Dynamic Sampling from Graphical Models

Weiming Feng

Nanjing University

Joint work with: Nisheeth K. Vishnoi (Yale University)

Yitong Yin (Nanjing University)

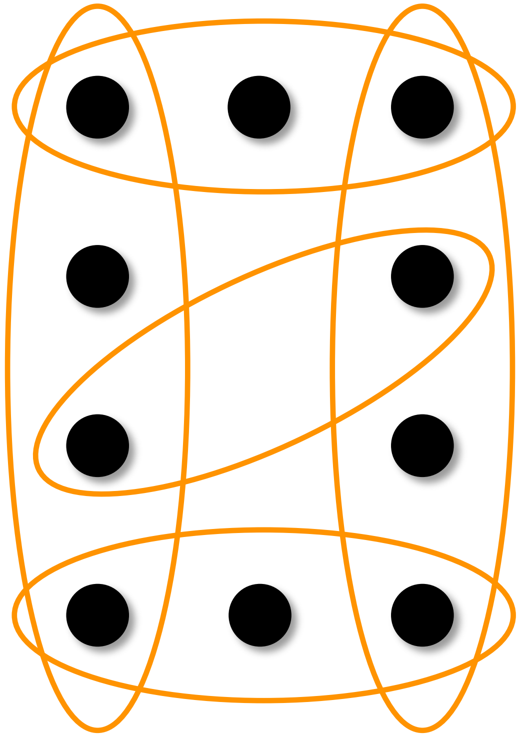
STOC 2019

Phoenix, AZ.

Graphical Model

- Hyper graph $H = (V, E)$
  - V: vertices
  - $E \subseteq 2^V$: hyper edges.
- Vertex: **variable** with domain Q.
- Hyper edge: **constraint** on its variables.
- **Weight functions(factors)**: $\Phi = (\phi_v)_{v \in V} \cup (\phi_e)_{e \in E}$
  - each variable $\phi_v: Q \to \mathbb{R}_{\geq 0}$;
  - each constraint $\phi_e: Q^e \to \mathbb{R}_{\geq 0}$.
- Each **configuration** $\sigma \in Q^V$: its **weight**

$$w(\sigma) = \prod_{v \in V} \phi_v(\sigma_v) \prod_{e \in E} \phi_e(\sigma_e).$$



hyper graph
$H = (V, E)$

Graphical Model

Instance $\mathcal{I} = (V, E, Q, \Phi)$

- V: **variables**
- E: **constraints**
- Q: **domain**
- $\Phi = (\phi_v)_{v \in V} \cup (\phi_e)_{e \in E}$: **weight functions (factors)**

Gibbs distribution μ over Q^V:

$$\forall \sigma \in Q^V: \ \mu(\sigma) \propto w(\sigma) = \prod_{v \in V} \phi_v(\sigma_v) \prod_{e \in E} \phi_e(\sigma_e)$$

Hardcore Model

- Graph $G = (V, E)$

$$I(G) = \{\text{independent sets in } G\}.$$

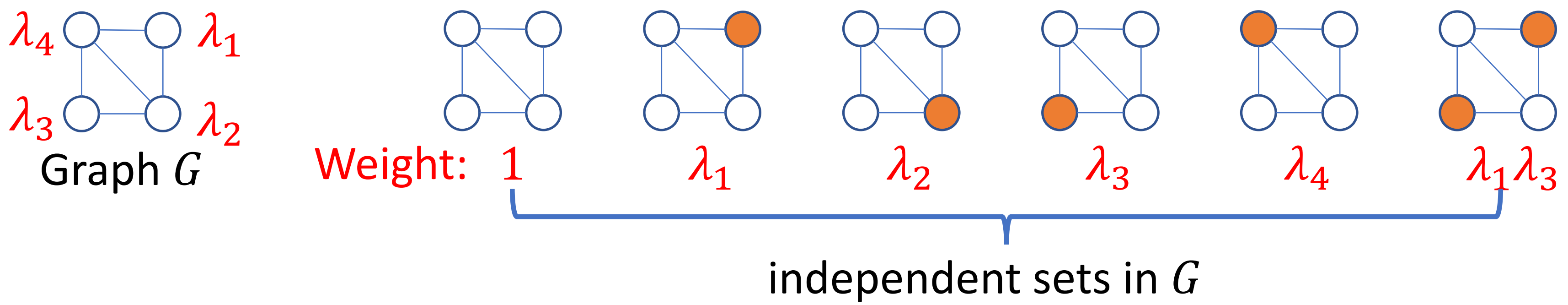
- **Fugacity** of vertex $v \in V$: $\lambda_v \in \mathbb{R}_{\geq 0}$.
- **Weight** of independent set $S \in I(G)$:

$$w(S) = \prod_{v \in S} \lambda_v .$$

product of vertex fugacities

- **Hardcore model**: distribution μ over $I(G)$, each $S \in I(G)$:

