Deterministic counting Lovász local lemma beyond linear programming

Chunyang Wang
Nanjing University

Joint work with: Kun He (Chinese Academy of Science)
Yitong Yin (Nanjing University)

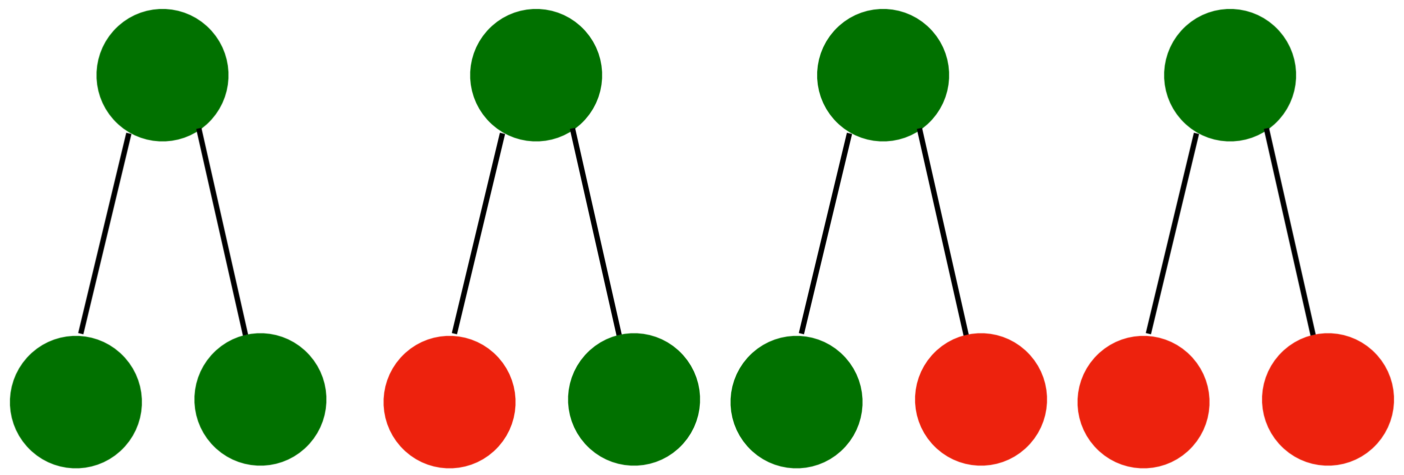
SODA 2023

Counting and Sampling

Given a weight function $w(\cdot)$ on state space Ω,

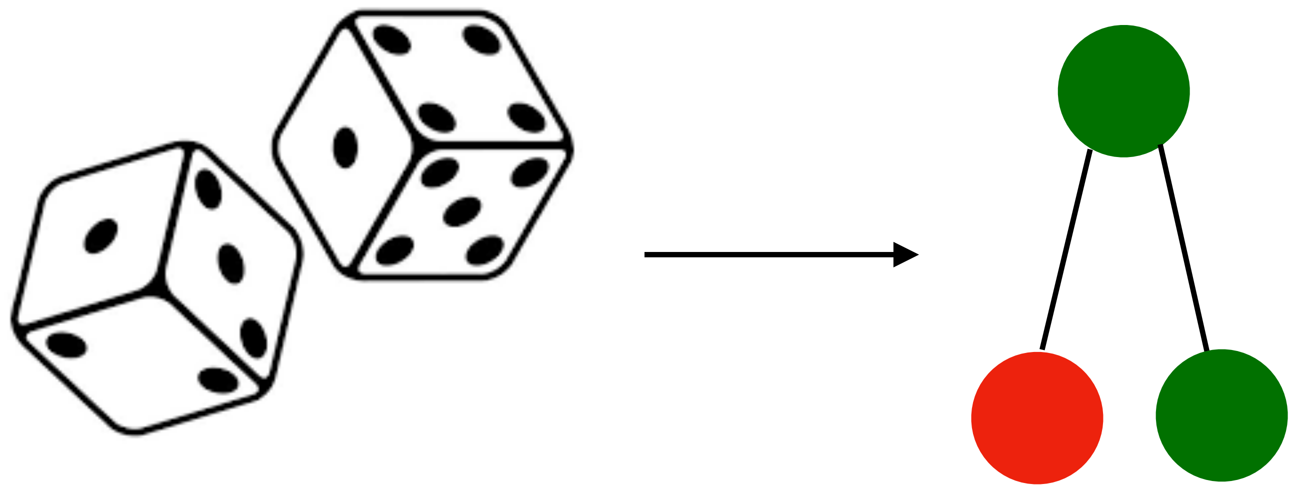
Counting problem:

compute $\sum_{x\in\Omega} w(x)$



Sampling problem:

Sample $x \sim \dfrac{w(x)}{\sum_{y\in\Omega} w(y)}$



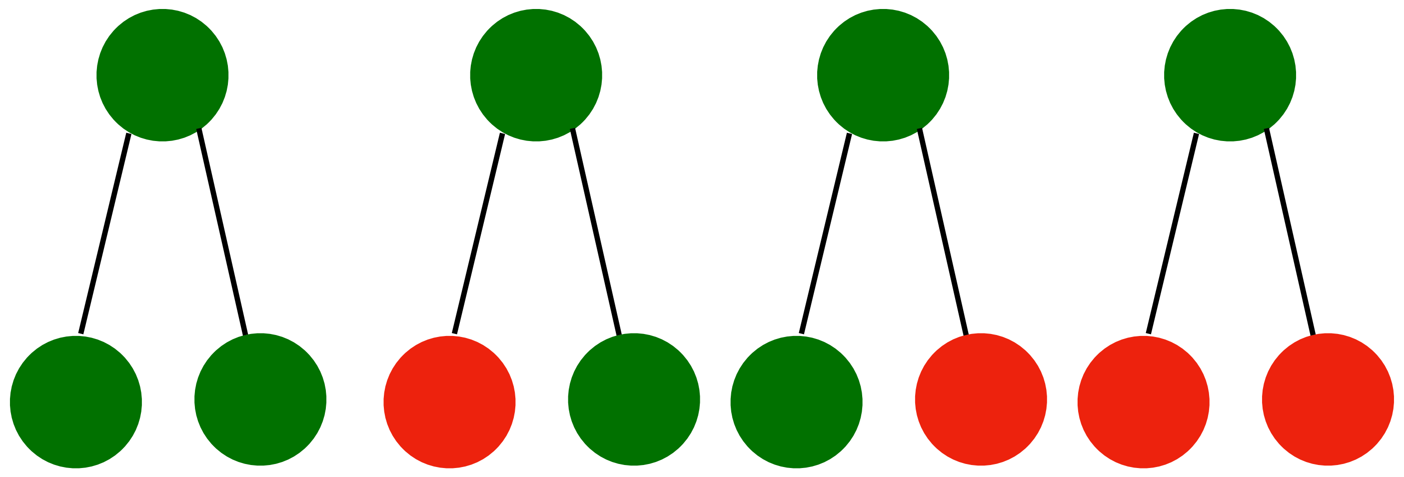
Counting and Sampling

Given a weight function $w(\cdot)$ on state space Ω,

typically exponentially large (or infinite)

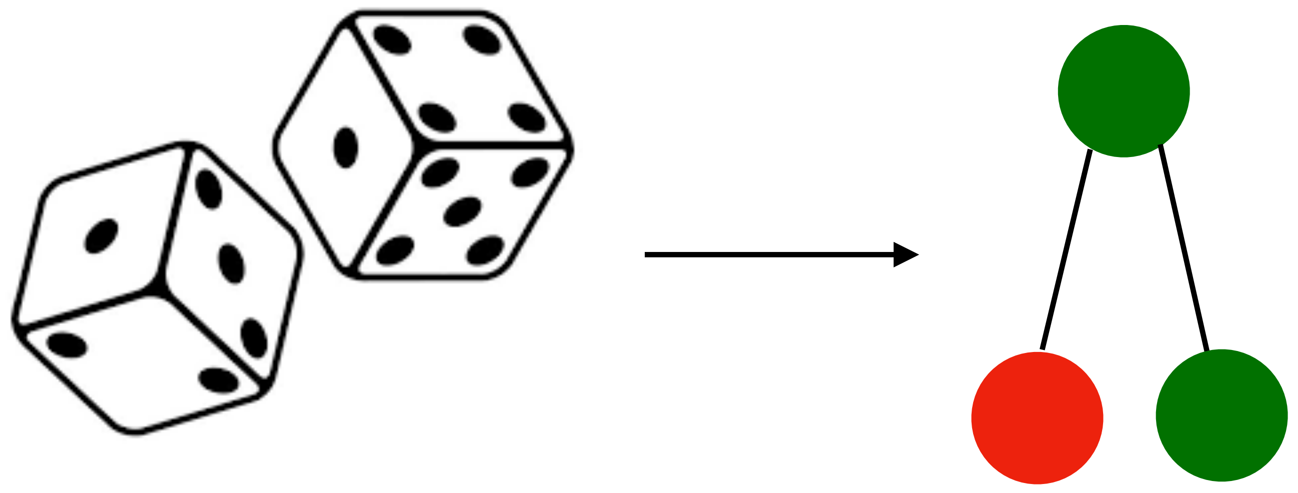
Counting problem:

compute $\sum_{x \in \Omega} w(x)$



Sampling problem:

Sample $x \sim \dfrac{w(x)}{\sum_{y \in \Omega} w(y)}$



Counting and Sampling

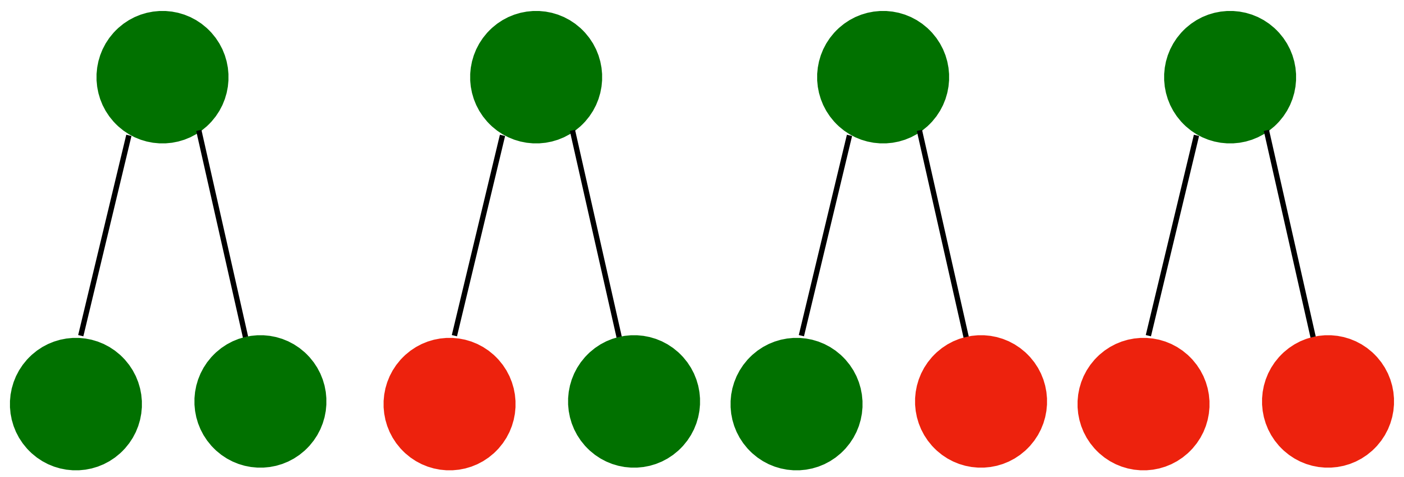
Given a weight function $w(\cdot)$ on state space Ω,

typically exponentially large (or infinite)

Counting problem:

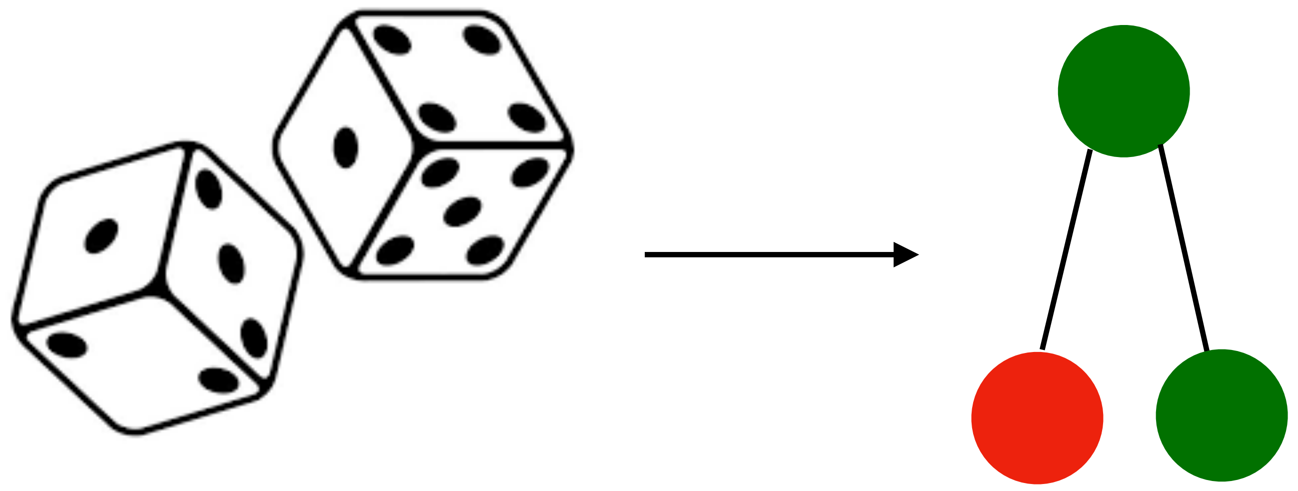
compute $\sum_{x \in \Omega} w(x)$

typically #P-hard to compute exactly



Sampling problem:

Sample $x \sim \frac{w(x)}{\sum_{y \in \Omega} w(y)}$



Counting and Sampling

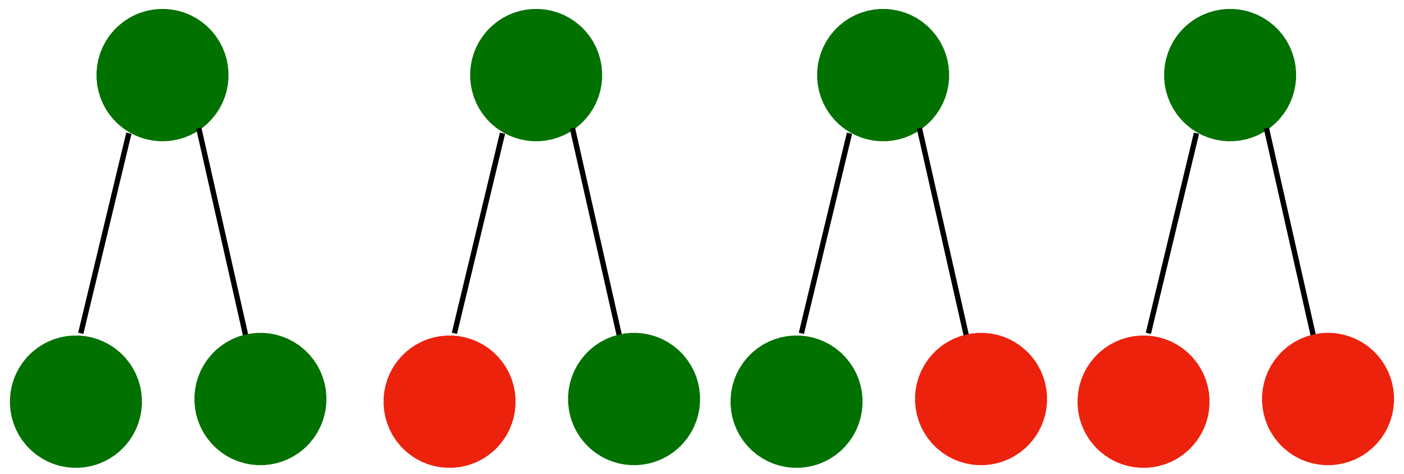
Given a weight function $w(\cdot)$ on state space Ω,

typically exponentially large (or infinite)

Counting problem:

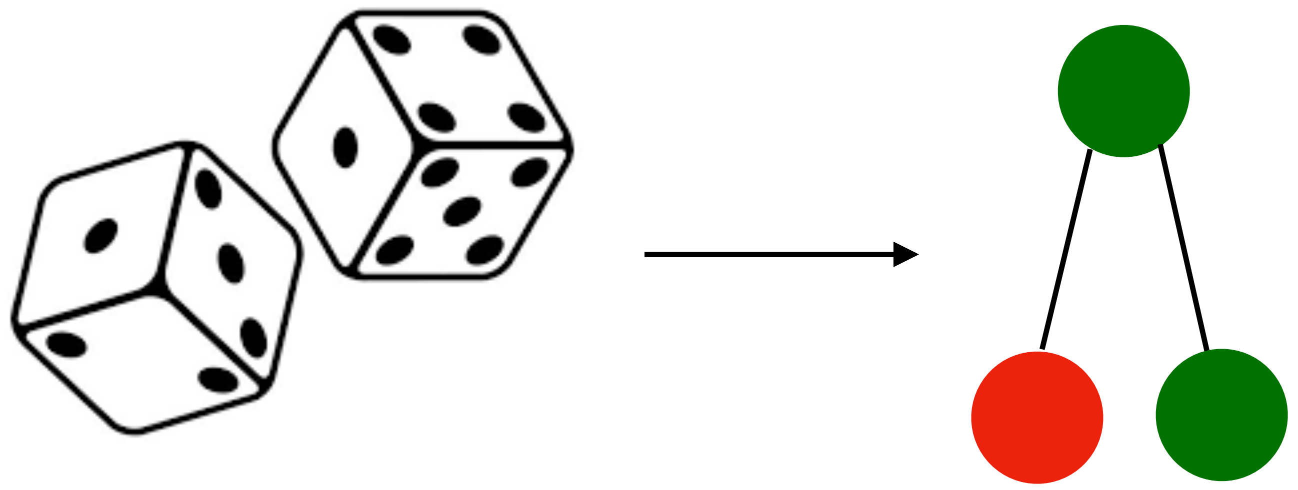
compute $\sum_{x\in\Omega} w(x)$

typically #P-hard to compute exactly



Sampling problem:

Sample $x \sim \dfrac{w(x)}{\sum_{y\in\Omega} w(y)}$



(Almost) Uniform Sampling

self-reduction

[Jerrum, Valiant, Vazirani 1986]

adaptive simulated annealing

[Dyer, Frieze, Kannan 1991]

[Štefankovič, Vempala, Vigoda 2009]

Randomized Approximate Counting

Deterministic Counting

Some approaches for deterministic counting:

- decay of correlation [Weitz ’06]
- zero-freeness [Barvinok ’16, Patel, Regts ’17]
- cluster-expansion [Helmuth, Perkins, Regts ’20, Jenssen, Keevash, Perkins ’20]
- linear programming for CSPs [Moitra ’19, Guo, Liao, Lu, Zhang ’20, Jain, Pham, Vuong ’21]

Deterministic Counting

Some approaches for deterministic counting:

- decay of correlation [Weitz ’06]
- zero-freeness [Barvinok ’16, Patel, Regts ’17]
- cluster-expansion [Helmuth, Perkins, Regts ’20, Jenssen, Keevash, Perkins ’20]
- linear programming for CSPs [Moitra ’19, Guo, Liao, Lu, Zhang ’20, Jain, Pham, Vuong ’21]

Developed independently from sampling algorithms!

Deterministic Counting

Some approaches for deterministic counting:

- decay of correlation [Weitz '06]
- zero-freeness [Barvinok '16, Patel, Regts '17]
- cluster-expansion [Helmuth, Perkins, Regts '20, Jenssen, Keevash, Perkins '20]
- linear programming for CSPs [Moitra '19, Guo, Liao, Lu, Zhang '20, Jain, Pham, Vuong '21]

Developed independently from sampling algorithms!

(Almost) Uniform Sampling $\xrightarrow{\text{Derandomization?}}$ Deterministic Approximate Counting

Constraint Satisfaction Problem

$$\Phi = (V, \boldsymbol{Q}, \mathscr{C})$$

Variables: $V = \{v_1, v_2, \ldots, v_n\}$ with finite domains Q_v for each $v \in V$

Constraints: $\mathscr{C} = \{c_1, c_2, \ldots, c_m\}$ with each $c \in \mathscr{C}$ defined on $\mathsf{vbl}(c) \subseteq V$

$$c : \bigotimes_{v \in \mathsf{vbl}(c)} Q_v \to \{\text{satisfied}, \text{not satisfied}\}$$

CSP solution: assignment $\boldsymbol{X} \in \bigotimes_{v \in V} Q_v$ s.t. all constraints are satisfied

Constraint Satisfaction Problem

$$\Phi = (V, \boldsymbol{Q}, \mathscr{C})$$

Variables: $V = \{v_1, v_2, \ldots, v_n\}$ with finite domains Q_v for each $v \in V$

Constraints: $\mathscr{C} = \{c_1, c_2, \ldots, c_m\}$ with each $c \in \mathscr{C}$ defined on $\mathsf{vbl}(c) \subseteq V$

$$c : \bigotimes_{v \in \mathsf{vbl}(c)} Q_v \to \{\text{satisfied}, \text{not satisfied}\}$$

CSP solution: assignment $\boldsymbol{X} \in \bigotimes_{v \in V} Q_v$ s.t. all constraints are satisfied

Decision: Can we efficiently decide if Φ has a solution?

Search: Can we efficiently find a solution of Φ?

Counting/Sampling: Can we efficiently (approximately) count the number of solutions/(almost) uniformly sample a solution of Φ?

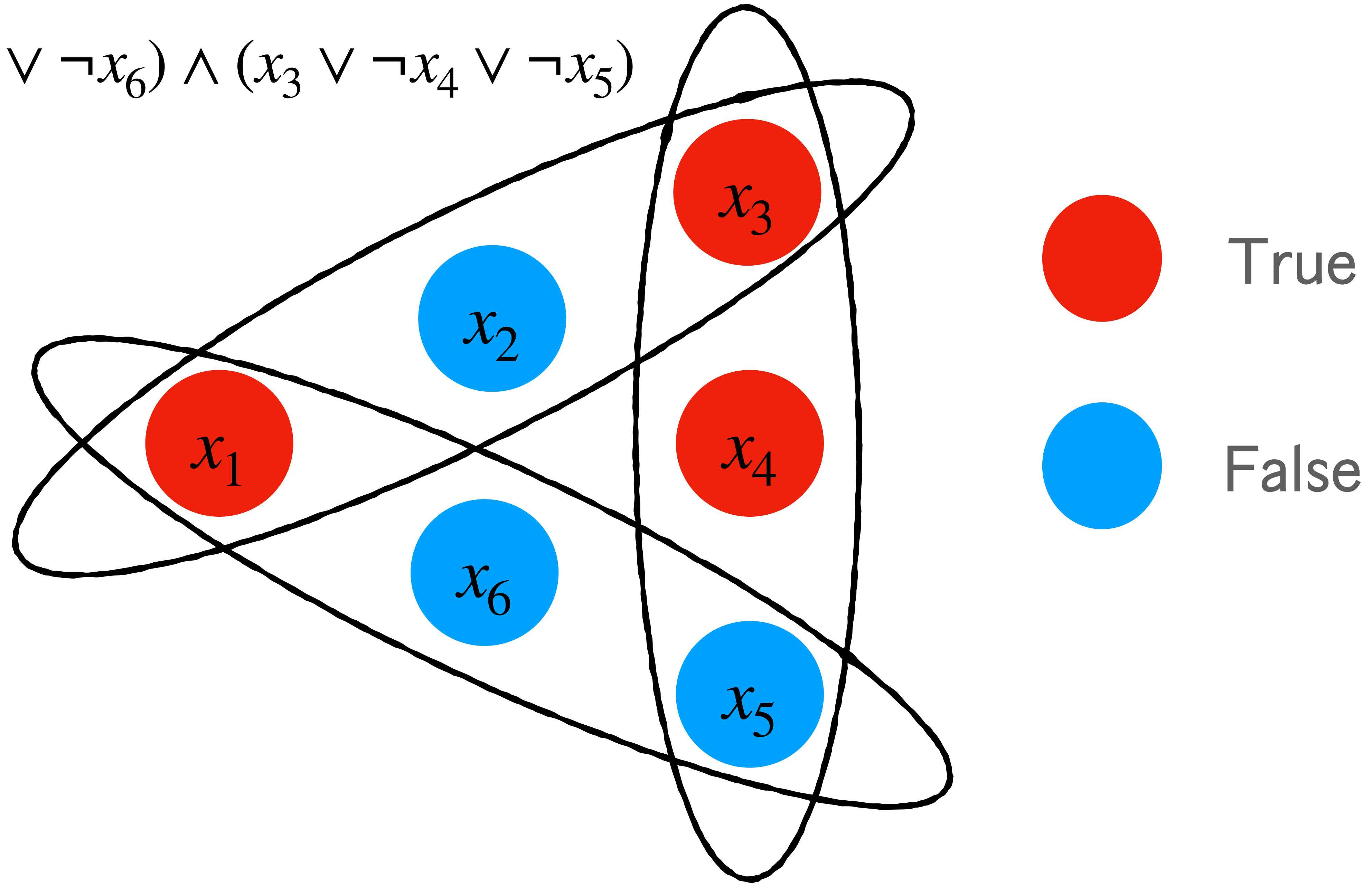
$$\Phi = (x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_5 \vee \neg x_6) \wedge (x_3 \vee \neg x_4 \vee \neg x_5)$$

