Sampling in High-Dimensional Space in Network Environment

(from a *Theory of Computing* Point of View)

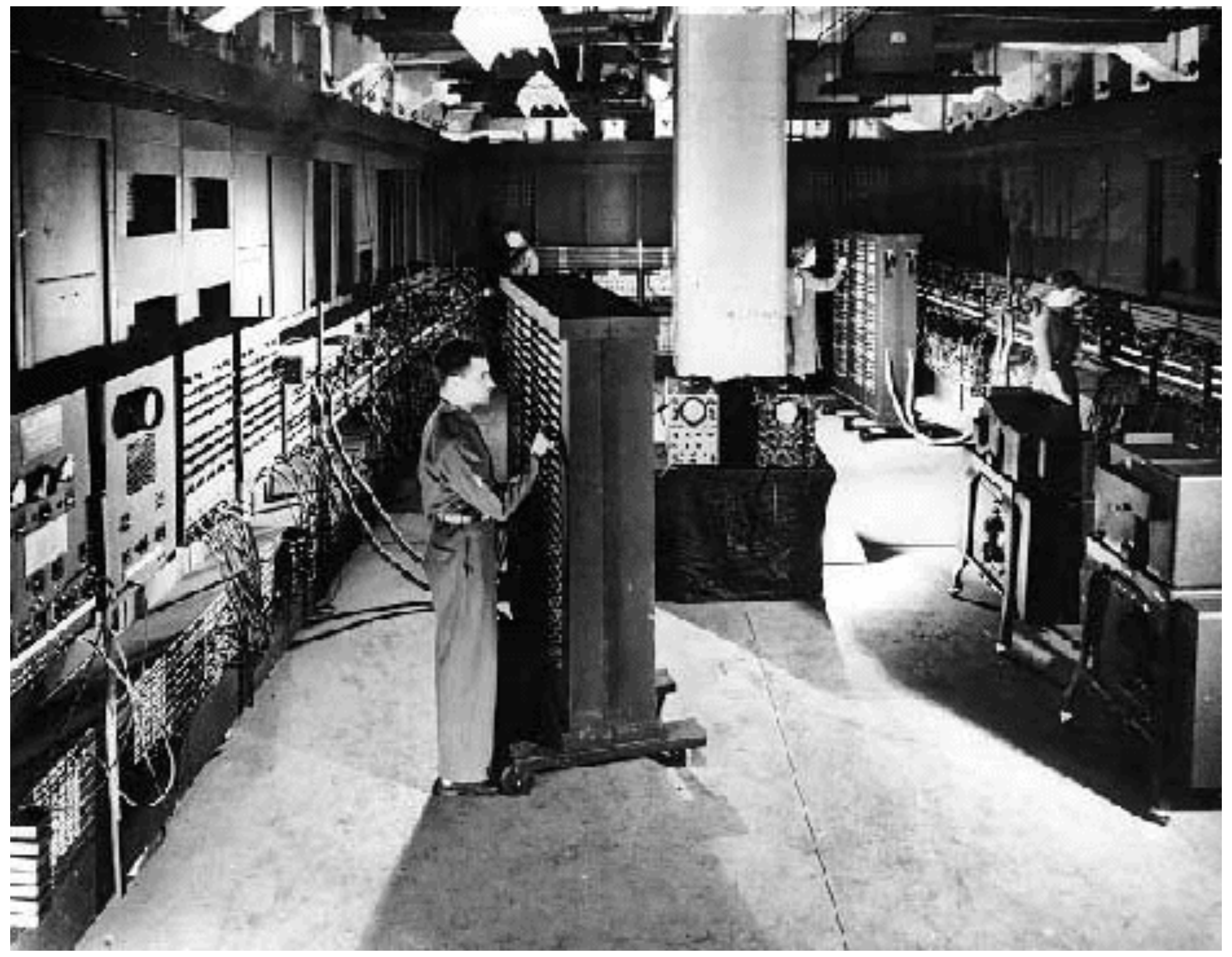
Yitong Yin (尹一通)
Nanjing University

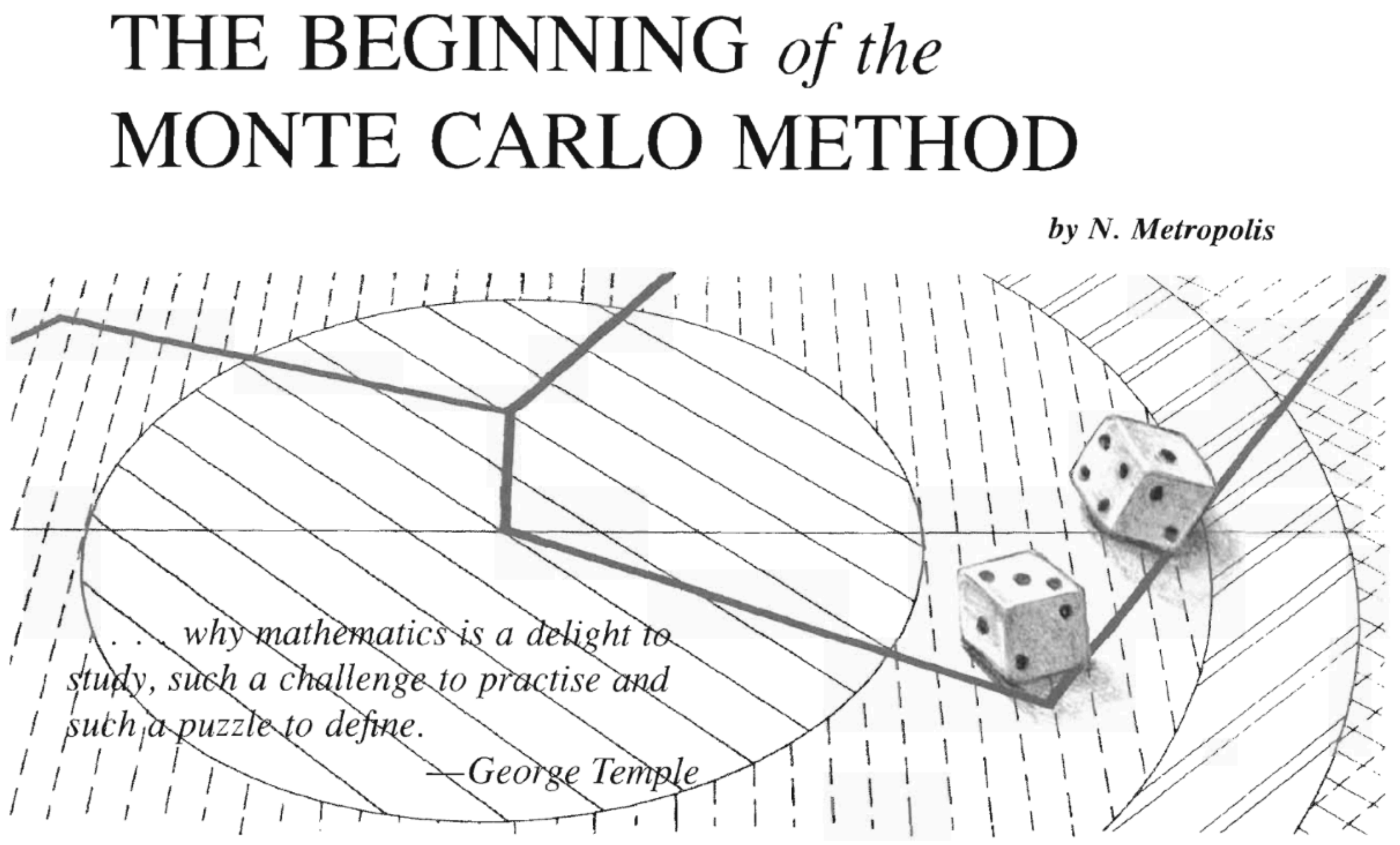
Huawei STW 2022
Shenzhen 2022.6.26-6.28

Sampling in High-Dimensional Space

Given an n-dimensional joint distribution μ,
draw a sample $\boldsymbol{X} = (X_1, X_2, \ldots, X_n) \sim \mu$.

- One of the earliest computer programs (*nuclear Monte Carlo simulations on ENIAC*)