$$\mu(S) \propto w(S).$$



Ising Model

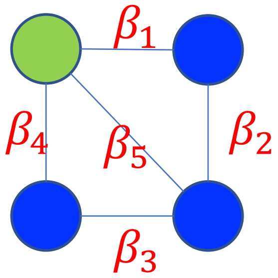
- Graph $G = (V, E)$.
- **Inverse temperature** of edge $e \in E$: $\beta_e \in \mathbb{R}_{\geq 0}$.
- **Spin state** of vertex $v \in V$: $\{-1, +1\}$.
- **Weight** of configuration $\sigma \in \{-1, +1\}^V$

$$w(\sigma) = \prod_{e=\{u,v\}\in E} \exp(\beta_e \sigma_u \sigma_v).$$

product of pairwise interactions

- **Ising model**: distribution μ over $\{-1, +1\}^V$:

$$\mu(\sigma) \propto w(\sigma).$$



Weight=$\exp(-\beta_1)\exp(\beta_2)\exp(\beta_3)\exp(-\beta_4)\exp(-\beta_5)$.

Graphical Model

- **Machine Learning**
  representation, inference, learning;
- **Statistical Physics**
  Ising model, hardcore model;
- **Theoretical Computer Science**
  sampling, counting.

application: image denoising

Sampling from Graphical Model

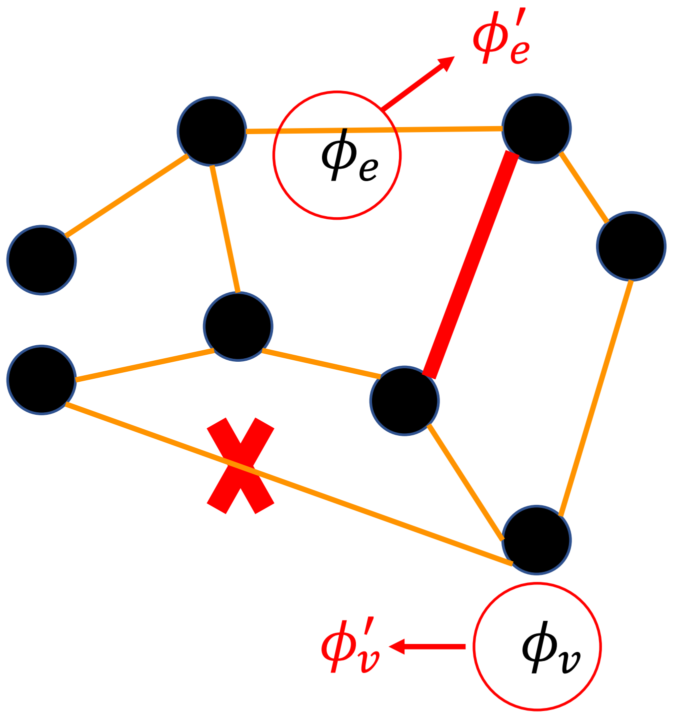
- **Input**: a graphical model $\mathcal{I}$;
- **Output**: a sample $\boldsymbol{X} \sim \mu_{\mathcal{I}}$.

Dynamic Sampling Problem

- Graphical model $\mathcal{I} = (V, E, Q, \Phi)$

$$\mu_{\mathcal{I}}(\sigma) \propto \prod_{v \in V} \phi_v(\sigma_v) \prod_{e \in E} \phi_e(\sigma_e).$$

- Random sample: $\boldsymbol{X} \sim \mu_{\mathcal{I}}$.
- Updates of graphical model $\mathcal{I} \to \mathcal{I}'$
  - add/delete constraints;
  - change weight functions.



Question: Can we modify $\boldsymbol{X}$ to $\boldsymbol{X}' \sim \mu_{\mathcal{I}'}$ with a ***small incremental cost***?

random sample for **updated** graphical model

Update is **represented** by a pair (D, Φ_D)

- $D \subseteq V \cup 2^V$: **updated variables & updated constraints**;
- $\Phi_D = (\phi_a)_{a \in D}$: **new weight functions**.

input graphical model

$\mathcal{I} = (V, E, Q, \Phi)$

update (D, Φ_D)