Example: k-CNF

$V = \{x_1, x_2, \dots, x_n\}$, $\mathscr{C} = (C_1, C_2, \dots, C_m)$, $|C_i| = k$

$Q_v \in \{\text{True}, \text{False}\}$ for each $v \in V$

Solution: an assignment such that each clause (constraint) evaluates to True

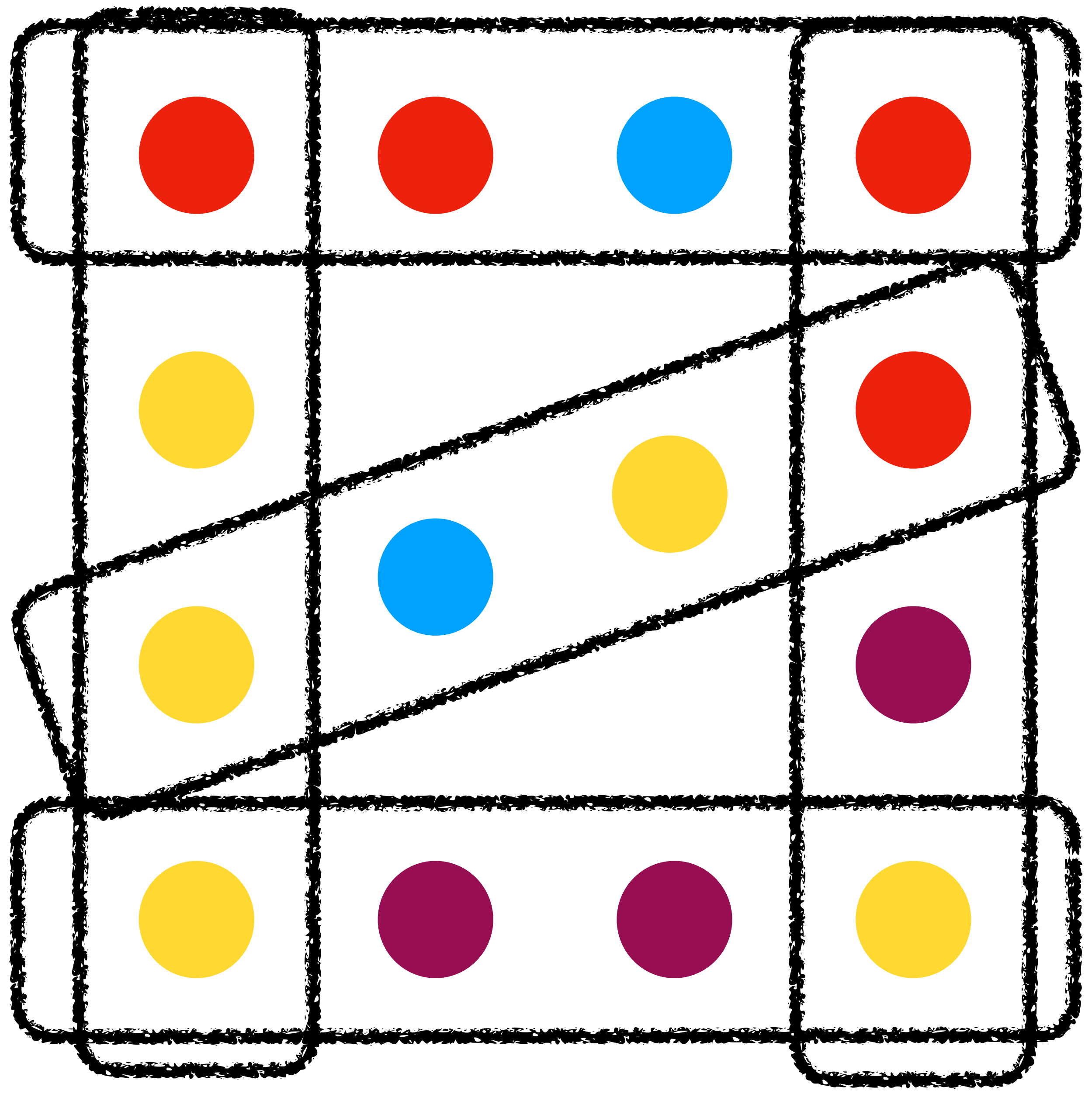


Example: hypergraph q-coloring

k-uniform hypergraph $H = (V, \mathscr{E})$

color set $[q]$ for each $v \in V$

Solution: an assignment such that no hyperedge (constraint) is monochromatic



Lovász Local Lemma

$$\Phi = (V, \boldsymbol{Q}, \mathscr{C})$$

Variable framework

- each $v \in V$ draws from Q_v uniformly and independently at random
- product distribution $\mathscr{P}$

Parameters

- **violation probability** $p = \max_{c \in \mathscr{C}} \Pr_{\mathscr{P}}[\neg c]$
- **constraint degree** $\Delta = \max_{c \in \mathscr{C}} |\{c' \in \mathscr{C} \mid \mathsf{vbl}(c) \cap \mathsf{vbl}(c') \neq \emptyset\}|$

Lovász Local Lemma

$$\Phi = (V, Q, \mathscr{C})$$

Variable framework

- each $v \in V$ draws from Q_v uniformly and independently at random
- product distribution $\mathscr{P}$

Parameters

- **violation probability** $p = \max_{c \in \mathscr{C}} \Pr_{\mathscr{P}}[\neg c]$
- **constraint degree** $\Delta = \max_{c \in \mathscr{C}} |\{c' \in \mathscr{C} \mid \mathsf{vbl}(c) \cap \mathsf{vbl}(c') \neq \emptyset\}|$

$$ep\Delta \leq 1$$

Lovász Local Lemma
[Erdos, Lovász '75]

Algorithmic Lovász Local Lemma
[Moser, Tardos '10]

A CSP solution exists
and can be efficiently found!

Counting/Sampling LLL

Input: a CSP formula $\Phi = (V, \boldsymbol{Q}, \mathscr{C})$ under LLL-like conditions $p\Delta^c \lesssim 1$ $\longrightarrow$ [BGGGS19,GGW22]: **NP-hard** if $p\Delta^2 \gtrsim 1$!

Output: Counting LLL: the approximate number of solutions of Φ

Sampling LLL: an (almost) uniform satisfying solution of Φ

Counting/Sampling LLL

Input: a CSP formula $\Phi = (V, \boldsymbol{Q}, \mathscr{C})$ under LLL-like conditions $p\Delta^c \lesssim 1$ → [BGGGS19,GGW22]: **NP-hard** if $p\Delta^2 \gtrsim 1$!

Output: Counting LLL: the approximate number of solutions of Φ

Sampling LLL: an (almost) uniform satisfying solution of Φ

Work	Instance	Condition	Technique
Moitra '17	k-CNF	$p\Delta^{60} \lesssim 1$	marginal approximator by linear programming
Guo, Liu, Lu, Zhang '19	hypergraph q-coloring	$p\Delta^{16} \lesssim 1$	
Jain, Pham, Vuong '21b	general CSP	$p\Delta^{7} \lesssim 1$	

Deterministic counting LLL: runs in $n^{\mathrm{poly}(k,\Delta,\log q)}$ time

Counting/Sampling LLL

Input: a CSP formula $\Phi = (V, \boldsymbol{Q}, \mathscr{C})$ under LLL-like conditions $p\Delta^c \lesssim 1$ $\longrightarrow$ [BGGGS19,GGW22]: **NP-hard** if $p\Delta^2 \gtrsim 1$!

Output: Counting LLL: the approximate number of solutions of Φ

Sampling LLL: an (almost) uniform satisfying solution of Φ

Work	Instance	Condition	Technique
Moitra '17	k-CNF	$p\Delta^{60} \lesssim 1$	marginal approximator by linear programming
Guo, Liu, Lu, Zhang '19	hypergraph q-coloring	$p\Delta^{16} \lesssim 1$	
Jain, Pham, Vuong '21b	general CSP	$p\Delta^{7} \lesssim 1$	

Deterministic counting LLL: runs in $n^{\mathrm{poly}(k,\Delta,\log q)}$ time

Work	Instance	Condition	Technique
Hermon, Sly, Zhang'16	monotone k-CNF	$p\Delta^{2} \lesssim 1$	Markov chain Monte Carlo (MCMC) or projected MCMC
Feng, Guo, Yin, Zhang '20	k-CNF	$p\Delta^{20} \lesssim 1$	
Feng, He, Yin,'21	atomic CSP	$p\Delta^{350} \lesssim 1$	
Jain, Pham, Vuong '21a He, Sun, Wu '21	atomic CSP	$p\Delta^{5.713} \lesssim 1$	
He, **W.**, Yin,'22	general CSP	$p\Delta^{7} \lesssim 1$	recursive marginal sampler

