A))} \\
&\leq \exp\left\{-\mathsf{KL}(\eta + \tau_1 \| \eta) \cdot \frac{\log(N)}{\log(\mathbf{e})}\right\} = N^{-\mathsf{KL}(\eta + \tau_1 \| \eta)} \\
&\qquad\qquad\qquad\qquad\qquad\qquad \text{(by Item (B) (C) (D) and Fact 5.20)}
\end{aligned}
$$

and similarly

$$
\begin{aligned}
\mathbb{E}\left[\mathbf{e}^{t_2 \cdot X(v,q) + (\eta - \tau_2) \cdot t_2 \cdot \log(N)}\right] &= \mathbb{E}\left[\mathbf{e}^{t_2 \cdot (X(v,q) - \eta \log(1/N)) - \tau_2 \cdot t_2 \cdot \log(N)}\right] \\
&\leq \exp\left\{\frac{(b_N - a_N)^2 t_2^2}{8} - \tau_2 \cdot t_2 \cdot \log(N)\right\} \quad \text{(by Fact 5.19 and Item (A))} \\
&\leq N^{-\mathsf{KL}(\eta - \tau_2 \| \eta)}. \qquad\qquad \text{(by Item (B) (C) (D) and Fact 5.20)}
\end{aligned}
$$

Now we give the construction for $N \geq 3$ and check Item (A) (B) (C) (D).

$\boxed{N = 5.}$ Let $p_5 \in [0, 1]$ be a constant to be determined. The construction for $N = 5$ is as follows.

- In Figure 3, we select the left tree with probability p_5 and the right with probability $1 - p_5$.
- After fixing the tree, we assign elements in Ω_v to the leaf nodes uniformly to obtain $\mathcal{T}_v$.
- Finally we put the internal nodes boxed by dashed squares into the marking $\mathcal{M}$.
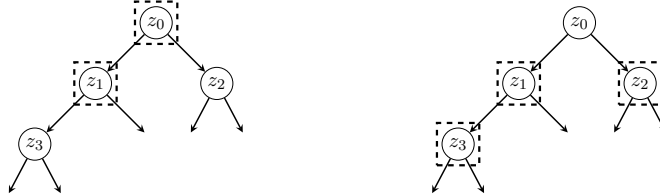
Figure 3: The construction for $N = 5$. Internal nodes boxed by dashed squares are put in $\mathcal{M}$.

Recall $X(v, q) = \sum_{z \in \mathsf{path}(q, \mathcal{T}_v) \cap \mathcal{M}} \log(\mathcal{D}_z(z(q)))$ where $z(q) \in \mathsf{childs}(z)$ is the child node of z such that $q \in \mathsf{leafs}(z(q))$. Since the leaf nodes are uniform, for any fixed $q \in \Omega_v$ we have

$$
\mathbb{E}\left[X(v, q) \mid \text{the left tree}\right] = (4\log(2/5) + \log(1/5))/5 < \eta \log(1/5)
$$

and

$$
\mathbb{E}\left[X(v, q) \mid \text{the right tree}\right] = (3\log(1/3) + 2\log(1/2))/5 > \eta \log(1/5).
$$

Thus we can easily set p_5 to make sure

$$
\mathbb{E}\left[X(v, q)\right] = p_5 \cdot \mathbb{E}\left[X(v, q) \mid \text{the left tree}\right] + (1 - p_5) \cdot \mathbb{E}\left[X(v, q) \mid \text{the right tree}\right] = \eta \log(1/5),
$$

which proves Item (A). As for Item (B), it suffices to observe

$$
X(v, q) \in \{\log(1/5), \log(1/3), \log(2/5), \log(1/2)\} \subseteq [\log(1/5), \log(1/2)].
$$