updated graphical model

$\mathcal{I}' = (V, E', Q, \Phi')$

updated constraints updated weight functions

Dynamic Sampling from Graphical Model

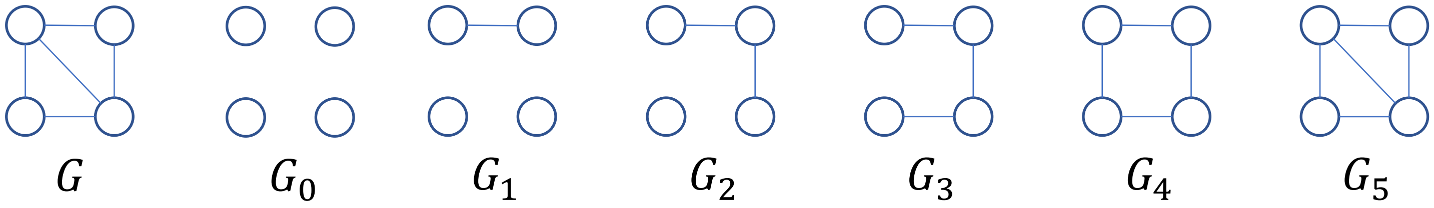
- **Input**: a graphical model $\mathcal{I}$; a sample $\boldsymbol{X} \sim \mu_{\mathcal{I}}$
  an update (D, Φ_D) that modifies $\mathcal{I}$ to $\mathcal{I}'$;
- **Output**: a sample $\boldsymbol{X}' \sim \mu_{\mathcal{I}'}$.

Offline adversary: update is **independent** with the input sample $\boldsymbol{X}$.

Motivations

- **Online learning with dynamic or streaming data**
- **Dynamic graphical models**
  - Video: a sequence of closely related images.

- **Approximate counting** [Jerrum, Valiant, Vazirani, 1986]
  - Graph $G = (V, E)$, count #{**independent sets** of G}.
  - **Self reduction**: **a sequence of graphs** $G_0, G_1, \ldots, G_{|E|}$:



Static Sampling

- **Input**: a graphical model $\mathcal{I}$;
- **Output**: a sample $\boldsymbol{X} \sim \mu_{\mathcal{I}}$.

Dynamic Sampling

- **Input**: a graphical model $\mathcal{I}$;
  a sample $\boldsymbol{X} \sim \mu_{\mathcal{I}}$
  a update (D, Φ_D)
- **Output**: a sample $\boldsymbol{X}' \sim \mu_{\mathcal{I}'}$.

Well studied

Lacking studies

Algorithms for static sampling

- Markov chain Monte Carlo (MCMC)
  - Metropolis Hastings [Metropolis 1953]
  - Glauber Dynamics [Glauber 1963]
- Coupling from the past (CFTP) [Propp and Wilson 1996]

Not suitable for dynamic sampling, per se.

- **can not use** the input sample $\boldsymbol{X}$;
- rerunning sampling algorithm on $\mathcal{I}'$ is **wasteful**.

Our Contribution

New Algorithm →

Dynamic Sampling Problem

- **Input**: a graphical model $\mathcal{I}$;
  a sample $\boldsymbol{X} \sim \mu_{\mathcal{I}}$
  a update (D, Φ_D)
- **Output**: a sample $\boldsymbol{X}' \sim \mu_{\mathcal{I}'}$.

- ***Fast***

  a broad class of graphical models → $\mathbb{E}[\text{running time}] = O(|D|)$
- ***Exact Sampling***

  $\boldsymbol{X}$ follows precisely distribution $\mu_{\mathcal{I}'}$
- ***Las Vegas***

  algorithm knows when to stop
- ***Distributed / Parallel***

  each step uses only local information

Graphical Model

Instance $\mathcal{I} = (V, E, Q, \Phi)$

- V: **variables**
- E: **constraints**
- Q: **domain**
- $\Phi = (\phi_v)_{v \in V} \cup (\phi_e)_{e \in E}$: **weight functions (factors)**

Gibbs distribution μ over Q^V:

$$\forall \sigma \in Q^V: \; \mu(\sigma) \propto w(\sigma) = \prod_{v \in V} \phi_v(\sigma_v) \prod_{e \in E} \phi_e(\sigma_e)$$

Rejection Sampling

Graphical model $\mathcal{I} = (V, E, Q, \Phi)$ with Gibbs distribution

$$\mu_{\mathcal{I}}(\sigma) \propto \prod_{v \in V} \phi_v(\sigma_v) \prod_{e \in E} \phi_e(\sigma_e)\,.$$

Assumption: normalized weighted functions

- each $\phi_v : Q \to [0,1]$ is a **distribution** over Q: $\sum_{c \in Q} \phi_v(c) = 1$;
- each $\phi_e : Q^e \to [0,1]$.

Rejection Sampling

- Each $v \in V$ samples $X_v \sim \phi_v$ ind.;
- Each $e \in E$ becomes **accepted** ind. w.p. $\phi_e(X_e)$; o.w. e becomes **rejected**
- **Accept** $\boldsymbol{X} = (X_v)_{v \in V}$ if all $e \in E$ are accepted;
- **Reject** $\boldsymbol{X}$ if otherwise.

$$\Pr[\boldsymbol{X} = \sigma \wedge \boldsymbol{X} \text{ is accepted}] = \underbrace{\prod_{v \in V} \phi_v(\sigma_v)}_{\text{generate } \boldsymbol{X} = \sigma} \underbrace{\prod_{e \in E} \phi_e(\sigma_e)}_{\text{all } e \in E \text{ are accepted}}.$$

$$\Pr[\text{all } e \in E \text{ are accepted}] = \exp\big(-\Omega(|E|)\big).$$

Rejection Sampling is **Correct** **but** **Slow**.