(Randomized) sampling LLL: runs in $\mathrm{poly}(n, k, \Delta, q)$ time

Our results

derandomized algorithm for counting LLL in an improved regime

A general CSP satisfying

$$q^2 \cdot k \cdot p \cdot \Delta^5 \leq \frac{1}{256\mathrm{e}^3}$$

$\Rightarrow$

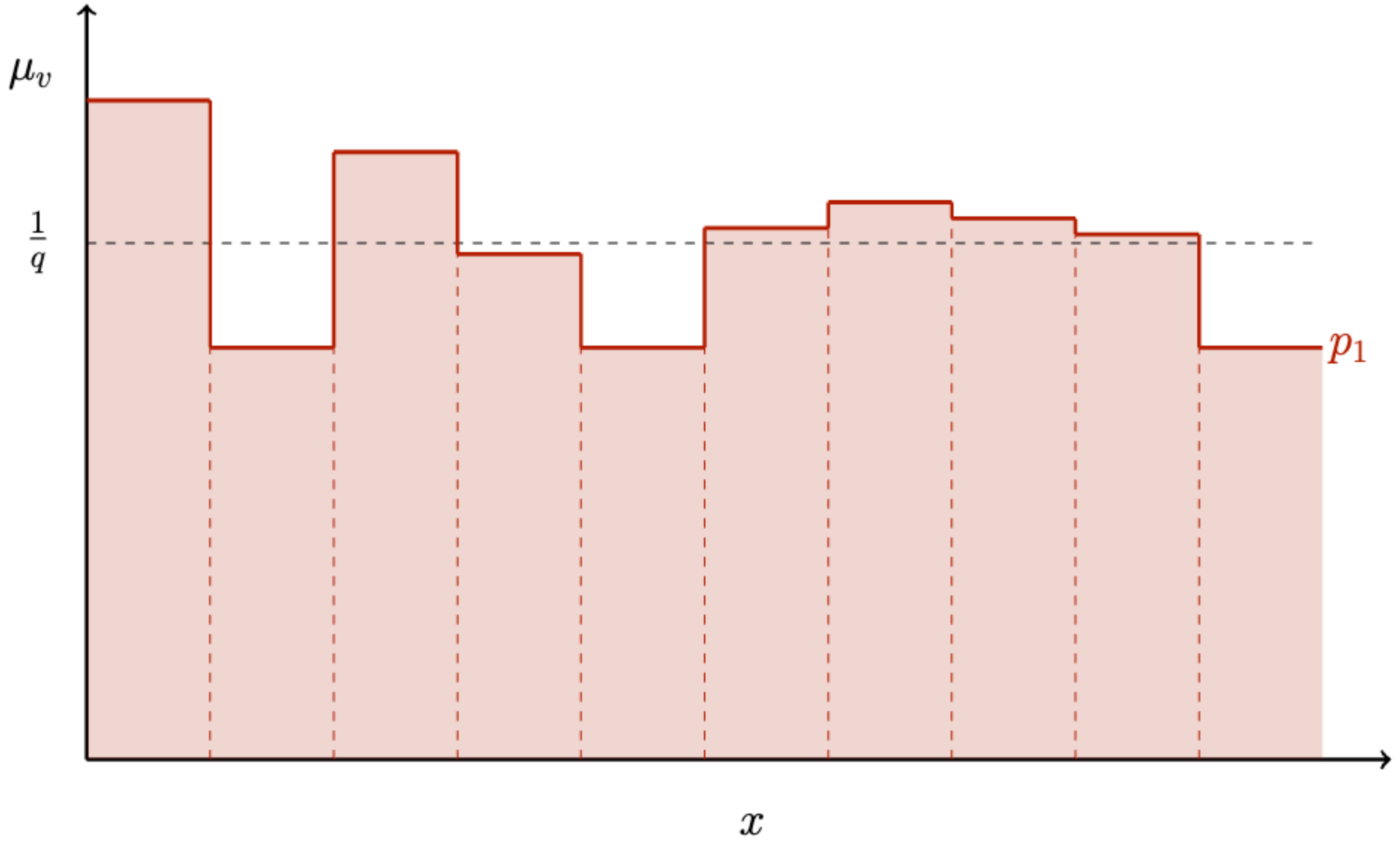
a deterministic FPTAS approximating # of satisfying solutions in time $n^{\mathrm{poly}(k,\Delta,\log q)}$

q: domain size
k: constraint width
p: violation probability
Δ: constraint degree
n: $|V|$

general CSPs:
$p\Delta^7 \lesssim 1 \to p\Delta^5 \lesssim 1$
atomic CSPs (including k-CNF):
$p\Delta^{5.713} \lesssim 1 \to p\Delta^5 \lesssim 1$

- is a derandomization of the recent fast sampling algorithm in [He, **W.**, Yin '22]
- relies on a combinatorial marginal approximator, which is arguably simpler than previous linear programming-based ones for counting LLL

Local Uniformity



μ: uniform distribution over solutions

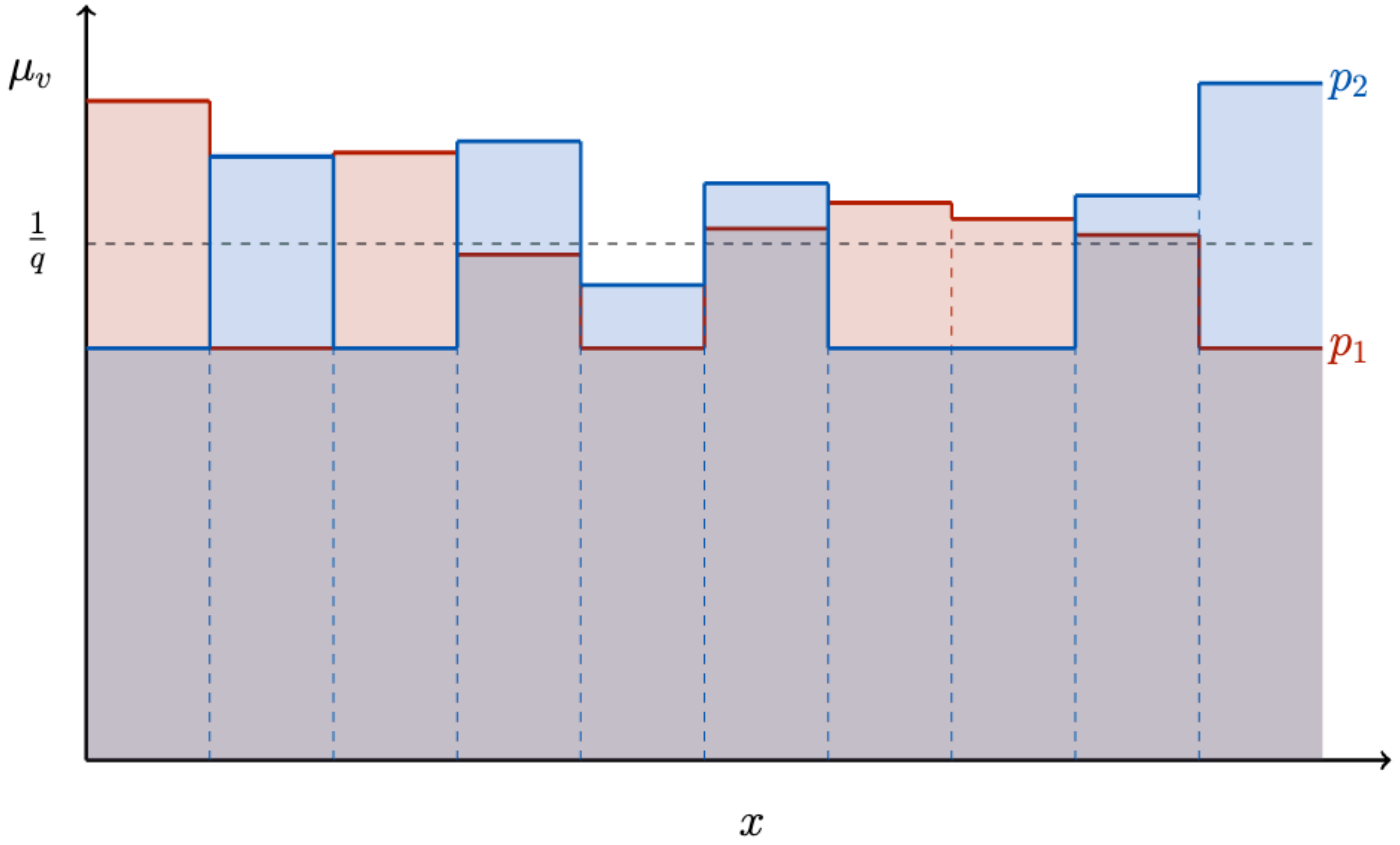
μ_v: marginal distribution at $v \in V$

[Haeupler, Saha, Srinivasan '11]:

LLL condition $\implies \mu_v \geq \theta$

where $\theta = (1 - o(1))\frac{1}{q}$

Local Uniformity



μ: uniform distribution over solutions

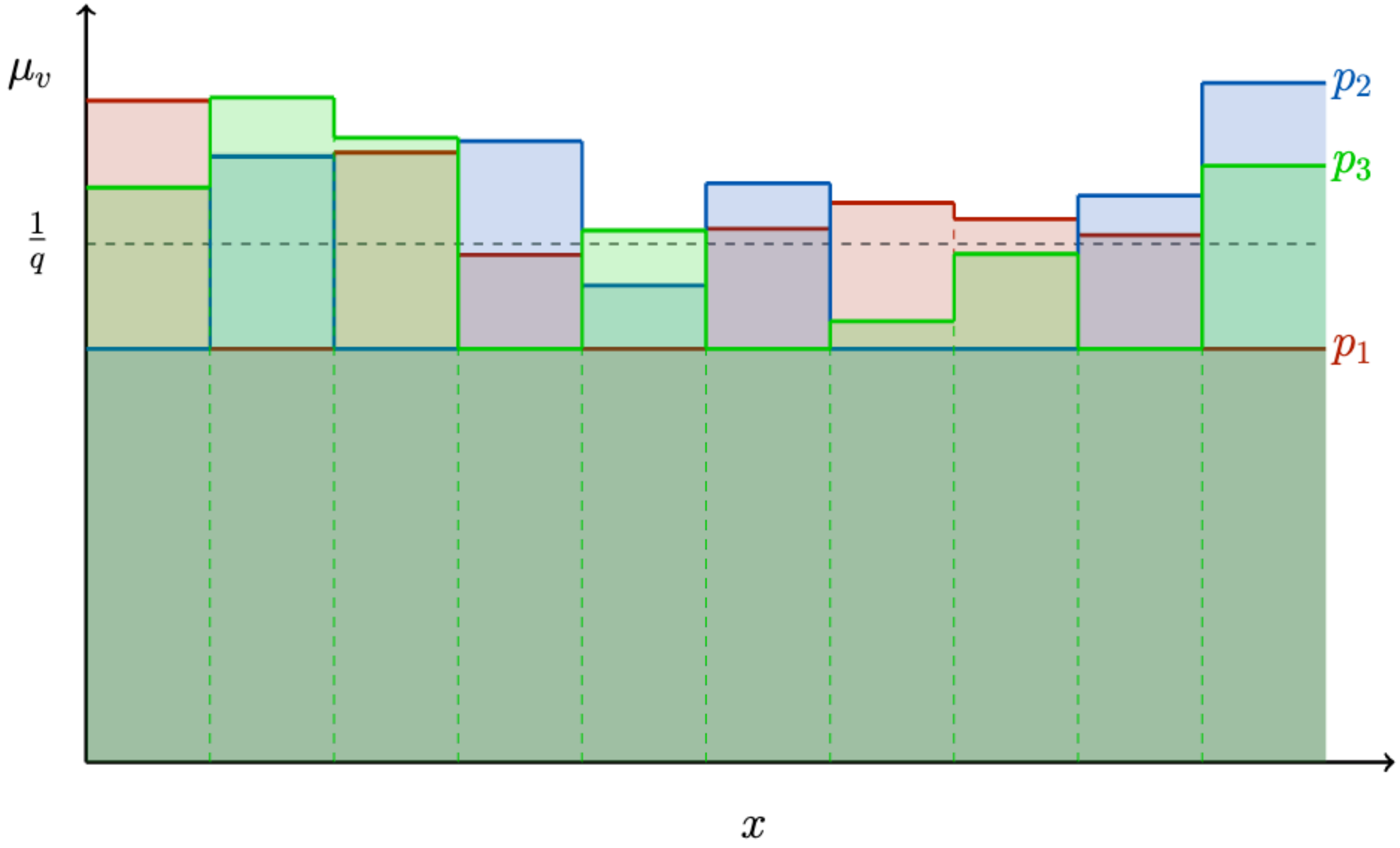
μ_v: marginal distribution at $v \in V$

[Haeupler, Saha, Srinivasan '11]:

LLL condition $\Longrightarrow \mu_v \geq \theta$

where $\theta = (1 - o(1))\dfrac{1}{q}$

Local Uniformity



μ: uniform distribution over solutions

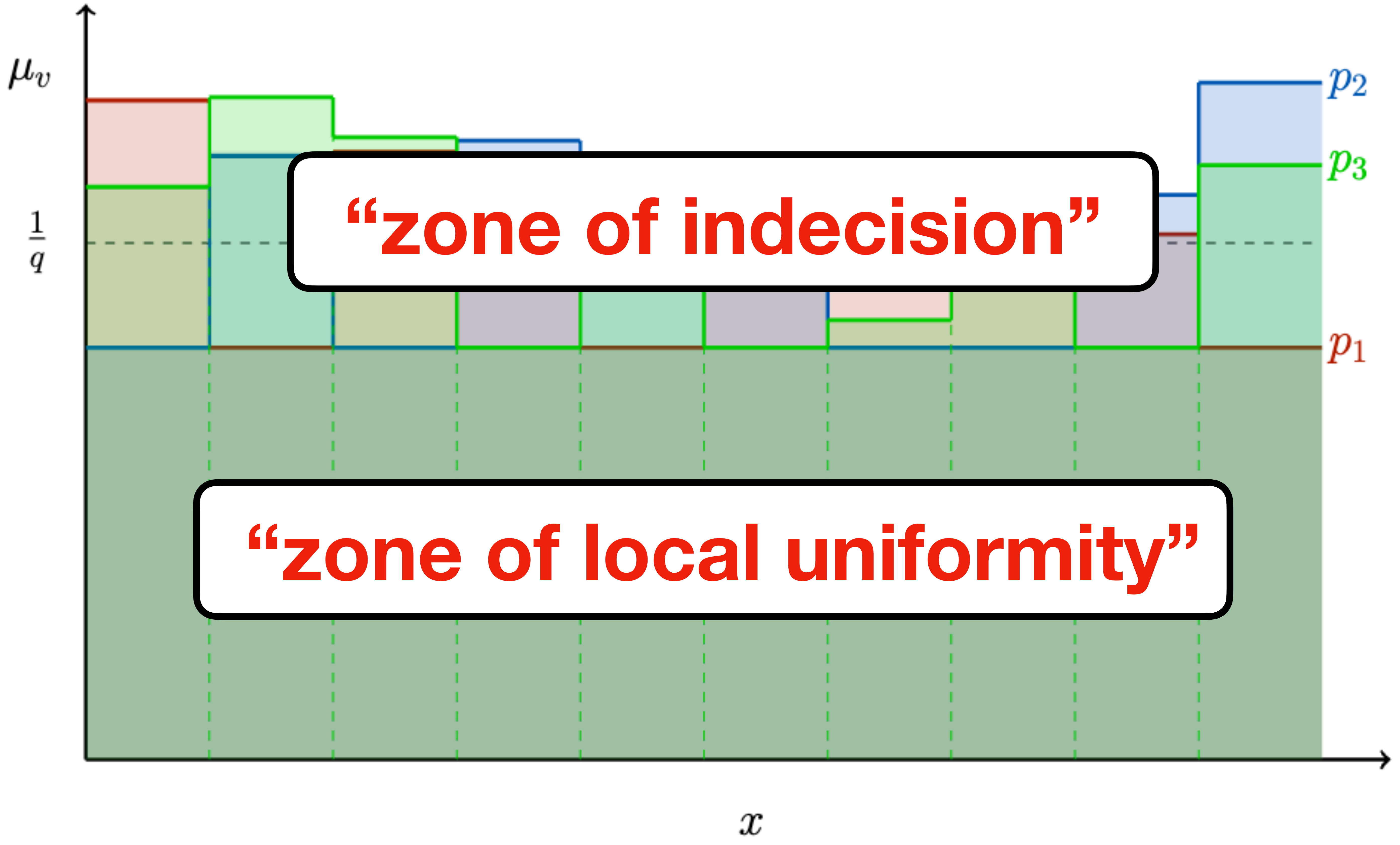
μ_v: marginal distribution at $v \in V$

[Haeupler, Saha, Srinivasan '11]:

LLL condition $\Longrightarrow \mu_v \geq \theta$

where $\theta = (1 - o(1))\dfrac{1}{q}$

Local Uniformity



μ: uniform distribution over solutions

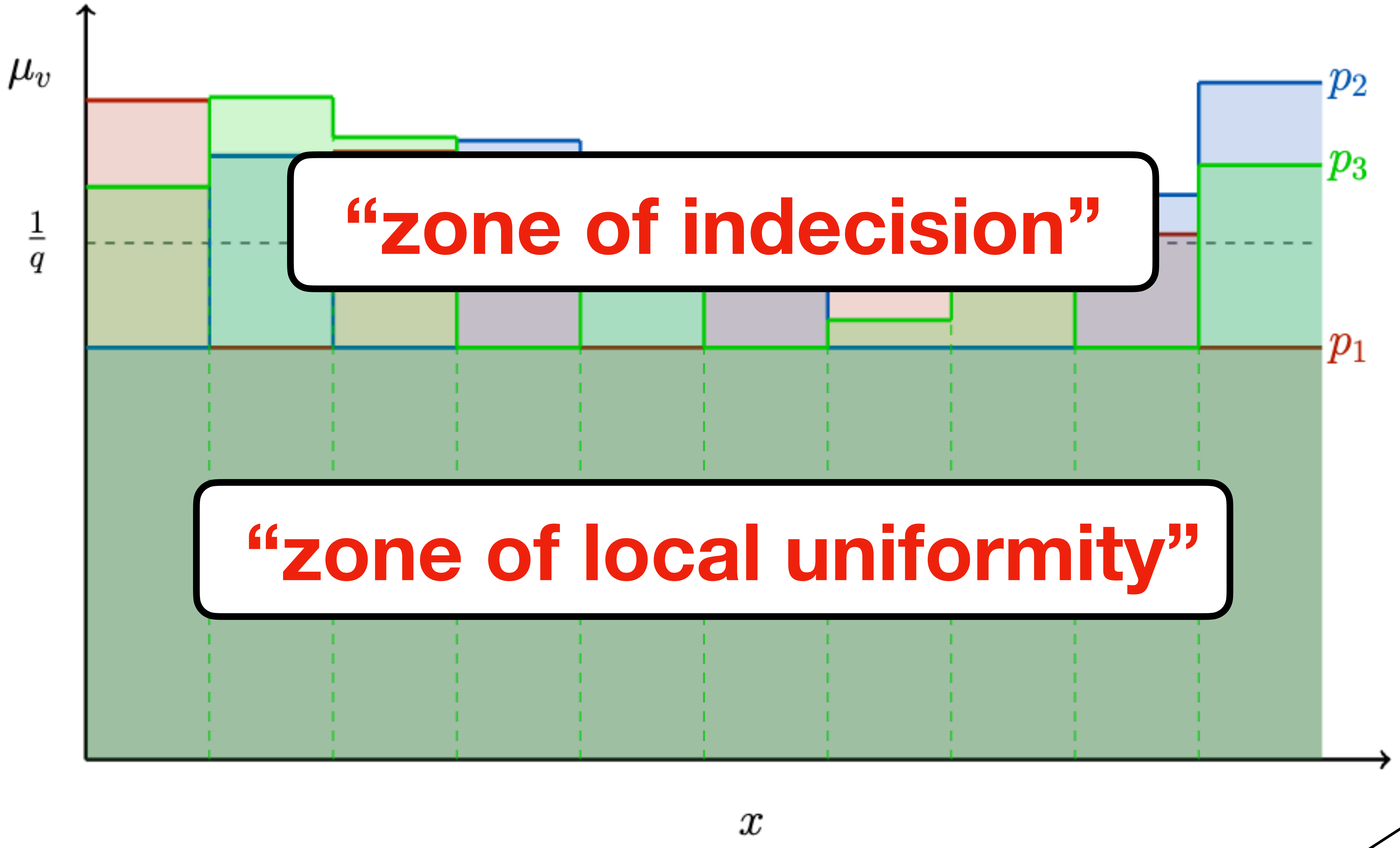
μ_v: marginal distribution at $v \in V$

[Haeupler, Saha, Srinivasan '11]:

LLL condition $\Longrightarrow \mu_v \geq \theta$

where $\theta = (1 - o(1))\dfrac{1}{q}$

Local Uniformity



μ: uniform distribution over solutions

μ_v: marginal distribution at $v \in V$

[Haeupler, Saha, Srinivasan ’11]:

LLL condition $\Longrightarrow \mu_v \geq \theta$

where $\theta = (1 - o(1))\dfrac{1}{q}$

$$\mu_v = q\theta \cdot \mathscr{U} + (1 - q\theta) \cdot \mathscr{D}_v$$

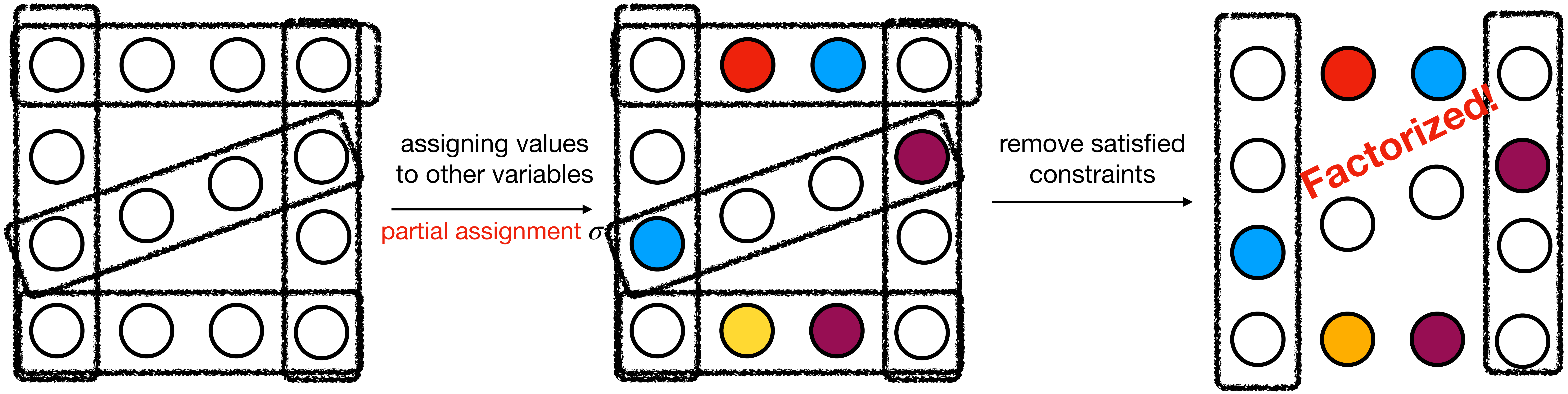
$\mathscr{U}$: uniform over $[q]$

“Overflow distribution”

$$\mathscr{D}_v(x) = \frac{\mu_v(x) - \theta}{1 - q\theta}$$

Factorization property

A key observation: assigning values to other variables may help factorize the formula!



When the connected component containing v is logarithmically small, we can use exhaustive enumeration to calculate μ_v^σ and $\mathscr{D}_v^\sigma$

A recursive marginal approximator

$$\mu_v^\sigma = q\theta \cdot \mathcal{U} + (1 - q\theta) \cdot \mathcal{D}_v^\sigma$$

To approximate μ_v^σ:

approximate $\mathcal{D}_v^\sigma$

$$\mathcal{D}_v^\sigma = \sum_{a \in [q]} \left(\mu_u^\sigma(a) \cdot \mathcal{D}_v^{\sigma_{u \leftarrow a}} \right)$$

To approximate $\mathcal{D}_v^\sigma$:

(1) If the formula is factorized with respect to v, use exhaustive enumeration to calculate $\mathcal{D}_v^\sigma$

(2) Otherwise, choose another variable u and recursively calculate μ_u^σ and $\mathcal{D}_v^{\sigma_{u \leftarrow a}}$ for $\mathcal{D}_v^\sigma$

choose wisely!

μ: uniform distribution over solutions

μ_v^σ: marginal distribution at $v \in V$

conditioning on partial assignment σ

$$\mathcal{D}_v^\sigma \triangleq \frac{\mu_v^\sigma - q\theta}{1 - q\theta}$$

$\sigma_{u \leftarrow a}$: The extended partial assignment

after assigning a to u

A recursive marginal approximator

$$\mu_v^\sigma = q\theta \cdot \mathcal{U} + (1 - q\theta) \cdot \mathcal{D}_v^\sigma$$

To approximate μ_v^σ:

approximate $\mathcal{D}_v^\sigma$

$$\mathcal{D}_v^\sigma = \sum_{a \in [q]} \left(\mu_u^\sigma(a) \cdot \mathcal{D}_v^{\sigma_{u \leftarrow a}} \right)$$

To approximate $\mathcal{D}_v^\sigma$:

(1) If the formula is factorized with respect to v, use exhaustive enumeration to calculate $\mathcal{D}_v^\sigma$

(2) Otherwise, choose another variable u and recursively calculate μ_u^σ and $\mathcal{D}_v^{\sigma_{u \leftarrow a}}$ for $\mathcal{D}_v^\sigma$

choose wisely!