49

$\boxed{N \in \{3,4,6,7\}.}$ The construction for $N \in \{3,4,6,7\}$ is the same as $N = 5$ above except that we now use Figure 4: we will mix the left and right trees to make sure $\mathbb{E}[X(v,q)] = \eta \log(1/N)$ as demanded by Item (A). To do so it suffices to check $\eta \log(1/N)$ is sandwiched between $\mathbb{E}[X(v,q) \,|\, \text{the left tree}]$ and $\mathbb{E}[X(v,q) \,|\, \text{the right tree}]$.
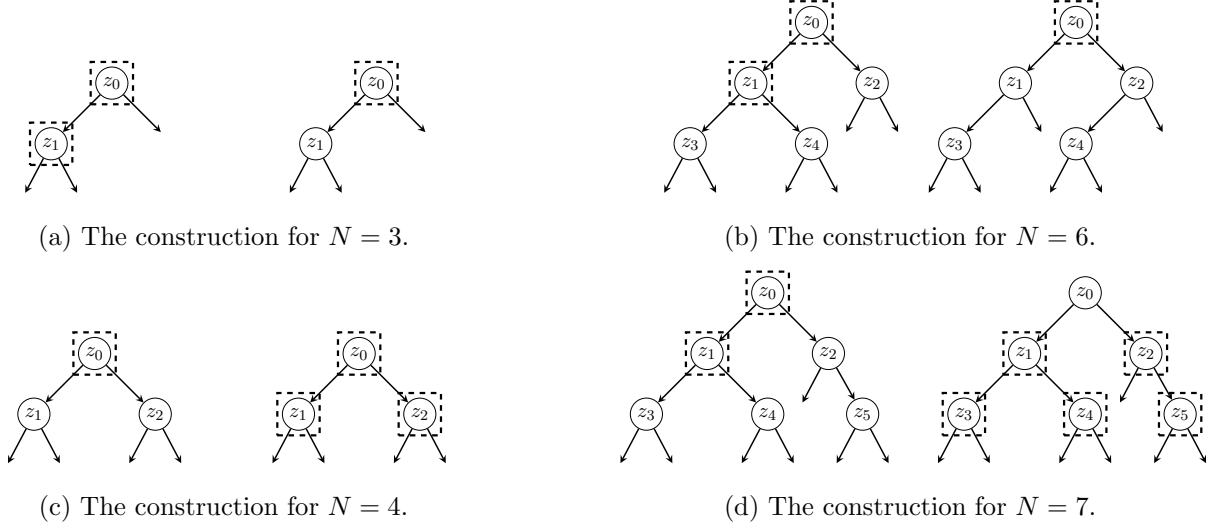

(a) The construction for $N = 3$.

(b) The construction for $N = 6$.

(c) The construction for $N = 4$.

(d) The construction for $N = 7$.

Figure 4: The construction for $N \in \{3,4,6,7\}$.

For $N = 3$ we use Figure 4a. Then $X(v,q) \in \{\log(1/3), \log(2/3)\}$ which proves Item (C). Also

$$\mathbb{E}[X(v,q) \,|\, \text{the left tree}] = \log(1/3) < \eta \log(1/3)$$

and

$$\mathbb{E}[X(v,q) \,|\, \text{the right tree}] = (\log(1/3) + 2\log(2/3))/3 > \eta \log(1/3).$$

For $N = 4$ we use Figure 4c. Then $X(v,q) \in \{\log(1/2), \log(1/4)\}$ which proves Item (C). Also

$$\mathbb{E}[X(v,q) \,|\, \text{the left tree}] = \log(1/2) > \eta \log(1/4)$$

and

$$\mathbb{E}[X(v,q) \,|\, \text{the right tree}] = \log(1/4) < \eta \log(1/4).$$

For $N = 6$ we use Figure 4b. Then $X(v,q) \in \{\log(1/3), \log(1/2)\}$ which proves Item (C). Also

$$\mathbb{E}[X(v,q) \,|\, \text{the left tree}] = \log(1/3) < \eta \log(1/6)$$

and

$$\mathbb{E}[X(v,q) \,|\, \text{the right tree}] = \log(1/2) > \eta \log(1/6).$$

For $N = 7$ we use Figure 4d. Then $X(v,q) \in \{\log(2/7), \log(3/7), \log(1/4), \log(1/3)\}$ which proves Item (C). Also

$$\mathbb{E}[X(v,q) \,|\, \text{the left tree}] = (4\log(2/7) + 3\log(3/7))/7 > \eta \log(1/7)$$

and

$$\mathbb{E}[X(v,q) \,|\, \text{the right tree}] = (4\log(1/4) + 3\log(1/3))/7 < \eta \log(1/7).$$

$\boxed{N \geq 8.}$ For each $x \in [N]$ define $A(N, x) = \lfloor N/x \rfloor \cdot (x + 1) - N$ and $B(N, x) = N - \lfloor N/x \rfloor \cdot x$. Let $R = \lfloor N^{1-\eta} \rfloor$. The proof of the following technical result is deferred to Appendix A.

Fact 5.21. *If $x \in \{R - 1, R, R + 1\}$, then $1 \leq x \leq N$ and $A(N, x), B(N, x) \geq 0$.*

For each $x \in \{R - 1, R, R + 1\}$ we have the following construction (See Figure 5 for an intuition), the correctness of which is guaranteed by Fact 5.21.