Question

Can we obtain an **efficient** rejection sampling algorithm ?

- Fast
- Dynamic
- Distributed / Parallel

This problem was **partially solved** by

Partial Rejection Sampling (**PRS**) [Guo, Jerrum, Liu, 2017].

- **Boolean** weight function $\phi_e \to \{0,1\}$
- **Not known** to be dynamic
- **Not** distributed / parallel

Our Contribution

New Algorithm

→

Dynamic Sampling Problem

- **Input**: a graphical model $\mathcal{I}$;
  a sample $\boldsymbol{X} \sim \mu_{\mathcal{I}}$
  a update (D, Φ_D)
- **Output**: a sample $\boldsymbol{X}' \sim \mu_{\mathcal{I}'}$.

Efficient Rejection Sampling

- Fast
- Dynamic
- Distributed/Parallel

Rejection Sampling

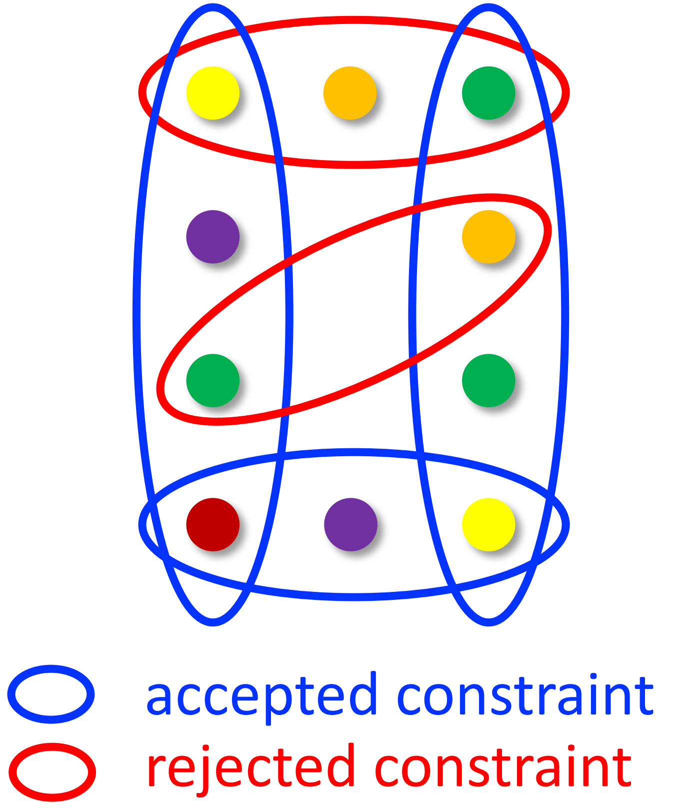
- Each $v \in V$ samples $X_v \sim \phi_v$ ind.;
- Each $e \in E$ becomes **accepted** ind. w.p. $\phi_e(X_e)$; o.w. e becomes **rejected**
- **Accept** $\boldsymbol{X} = (X_v)_{v \in V}$ if all $e \in E$ are accepted;
- **Reject** $\boldsymbol{X}$ if otherwise.

The sample $\boldsymbol{X}$ is **rejected**

Set of **Bad Variables**

$$\mathcal{R} = \bigcup_{\substack{e \in E: \\ e \text{ is rejected}}} e,$$

$\mathcal{R}$: **variables** in **rejected constraints**



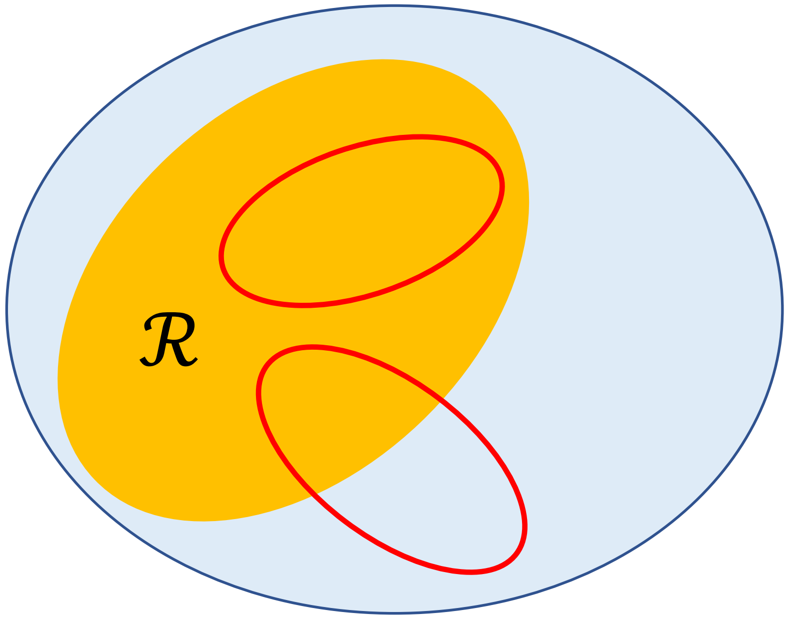
A "Natural" Resampling Algorithm

- each $v \in \mathcal{R}$ resamples $X_v \sim \phi_v$ ind.;
- each $e \in \mathrm{ICD}(\mathcal{R})$ becomes **accepted** ind. w.p. $\phi_e(X_e)$;
- construct new $\mathcal{R} \leftarrow \bigcup_{e \in E : e \text{ is rejected}} e$.