μ: uniform distribution over solutions

μ_v^σ: marginal distribution at $v \in V$ conditioning on partial assignment σ

$$\mathcal{D}_v^\sigma \triangleq \frac{\mu_v^\sigma - q\theta}{1 - q\theta}$$

$\sigma_{u \leftarrow a}$: The extended partial assignment after assigning a to u

chain rule→correctness!

A recursive marginal approximator

$$\mu_v^\sigma = q\theta \cdot \mathscr{U} + (1 - q\theta) \cdot \mathscr{D}_v^\sigma$$

To approximate μ_v^σ:

approximate $\mathscr{D}_v^\sigma$ (a decay in error)

$$\mathscr{D}_v^\sigma = \sum_{a \in [q]} \left(\mu_u^\sigma(a) \cdot \mathscr{D}_v^{\sigma_{u \leftarrow a}} \right)$$

To approximate $\mathscr{D}_v^\sigma$:

(1) If the formula is factorized with respect to v, use exhaustive enumeration to calculate $\mathscr{D}_v^\sigma$

(2) Otherwise, choose another variable u and recursively calculate μ_u^σ and $\mathscr{D}_v^{\sigma_{u \leftarrow a}}$ for $\mathscr{D}_v^\sigma$

μ: uniform distribution over solutions

μ_v^σ: marginal distribution at $v \in V$

conditioning on partial assignment σ

$$\mathscr{D}_v^\sigma \triangleq \frac{\mu_v^\sigma - q\theta}{1 - q\theta}$$

$\sigma_{u \leftarrow a}$: The extended partial assignment after assigning a to u

Caveat:

“bad” assignments exist

A recursive marginal approximator

$$\mu_v^\sigma = q\theta \cdot \mathcal{U} + (1 - q\theta) \cdot \mathcal{D}_v^\sigma$$

To approximate μ_v^σ:

approximate $\mathcal{D}_v^\sigma$ (a decay in error)

$$\mathcal{D}_v^\sigma = \sum_{a \in [q]} \left(\mu_u^\sigma(a) \cdot \mathcal{D}_v^{\sigma_{u \leftarrow a}} \right)$$

To approximate $\mathcal{D}_v^\sigma$:

(1) If the formula is factorized with respect to v, use exhaustive enumeration to calculate $\mathcal{D}_v^\sigma$

(2) Otherwise, choose another variable u and recursively calculate μ_u^σ and $\mathcal{D}_v^{\sigma_{u \leftarrow a}}$ for $\mathcal{D}_v^\sigma$

μ: uniform distribution over solutions

μ_v^σ: marginal distribution at $v \in V$ conditioning on partial assignment σ

$$\mathcal{D}_v^\sigma \triangleq \frac{\mu_v^\sigma - q\theta}{1 - q\theta}$$

$\sigma_{u \leftarrow a}$: The extended partial assignment after assigning a to u

Caveat:

"bad" assignments exist

Solution:

truncate properly so that

(1) exhaustive enumeration is efficient

(2) truncation error can be well-controlled

A recursive marginal approximator

$$\mu_v^\sigma = q\theta \cdot \mathscr{U} + (1 - q\theta) \cdot \mathscr{D}_v^\sigma$$

To approximate μ_v^σ:

approximate $\mathscr{D}_v^\sigma$ (a decay in error)

$$\mathscr{D}_v^\sigma = \sum_{a \in [q]} \left(\mu_u^\sigma(a) \cdot \mathscr{D}_v^{\sigma_{u \leftarrow a}} \right)$$

To approximate $\mathscr{D}_v^\sigma$:

(1) If the formula is factorized with respect to v, use exhaustive enumeration to calculate $\mathscr{D}_v^\sigma$

(2) Otherwise, choose another variable u and recursively calculate μ_u^σ and $\mathscr{D}_v^{\sigma_{u \leftarrow a}}$ for $\mathscr{D}_v^\sigma$

μ: uniform distribution over solutions

μ_v^σ: marginal distribution at $v \in V$ conditioning on partial assignment σ

$$\mathscr{D}_v^\sigma \triangleq \frac{\mu_v^\sigma - q\theta}{1 - q\theta}$$

$\sigma_{u \leftarrow a}$: The extended partial assignment after assigning a to u

Caveat:
LLL condition
not self-reducible!

A recursive marginal approximator

$$\mu_v^\sigma = q\theta \cdot \mathscr{U} + (1 - q\theta) \cdot \mathscr{D}_v^\sigma$$

To approximate μ_v^σ:

approximate $\mathscr{D}_v^\sigma$ (a decay in error)

$$\mathscr{D}_v^\sigma = \sum_{a \in [q]} \left(\mu_u^\sigma(a) \cdot \mathscr{D}_v^{\sigma_{u \leftarrow a}} \right)$$

To approximate $\mathscr{D}_v^\sigma$:

(1) If the formula is factorized with respect to v, use exhaustive enumeration to calculate $\mathscr{D}_v^\sigma$

(2) Otherwise, choose another variable u and recursively calculate μ_u^σ and $\mathscr{D}_v^{\sigma_{u \leftarrow a}}$ for $\mathscr{D}_v^\sigma$

μ: uniform distribution over solutions

μ_v^σ: marginal distribution at $v \in V$

conditioning on partial assignment σ

$$\mathscr{D}_v^\sigma \triangleq \frac{\mu_v^\sigma - q\theta}{1 - q\theta}$$

$\sigma_{u \leftarrow a}$: The extended partial assignment after assigning a to u

Caveat:
LLL condition
not self-reducible!

Solution:

freezing[Beck' 91][JPV' 21b]

$\Pr_{\mathscr{P}}[\neg c \mid X] > \alpha \implies$ "freeze" c!

Marginal Approximator

MarginalApproximator(σ, v): approximates μ_v^σ

$\hat{\mathscr{D}}_v^\sigma \leftarrow$ ***RecursiveApproximator***(σ, v);

return $q\theta \cdot \mathscr{U} + (1 - q\theta) \cdot \hat{\mathscr{D}}_v^\sigma$

local uniformity
[Haeupler, Saha, Srinivasan '11]:

worst-case LLL cond. $\Longrightarrow \mu_v^\sigma \geq \theta$
where $\theta = (1 - o(1))1/q$

RecursiveApproximator(σ, v): approximates $(\mu_v^\sigma - \theta)/(1 - q\theta)$

If some truncation condition is reached

return an arbitrary distribution on $[q]$;

No linear program involved!

If $u = \mathrm{NextVar}(\sigma, v) \neq \perp$

$\hat{\mu}_u^\sigma \leftarrow$ ***MarginalApproximator***(σ, v), $\hat{\mathscr{D}}_v^{\sigma_{u \leftarrow a}} \leftarrow$ ***RecursiveApproximator***(σ, v) for each $a \in [q]$;

return $\sum_{a \in [q]} \left(\hat{\mu}_u^\sigma(a) \cdot \hat{\mathscr{D}}_v^{\sigma_{u \leftarrow a}} \right)$

Compute and return $\mathscr{D}_v^\sigma$ using exhaustive enumeration

$\mathrm{NextVar}$: subroutine for choosing the next variable to assign

(informal) boundary variable over connected component of frozen constraints containing v

The main algorithm

1. $\frac{|\mathcal{S}_{P_i}|}{|\mathcal{S}_{P_{i-1}}|}$ can be well-approximated
2. $|\mathcal{S}_{P_s}|$ can be efficiently enumerated

Main Algorithm (sketch):

Find a "good" sequence of partial assignments $P_0, P_1, \ldots, P_s$ s.t. P_0 is empty and P_i extends P_{i-1} on some unassigned variable, deterministically;

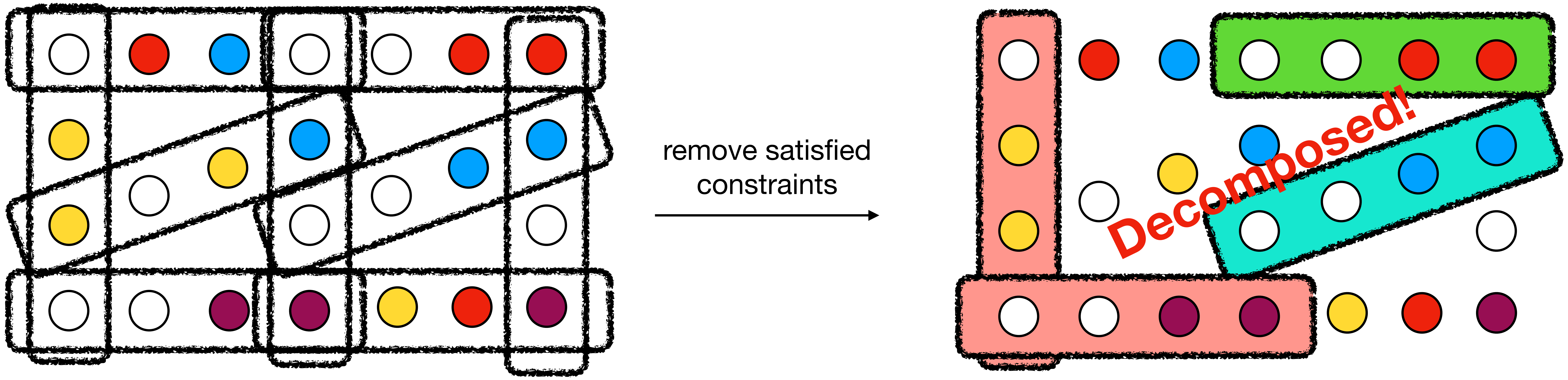
Approximate $\frac{|\mathcal{S}_{P_s}|}{|\mathcal{S}_{P_0}|}$ as a telescopic product of marginal distributions;

Calculate $|\mathcal{S}_{P_s}|$ using exhaustive enumeration, yielding $|\mathcal{S}_{P_0}|$.