- Let $\mathcal{T}_1, \ldots, \mathcal{T}_{A(N,x)}$ (resp., $\mathcal{T}_{A(N,x)+1}, \ldots, \mathcal{T}_{A(N,x)+B(N,x)}$) be balanced binary trees that each has x (resp., $x + 1$) leaf nodes.[21]

- Define distribution $\mathcal{D}_x$ supported on $\{\mathcal{T}_1, \ldots, \mathcal{T}_{A(N,x)+B(N,x)}\}$ by setting

$$\mathcal{D}_x(\mathcal{T}_i) = \begin{cases} x/N & i \leq A(N, x), \\ (x+1)/N & i > A(N, x). \end{cases}$$

Then construct a binary tree on top of $\{\mathcal{T}_1, \ldots, \mathcal{T}_{A(N,x)+B(N,x)}\}$ using Lemma 5.10 and $\mathcal{D}_x$.[22]

- Assign elements in Ω_v uniformly to $A(N, x) \cdot x + B(N, x) \cdot (x + 1) = N$ leaf nodes to get $\mathcal{T}_v$.

- Finally we put all the nodes on top of $\{\mathcal{T}_1, \ldots, \mathcal{T}_{A(N,x)+B(N,x)}\}$ into the marking $\mathcal{M}$.
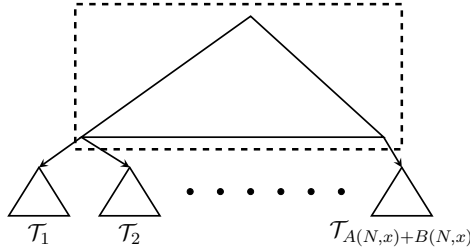
Figure 5: The construction for $N \geq 8$ and $x \in \{R - 1, R, R + 1\}$. Nodes inside the dashed square are put in $\mathcal{M}$.

Observe that for any fixed $x \in \{R - 1, R, R + 1\}$ and any $q \in \Omega_v$, the construction gives $X(v, q) \in \{\log(x/N), \log((x+1)/N)\}$ and

$$\mathbb{E}\left[X(v, q) \mid x\right] = \frac{A(N, x) \cdot x}{N} \log\left(\frac{x}{N}\right) + \frac{B(N, x) \cdot (x + 1)}{N} \log\left(\frac{x + 1}{N}\right) \in \left[\log\left(\frac{x}{N}\right), \log\left(\frac{x + 1}{N}\right)\right].$$

Therefore

$$\mathbb{E}\left[X(v, q) \mid x = R + 1\right] \geq \log((R + 1)/N) = \log\left(\left(\lfloor N^{1-\eta} \rfloor + 1\right)/N\right) \geq \eta \log(1/N)$$

and

$$\mathbb{E}\left[X(v, q) \mid x = R - 1\right] \leq \log(R/N) = \log\left(\lfloor N^{1-\eta} \rfloor/N\right) \leq \eta \log(1/N).$$

Now we have two cases:

---

[21]Here "balanced" simply means the sub-trees of sibling nodes have size difference at most 1. In particular, this guarantees Item (2) of Proposition 5.17.

[22]Note that $\kappa(\mathcal{D}_x) \leq (x + 1)/x \leq 2$. By Item (2) of Lemma 5.10, Item (2) of Proposition 5.17 is still preserved. Meanwhile, since $A(N, x) + B(N, x) \leq N/x = O(N^\eta)$, this step takes $O(N^\eta \log(N)) = O(N)$ time.

- If $\mathbb{E}\left[X(v,q)\,|\,x=R\right] \geq \eta\log(1/N)$[23], we mix the construction of $x = R-1$ and $x = R$ to make sure $\mathbb{E}\left[X(v,q)\right] = \eta\log(1/N)$ for Item (A). Then $X(v,q) \in \left\{\log\left(\frac{R-1}{N}\right), \log\left(\frac{R}{N}\right), \log\left(\frac{R+1}{N}\right)\right\}$ which verifies Item (D) since $R \geq 2$.