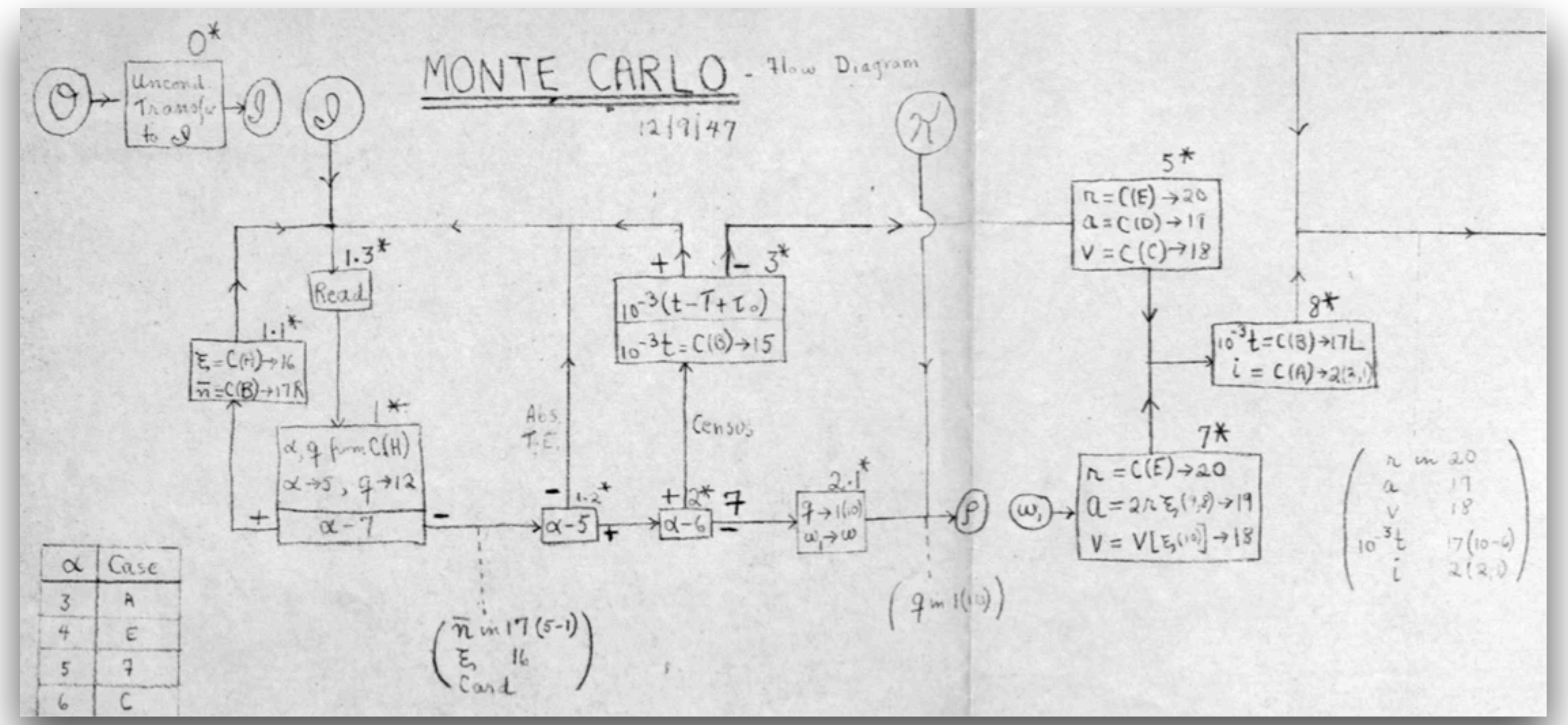




THE BEGINNING *of the*
MONTE CARLO METHOD

by N. Metropolis

. . . why mathematics is a delight to study, such a challenge to practise and such a puzzle to define.
—George Temple



Sampling in High-Dimensional Space

Given an n-dimensional joint distribution μ,
draw a sample $\boldsymbol{X} = (X_1, X_2, \ldots, X_n) \sim \mu$.

- One of the earliest computer programs (*nuclear Monte Carlo simulations on ENIAC*)
- Crucial for **today's** computational tasks:
  - **Probabilistic inference:** guessing possible values of X_i given values of X_S
  - **Optimization via sampling:** finding $\boldsymbol{x}$ with max $\mu(\boldsymbol{x})$ by drawing $\boldsymbol{X} \sim \mu$
  - **High-dimensional integration:** calculating $\int_{\mathbb{R}^n}$ or estimating volumes
  - **Statistical physics:** simulating interacting particle systems
  - **Approximate counting:** e.g. estimating *Network Reliability*

Gibbs Distribution

- **High-dimensional distribution μ described by local constraints:**
  - n variables on finite discrete domain Ω
  - a set $\mathscr{C}$ of local constraints (f, S) with scope $S \subseteq [n]$ and $f : \Omega^S \to [0,1]$

$$\forall \boldsymbol{x} \in \Omega^n: \qquad \mu(\boldsymbol{x}) \propto \prod_{(f,S)\in\mathscr{C}} f(\boldsymbol{x}_S)$$