o.w. e becomes **rejected**

$\mathrm{ICD}(\mathcal{R})$: constraints **incident** to $\mathcal{R}$

$$\mathrm{ICD}(\mathcal{R}) = \{ e \in E \mid e \cap \mathcal{R} \neq \emptyset \}.$$



A "Natural" Resampling Algorithm

- each $v \in \mathcal{R}$ resamples $X_v \sim \phi_v$ ind.;
- each $e \in \mathrm{ICD}(\mathcal{R})$ becomes **accepted** ind. w.p. $\phi_e(X_e)$; o.w. e becomes **rejected**
- construct new $\mathcal{R} \leftarrow \bigcup_{e \in E : e \text{ is rejected}} e$.

While($\mathcal{R} \neq \emptyset$)

  Update $(X, \mathcal{R})$ by "Natural" Resampling Algorithm

Output $\boldsymbol{X}$.

Wrong Distribution

- Similar to **Moser-Tardos** for LLL. [Moser, Tardos, 2009]
- The output $\boldsymbol{X}$ does **NOT** follow the Gibbs distribution μ. [Harris, Srinivasan, 2016], [Guo, Jerrum, Liu, 2017]

A "Natural" Resampling Algorithm

- each $v \in \mathcal{R}$ resamples $X_v \sim \phi_v$ ind.;
- each $e \in \mathrm{ICD}(\mathcal{R})$ becomes **accepted** ind. w.p. $\phi_e(X_e)$;
- construct new $\mathcal{R} \leftarrow \bigcup_{e \in E: e \text{ is rejected}} e$.

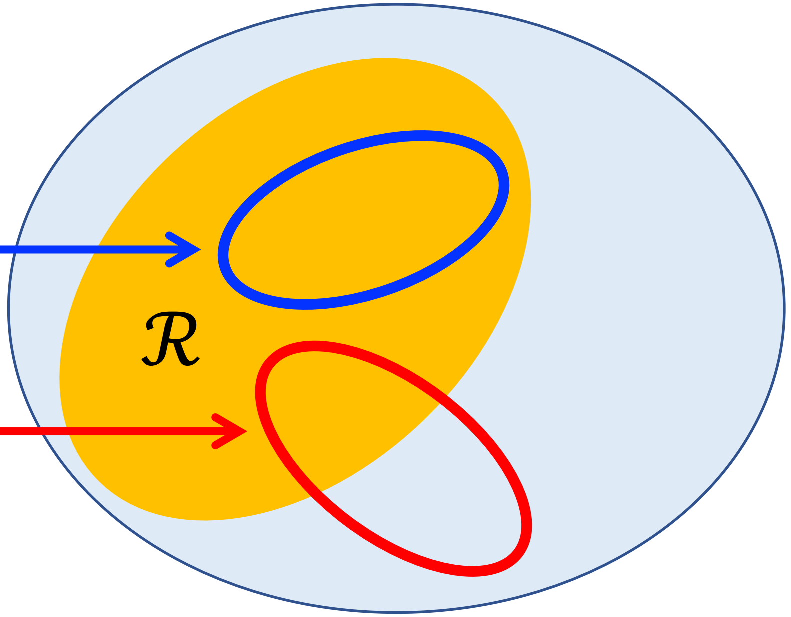
$\mathrm{ICD}(\mathcal{R})$: constraints **incident** to set $\mathcal{R}$

$\mathrm{ICD}(\mathcal{R})$

- **internal constraints**

  $E(\mathcal{R}) = \{ e \in E \mid e \subseteq \mathcal{R} \}$;
- **boundary constraints**

  $\delta(\mathcal{R}) = \{ e \in E \setminus E(\mathcal{R}) \mid e \cap \mathcal{R} \neq \emptyset \}$.



A "Natural" Resampling Algorithm

- each $v \in \mathcal{R}$ resamples $X_v \sim \phi_v$ ind.;
- each $e \in \mathrm{ICD}(\mathcal{R})$ becomes **accepted** ind. w.p. $\phi_e(X_e)$;
- construct new $\mathcal{R} \leftarrow \bigcup_{e \in E: e \text{ is rejected}} e$.