"guiding assignment" in [JPV' 21b]

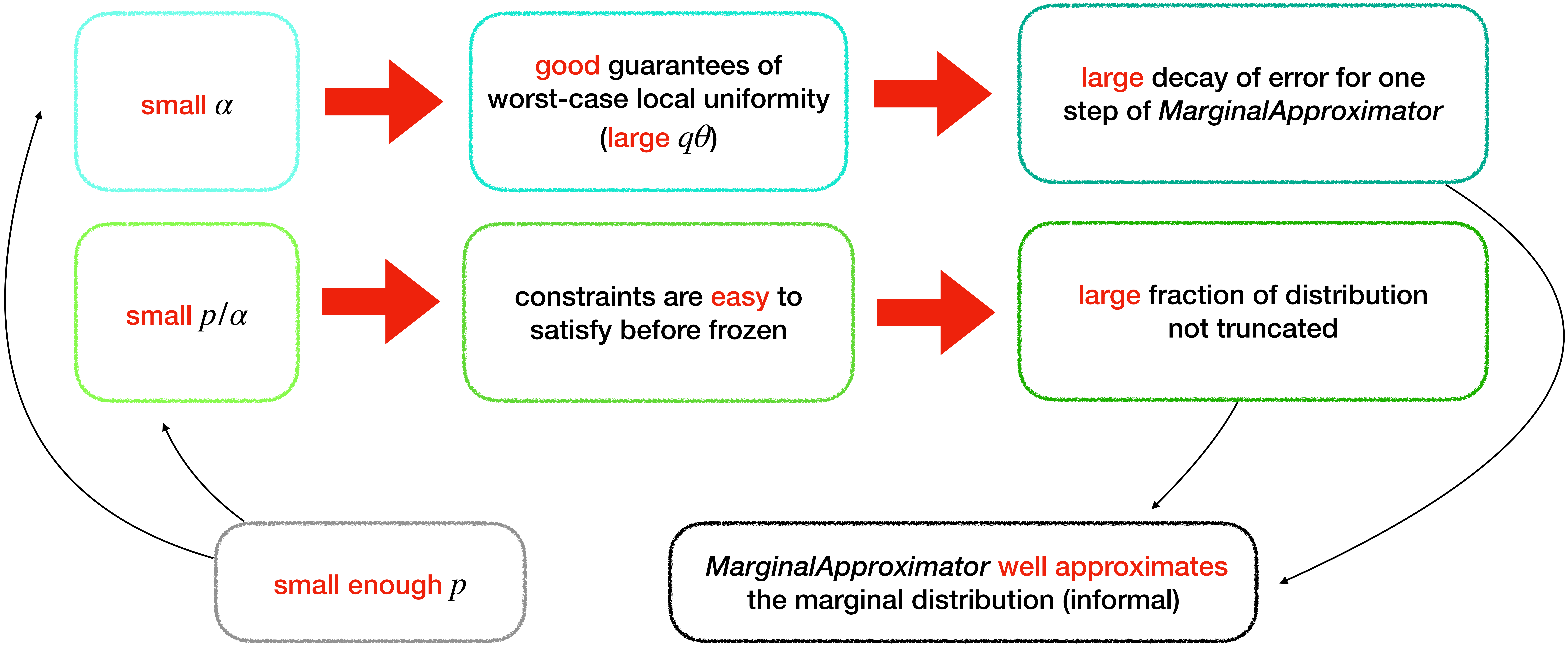
Method of conditional expectation

$\mathcal{S}_\sigma$: set of satisfying solutions extending σ

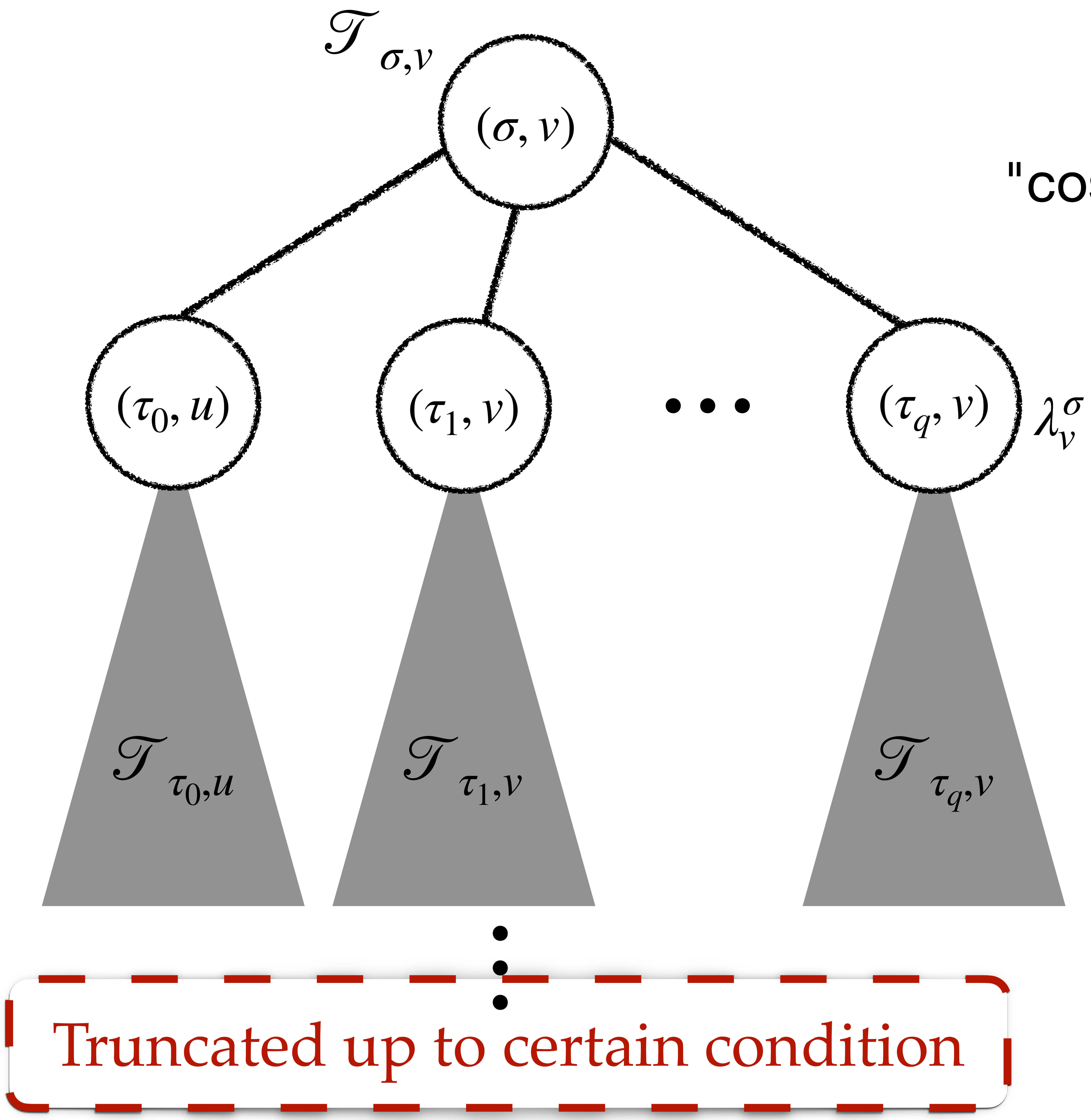


The freezing threshold α

and how it relates to approximation error of MarginalApproximator



Tree recursion of approximation error



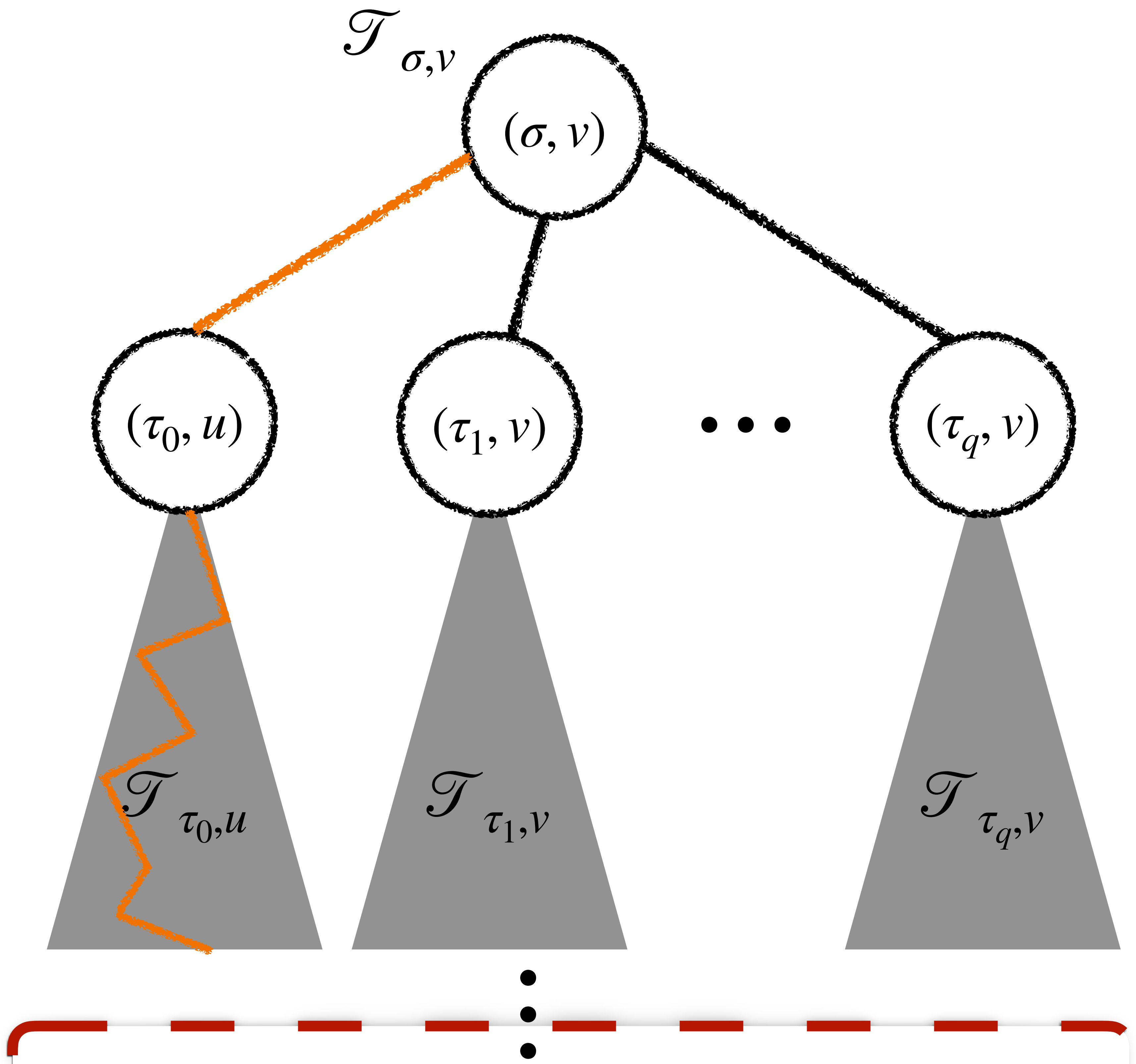
"cost function" λ_v^σ:

$$\lambda_v^\sigma = \begin{cases} 0 & \mathrm{NextVar}(\sigma, v) = \perp \\ 1 & (\sigma, v) \text{ is truncated} \\ (1 - q\theta)\lambda_u^{\tau_0} + \sum_{a \in [q]} \left(\mu_u^\sigma(a) \cdot \lambda_v^{\tau_a}\right) & \text{otherwise} \end{cases}$$

$$d_{\mathrm{TV}}(\mathscr{D}_v^\sigma, \hat{\mathscr{D}}_v^\sigma) \leq \lambda_v^\sigma$$

$\hat{\mathscr{D}}_v^\sigma$: distribution of ***RecursiveApproximator***(σ, v)

Decay of approximation error



Truncated up to certain condition

Percolation-style analysis

Analyze truncated root-to-leaf paths

Witness argument ($p\Delta^7 \lesssim 1$ bound)