- Otherwise, we mix the construction of $x = R$ and $x = R+1$ in the similar way. Then $X(v,q) \in \left\{\log\left(\frac{R}{N}\right), \log\left(\frac{R+1}{N}\right), \log\left(\frac{R+2}{N}\right)\right\}$ which also verifies Item (D). $\qquad\square$

Acknowledgement

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. 4

[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. 4

[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

[BSVV08] Ivona Bezáková, Daniel Stefankovic, Vijay V. Vazirani, and Eric Vigoda. Accelerating simulated annealing for the permanent and combinatorial counting problems. *SIAM J. Comput.*, 37(5):1429–1454, 2008. doi:10.1137/050644033. 5

[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

---

[23]We emphasize that $\mathbb{E}\left[X(v,q)\,|\,x\right]$ has the same value for all $q \in \Omega_v$.

[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, 4, 11

[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, 6, 19

[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, 6, 9, 38

[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. 4

[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. 4

[FVY19]   Weiming Feng, Nisheeth K. Vishnoi, and Yitong Yin. Dynamic sampling from graphical models. In *STOC*, pages 1070–1081. ACM, 2019. 4

[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

[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. 31

[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. 7

[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. 4, 7, 29

[Hoe94]   Wassily Hoeffding. Probability inequalities for sums of bounded random variables. In *The collected works of Wassily Hoeffding*, pages 409–426. Springer, 1994. 37, 48

[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, 11

[HSZ19] Jonathan Hermon, Allan Sly, and Yumeng Zhang. Rapid mixing of hypergraph independent sets. *Random Struct. Algorithms*, 54(4):730–767, 2019. 8

[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. 4, 7, 29

[Hub04] Mark Huber. Perfect sampling using bounding chains. *Ann. Appl. Probab.*, 14(2):734–753, 2004. 4

[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, 5, 42

[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, 4, 5, 6, 8, 19, 42

[JSS20] Vishesh Jain, Ashwin Sah, and Mehtaab Sawhney. Perfectly sampling $k \geq (8/3 + o(1))\Delta$-colorings in graphs. *CoRR*, abs/2007.06360, 2020. URL: `https://arxiv.org/abs/2007.06360`, `arXiv:2007.06360`. 4

[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, 5

[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, 4

[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`. 4, 27

[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. 8

[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

[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, 4, 35

[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, 7, 29

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

[ŠVV09a] 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

[SVV09b] Daniel Stefankovic, Santosh S. Vempala, and Eric Vigoda. Adaptive simulated annealing: A near-optimal connection between sampling and counting. *J. ACM*, 56(3):18:1–18:36, 2009. `doi:10.1145/1516512.1516520`. 5

[Wig19] Avi Wigderson. *Mathematics and computation*. Princeton University Press, 2019. 6

A Proof of Fact 5.21

Recall our setting: $N \geq 8$ is an integer, $R = \lfloor N^{1-\eta} \rfloor$ where $\eta = 0.595$, $A(N, x) = \lfloor N/x \rfloor \cdot (x+1) - N$, and $B(N, x) = N - \lfloor N/x \rfloor \cdot x$.

Fact (Fact 5.21 restated). *If $x \in \{R - 1, R, R + 1\}$, then $1 \leq x \leq N$ and $A(N, x), B(N, x) \geq 0$.*

Proof. Since $1 \leq x \leq N$ is equivalent to $2 \leq R \leq N - 1$, we only need to check

$$2 = \lfloor 8^{1-\eta} \rfloor \leq \lfloor N^{1-\eta} \rfloor = R \leq N^{1-\eta} \leq \sqrt{N} \leq N - 1.$$

Also $B(N, x)$ is always non-negative as $\lfloor N/x \rfloor \leq N/x$. Hence we focus on the $A(N, x) \geq 0$ part.

Since $\lfloor t \rfloor > t - 1$, we have

$$A(N, x) > \left(\frac{N}{x} - 1 \right) \cdot (x + 1) - N = \frac{N}{x} - x - 1.$$

Since $n > t$ is equivalent to $n \geq \lfloor t + 1 \rfloor$ for all integer n, we have

$$A(N, x) \geq \left\lfloor \frac{N}{x} - x \right\rfloor.$$

Therefore if $x \leq \sqrt{N}$, then $A(N,x) \geq 0$. Since $R = \lfloor N^{1-\eta} \rfloor \leq \sqrt{N}$, this shows $A(N, R-1)$ and $A(N,R)$ are all non-negative. Now we deal with $A(N, R+1)$. Note that $\lfloor N^{1-\eta} \rfloor \leq N^{1-\eta} \leq \sqrt{N}-1$ holds for all $N \geq 18$, thus $R+1 \leq \sqrt{N}$ and $A(N, R+1) \geq 0$ for all $N \geq 18$. Finally for $8 \leq N \leq 17$, $A(N, R+1) \geq 0$ can be verified numerically. □