Our Algorithm: Local-Resample$(X, \mathcal{R})$

- each $v \in \mathcal{R}$ resamples $X_v \sim \phi_v$ ind.;
- each $e \in E(\mathcal{R})$ becomes **accepted** ind. w.p. $\phi_e(X_e)$;
- each $e \in \delta(\mathcal{R})$ becomes **accepted** ind. with **a modified probability**;
- construct new $\mathcal{R} \leftarrow \bigcup_{e \in E: e \text{ is rejecte}} e$.
- return $(X, \mathcal{R})$;

Our Algorithm: Local-Resample$(X, \mathcal{R})$

① each $v \in \mathcal{R}$ resamples $X_v \sim \phi_v$ ind.;

② each $e \in E(\mathcal{R})$ becomes **accepted** ind. w.p. $\phi_e(X_e)$;

③ each $e \in \delta(\mathcal{R})$ becomes **accepted** ind. w.p. $C_e \cdot \frac{\phi_e(X_e)}{\phi_e(X_e^{old})} \le 1$;

④ construct new $\mathcal{R} \leftarrow \bigcup_{e \in E: e \text{ is rejected}} e$.

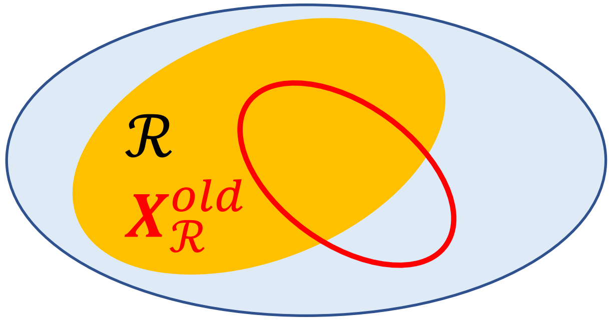
⑤ return $(X, \mathcal{R})$;

normalization factor

$\boldsymbol{X}^{old} \in Q^V$ is the old $\boldsymbol{X}$ **before** the resampling in step ①

Normalization Factor $C_e = C_e(\boldsymbol{X}_{\mathcal{R}}^{old})$:

$$C_e = \min_{y \in Q^e : y_{e \cap \mathcal{R}} = X_{e \cap \mathcal{R}}^{old}} \phi_e(y).$$



While$(\mathcal{R} \neq \emptyset)$

$(X, \mathcal{R}) \leftarrow$ **Local-Resample**$(X, \mathcal{R})$

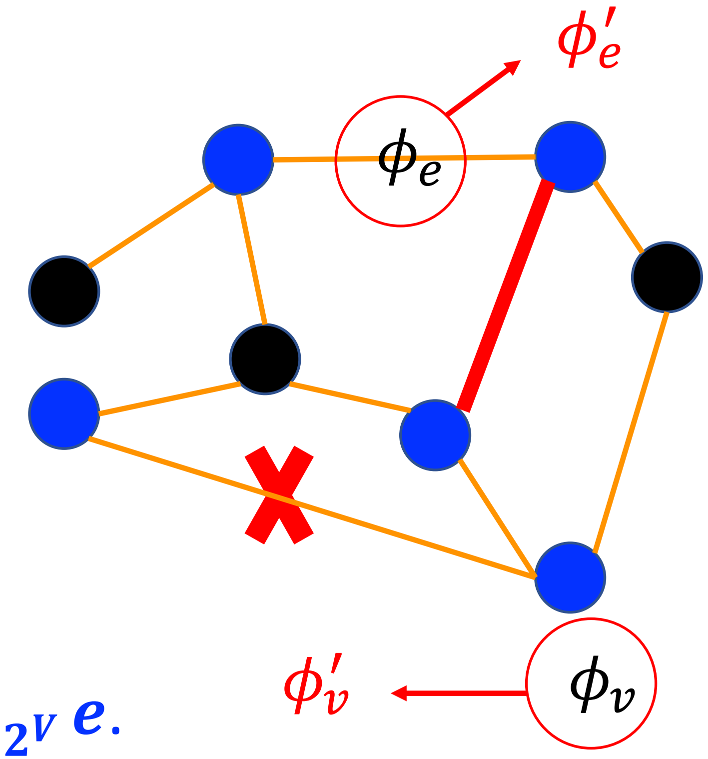
Output $\boldsymbol{X}$.

Correct Distribution

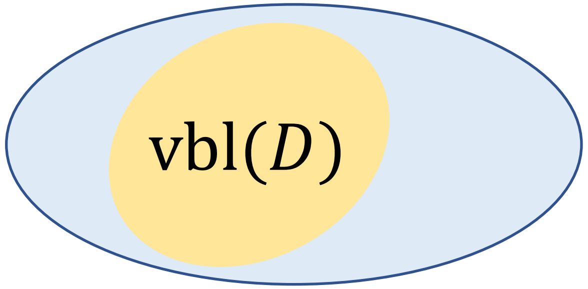
Dynamic Sampler

Dynamic Sampling from Graphical Model

- **Input**: a graphical model $\mathcal{I}$; a sample $\boldsymbol{X} \sim \mu_{\mathcal{I}}$
  a update (D, Φ_D) that modifies $\mathcal{I}$ to $\mathcal{I}'$;
- **Output**: a sample $\boldsymbol{X}' \sim \mu_{\mathcal{I}'}$.