A truncated path → a large $\{2,3\}$-tree [Alon' 91] with ind. bad events

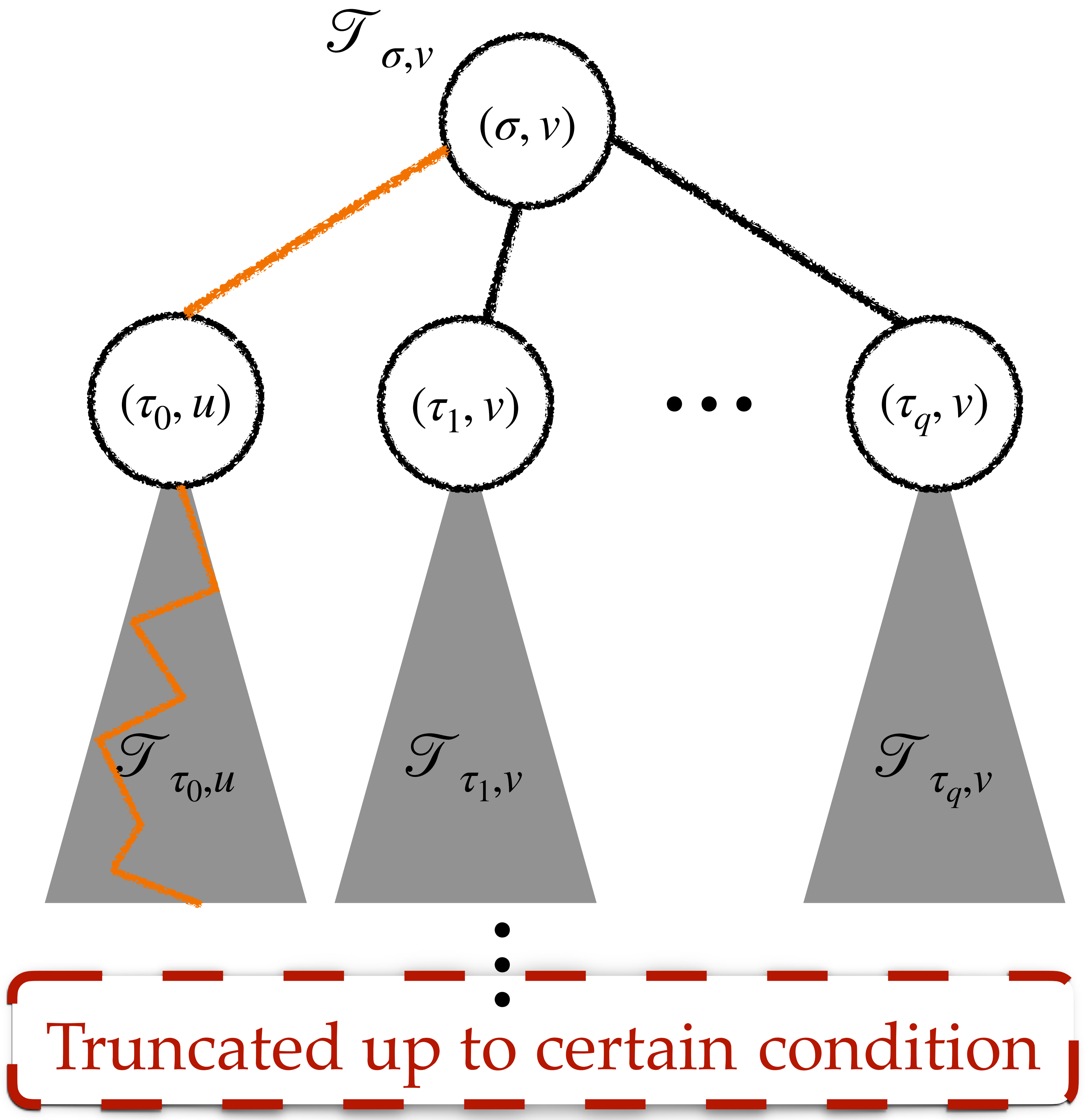
Bad events

Type 1: $(\sigma, v) \to (\tau_0, u)$ (fraction $q\theta = O(\Delta^{-3})$)

Type 2: $c \in \mathscr{C}$ is frozen (fraction roughly $\alpha/p = O(\Delta^{-3})$)

$$\lambda_v^\sigma = \begin{cases} 0 & \mathrm{NextVar}(\sigma, v) = \perp \\ 1 & (\sigma, v) \text{ is truncated} \\ (1 - q\theta)\lambda_u^{\tau_0} + \sum_{a \in [q]} \left(\mu_u^\sigma(a) \cdot \lambda_v^{\tau_a}\right) & \text{otherwise} \end{cases}$$

Decay of approximation error



Percolation-style analysis

Analyze truncated root-to-leaf paths

Witness argument ($p\Delta^7 \lesssim 1$ bound)

A truncated path → a large $\{2,3\}$-tree [Alon' 91] with ind. bad events

Bad events

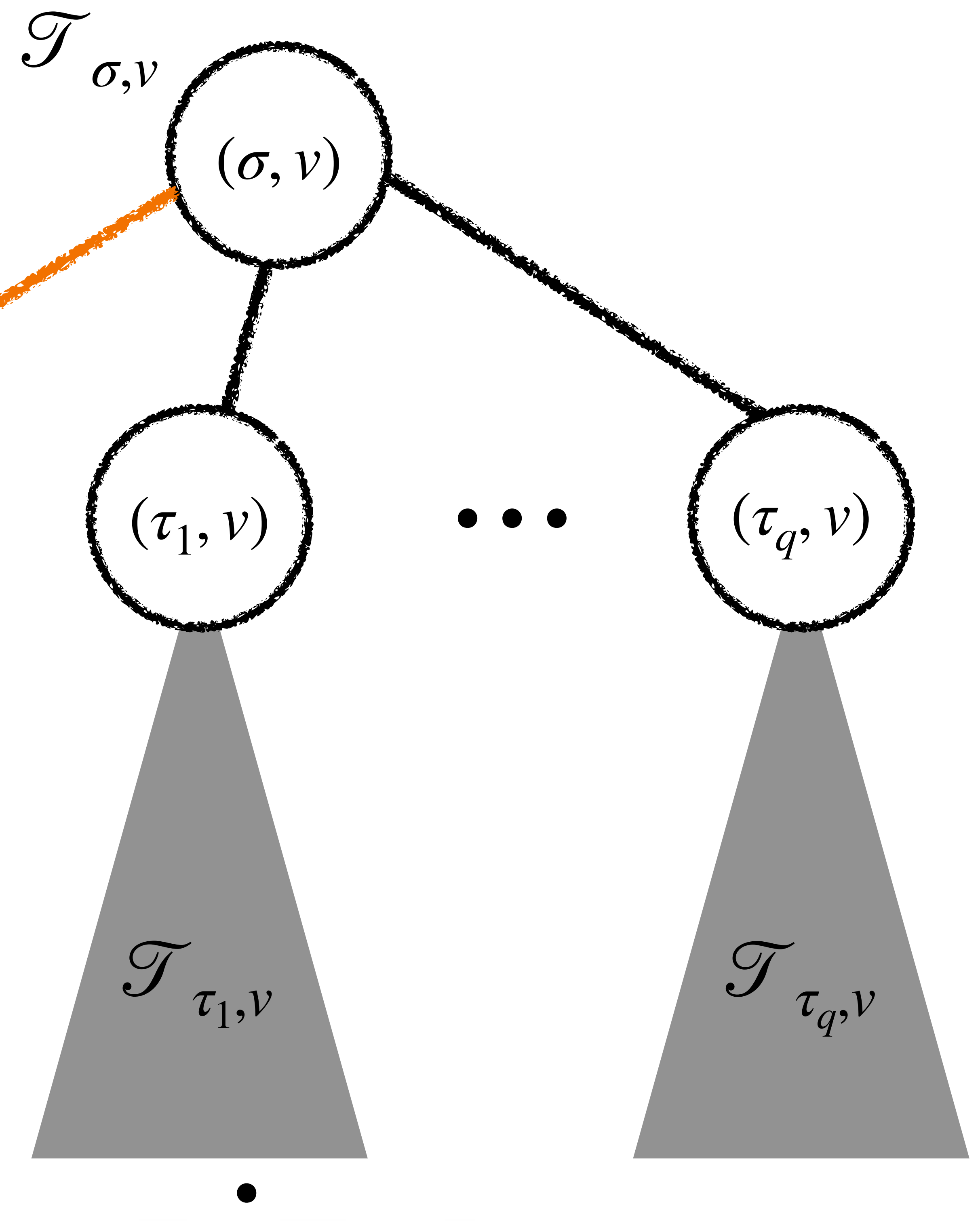
Type 1: $(\sigma, v) \to (\tau_0, u)$ (fraction $q\theta = O(\Delta^{-3})$) **(variable)**

Type 2: $c \in \mathscr{C}$ is frozen (fraction roughly $\alpha/p = O(\Delta^{-3})$) **(constraint)**

$$\lambda_v^\sigma = \begin{cases} 0 & \mathrm{NextVar}(\sigma, v) = \perp \\ 1 & (\sigma, v) \text{ is truncated} \\ (1 - q\theta)\lambda_u^{\tau_0} + \sum_{a \in [q]} \left(\mu_u^\sigma(a) \cdot \lambda_v^{\tau_a}\right) & \text{otherwise} \end{cases}$$

Different densities!

Decay of approximation error



Truncated up to certain condition

Percolation-style analysis

Analyze truncated root-to-leaf paths

Witness argument ($p\Delta^5 \lesssim 1$ bound)

A truncated path

a large generalized $\{2,3\}$-tree
with ind. bad events

Bad events

Type 1: $(\sigma, v) \to (\tau_0, u)$ (fraction $q\theta = O(\Delta^{-1})$) **(variable)**

Type 2: $c \in \mathscr{C}$ is frozen (fraction roughly $\alpha/p = O(\Delta^{-3})$) **(constraint)**

$$\lambda_v^\sigma = \begin{cases} 0 & \mathrm{NextVar}(\sigma, v) = \perp \\ 1 & (\sigma, v) \text{ is truncated} \\ (1 - q\theta)\lambda_u^{\tau_0} + \sum_{a\in[q]} \left(\mu_u^\sigma(a) \cdot \lambda_v^{\tau_a}\right) & \text{otherwise} \end{cases}$$

Different densities!

Generalized $\{2,3\}$-tree

$H = (V, \mathscr{E})$: a hypergraph Lin(H): the line graph of H

dist$_{\mathrm{Lin}(H)}$:shortest path distance in Lin(H)

$T \subseteq \mathscr{E}$ is a $\{2,3\}$-tree of Lin(H) if:

- for any distinct $u, v \in T$, dist$_{\mathrm{Lin}(H)}(u, v) \geq 2$;
- T is connected if an edge is added between every $u, v \in T$ such that dist$_{\mathrm{Lin}(H)}(u, v) \in \{2,3\}$.

$T = U \cup E$, where $U \subseteq V, E \subseteq \mathscr{E}$, is a generalized $\{2,3\}$-tree of H if:

- for any distinct $u, v \in E$, dist$_{\mathrm{Lin}(H)}(u, v) \geq 2$;
- It holds for the directed graph $G(T, \mathscr{A})$ that there is a vertex $r \in T$ (called a root) which can reach all other vertices through directed paths, where the $G(T, \mathscr{A})$ is constructed on the vertex set T as that, for any $u, v \in T$ there is an arc $u, v \in \mathscr{A}$ if and only if at least one of the following conditions is satisfied:
  - $u, v \in E$ and dist$_{\mathrm{Lin}(H)}(u, v) \in \{2,3\}$;
  - $u \in U, v \in E$ and there exists $e \in \mathscr{E}$ such that $u \in e \wedge$ dist$_{\mathrm{Lin}(H)}(v, e) = 1$;
  - $u \in E, v \in U$ and there exists $e \in \mathscr{E}$ such that $u \in v \wedge$ dist$_{\mathrm{Lin}(H)}(u, e) \in \{1,2\}$;
  - $u, v \in U$ and there exists $e \in \mathscr{E}$ such that $u, v \in e$.

Summary

We develop a new approach for deterministic approximate counting general CSP solutions in the LLL regime by derandomizing the recursive sampler in [He, **W.**, Yin '22].

We invent a refined combinatorial structure of generalized $\{2,3\}$-tree, leading to a state-of-the-art regime $p\Delta^5 \lesssim 1$ for counting/sampling general CSP solutions.

Summary

We develop a new approach for deterministic approximate counting general CSP solutions in the LLL regime by derandomizing the recursive sampler in [He, **W.**, Yin '22].

We invent a refined combinatorial structure of generalized $\{2,3\}$-tree, leading to a state-of-the-art regime $p\Delta^5 \lesssim 1$ for counting/sampling general CSP solutions.

Future work

- A better LLL condition? (Current $p\Delta^5 \lesssim 1$ versus lower bound $p\Delta^2 \gtrsim 1$)
- Derandomizing more general methods for sampling LLL, such as MCMC? (Resolved by [Feng, Guo, **W.**, Wang, Yin '22])

Summary

We develop a new approach for deterministic approximate counting general CSP solutions in the LLL regime by derandomizing the recursive sampler in [He, **W.**, Yin '22].

We invent a refined combinatorial structure of generalized $\{2,3\}$-tree, leading to a state-of-the-art regime $p\Delta^5 \lesssim 1$ for counting/sampling general CSP solutions.

Thanks! Any questions?

Future work

- A better LLL condition? (Current $p\Delta^5 \lesssim 1$ versus lower bound $p\Delta^2 \gtrsim 1$)
- Derandomizing more general methods for sampling LLL, such as MCMC?
  (Resolved by [Feng, Guo, **W.**, Wang, Yin '22])