- D: updated variables & updated constraints;
- $\mathrm{vbl}(D)$: variables **involved** by the update:
  - **updated variables: $\boldsymbol{D} \cap \boldsymbol{V}$;**
  - **variables incident to updated constraints:$\bigcup_{e \in D \cap 2^V} \boldsymbol{e}$.**



Graphical models $\mathcal{I}$ and $\mathcal{I}'$ **differ** only on $\mathrm{vbl}(D)$.

⇓

The initial **bad set** $\mathcal{R} = \mathrm{vbl}(D)$.

Dynamic Sampler

- Apply changes (D, Φ_D) to current graphical model $\mathcal{I}$.
- $\mathcal{R} \leftarrow \mathrm{vbl}(D)$;
- **While**$(\mathcal{R} \neq \emptyset)$
  - $(\boldsymbol{X}, \mathcal{R}) \leftarrow$**Local-Resample**$(\boldsymbol{X}, \mathcal{R})$;
- **Return** $\boldsymbol{X}$;

Theorem: Correctness [This Work]

Upon termination, the dynamic sampler outputs $\boldsymbol{X} \sim \mu_{\mathcal{I}'}$.

A dynamic sampler for **general graphical model**:

- ***Exact sampling***
- ***Las Vegas***
- ***Distributed / Parallel***

Proof of Correctness

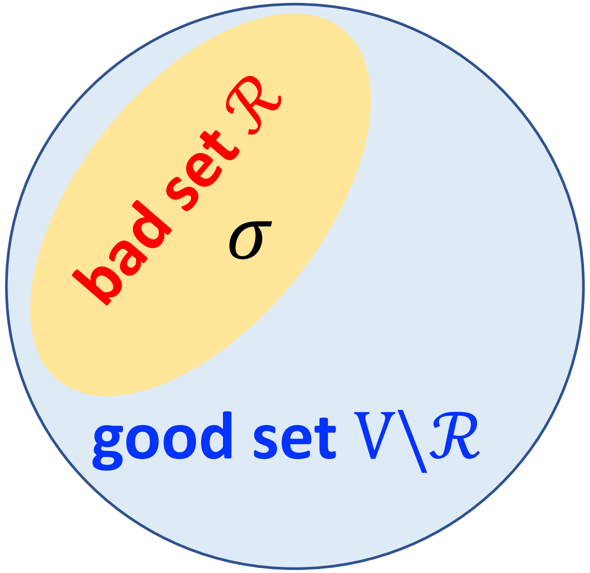
Dynamic Sampler

- Apply changes (D, Φ_D) to current graphical model $\mathcal{I}$.
- $\mathcal{R} \leftarrow \mathrm{vbl}(D)$;
- **While**$(\mathcal{R} \neq \emptyset)$
  - $(\boldsymbol{X}, \mathcal{R}) \leftarrow$**Local-Resample**$(\boldsymbol{X}, \mathcal{R})$;
- **Return** $\boldsymbol{X}$;

The algorithm maintains $(X, \mathcal{R}) \in Q^V \times 2^V$

- $\mathcal{R}$: **bad set**;
- $V \setminus \mathcal{R}$: **good set**

  $X_{V\setminus\mathcal{R}}$ follows "**correct**" distribution.



Conditional Gibbs property w.r.t. $\boldsymbol{\mu}$

Conditioning on any $\mathcal{R} \subseteq V$ and any assignment $\sigma \in Q^{\mathcal{R}}$ of $X_{\mathcal{R}}$, the distribution of $X_{V\setminus\mathcal{R}}$ is $\mu^{\sigma}_{V\setminus\mathcal{R}}$.

$\mu^{\sigma}_{V\setminus\mathcal{R}}$: marginal distribution of μ on $V\setminus\mathcal{R}$ conditioning on σ.

Local-Resample$(\boldsymbol{X}, \mathcal{R})$

define →

Resampling chain

- **Markov chain** on $\Omega = Q^V \times 2^V$
- Transition Matrix $P \in \mathbb{R}^{\Omega\times\Omega}$

$$P: (\boldsymbol{X}, \mathcal{R}) \to (\boldsymbol{X}', \mathcal{R}')$$

Equilibrium Condition

If $(\boldsymbol{X}, \mathcal{R})$ satisfies the conditionally Gibbs property w.r.t. μ, then so does $(\boldsymbol{X}', \mathcal{R}')$.

↑

Equation System for Equilibrium Condition

$\forall\ S, T \subseteq V, \sigma \in Q^{V\setminus S}$ and $\tau \in Q^{V\setminus T}$,

$$\forall y \in Q^V, y_{V\setminus T} = \tau: \sum_{\substack{x\in Q^V\\ x_{V\setminus S}=\sigma}} \mu_S^\sigma(x_S)\cdot P\big((x,S),(y,T)\big) = C(S,\sigma,T,\tau)\cdot \mu_T^\tau(y_T).$$

↑

Our algorithm is **a solution to** this equation system.

Theorem: Fast Convergence [This Work]

The updated graphical model satisfies $d = O(1)$, $\max_{e \in E'} |e| = O(1)$, and

$$\forall e \in E': \ \min_x \phi'_e(x) > \sqrt{1 - \frac{1}{d+1}}$$

where d is the maximum degree of the dependency graph.

The **cost** of the dynamic sampler is

- $O(\log |D|)$ **iterations** in **expectation**;
- $O(|D|)$ **resamplings** in **expectation**.

$$\mathcal{J} \xrightarrow{\text{Update } (\boldsymbol{D}, \Phi_D)} \mathcal{J}'$$

Ising Model: $\forall e \in E$:

$$1 - \exp(-2|\beta_e|) < \frac{1}{4\Delta}$$

Uniqueness Regime: $\forall e \in E$:

$$1 - \exp(-2|\beta_e|) < \frac{2}{\Delta}$$

Theorem: Fast Convergence [This Work]

Hardcore model and Ising model on bounded degree graph s.t.

- Hardcore model: $\forall v \in V$: $\lambda_v \leq \frac{1}{\sqrt{2}\Delta - 1}$.
- Ising model: $\forall e \in E$: $1 - \exp(-2|\beta_e|) \leq \frac{1}{2.221\Delta + 1}$,

where Δ is the maximum degree.

The **cost** of the dynamic sampler is

- $O(\log|D|)$ **iterations** in **expectation;**
- $O(|D|)$ **resamplings** in **expectation;**

Uniqueness Regime:

- Hardcore model: $\forall v \in V$: $\lambda_v < \frac{(\Delta-1)^{\Delta-1}}{(\Delta-2)^{\Delta}} \approx \frac{e}{\Delta-2}$. $\lambda_v = O\left(\frac{1}{\Delta}\right)$
- Ising model: $\forall e \in E$: $1 - \exp(-2|\beta_e|) < \frac{2}{\Delta}$. $1 - \exp(-2|\beta_e|) = O\left(\frac{1}{\Delta}\right)$

Proof of the Fast Convergence

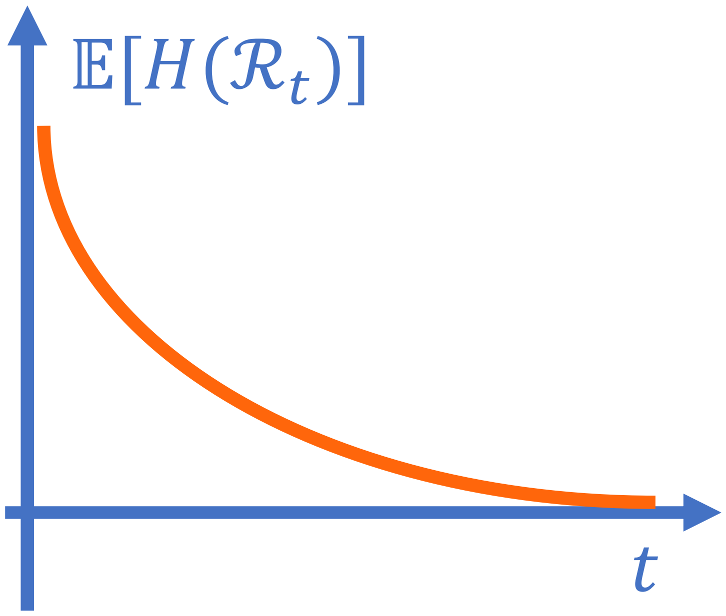
$\mathcal{R}_0 = \text{vbl}(D)$ → $\mathcal{R}_1$ → $\mathcal{R}_2$ → • • • → $\mathcal{R}_T = \emptyset$

Potential function on **bad set** $\mathcal{R}_t$

$$H: 2^V \to \mathbb{Z}_{\geq 0}$$

Step-wise decay on **expectation** of $H(\mathcal{R}_t)$

$$\mathbb{E}[H(\mathcal{R}_t)] \leq (1-\delta)\mathbb{E}[H(\mathcal{R}_{t-1})].$$



Summary

- **Dynamic sampling problem**.
- **Dynamic sampler** for general graphical models

  Exact Sampling & Las Vegas & Distributed/Parallel.

- **Equilibrium conditions** for resampling.

Future Work

- **Dynamic MCMC sampling** [Feng, He, Yin, Sun, arXiv:1904.11807]
- Improve the **regimes** for efficient dynamic sampling

  correlation decay →? *efficient dynamic sampling algorithm.*

- Extend to **continuous distributions** & **global constraints**.

Thank You

See you at the poster session #131