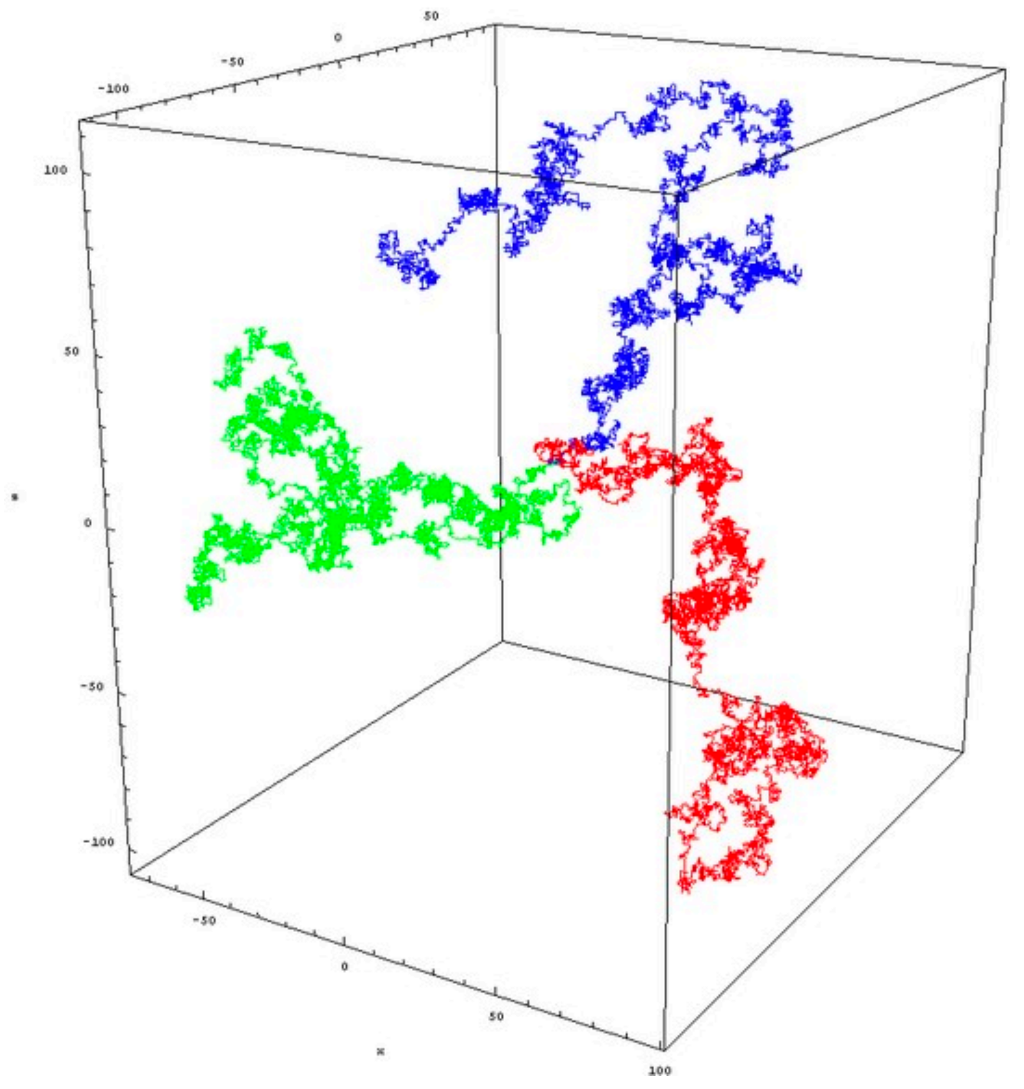
- For hard constraints $f : \Omega^S \to \{0,1\}$, the μ is uniform distribution over *constraint satisfaction solutions*
- **Examples of Gibbs distributions:** graphical model, Bayesian network, Boltzmann machine, Markov random field, factor graph, spin system, weighted CSP, …

Gibbs Sampler

(a.k.a. *Glauber dynamics, heat-bath dynamics*) **[Glauber 1963]**

the **Markov chain** maintains an $\boldsymbol{x} \in \Omega^n$; at each step:

- pick an $v \in \{1,2,\ldots,n\}$ uniformly at random;
- update x_v randomly according to the marginal distribution $\mu_v(\cdot \mid x_{N(v)})$;



Random walk in Ω^n

Gibbs Sampler

(a.k.a. *Glauber dynamics, heat-bath dynamics*) **[Glauber 1963]**

the **Markov chain** maintains an $\boldsymbol{x} \in \Omega^n$; at each step:

- pick an $v \in \{1,2,\ldots,n\}$ uniformly at random;
- update x_v randomly according to the marginal distribution $\mu_v(\cdot \mid x_{N(v)})$;

- The **Gibbs sampler** converges (**mixes**) to μ.
  - Many other Markov chains converge to μ, e.g. *the Metropolis algorithm* **[Metropolis 1953]**
- The **convergence rate** (**mixing time**) depends on properties of μ.

$$T_{\text{mix}} = \max_{\boldsymbol{x}} \min \left\{ t \geq 1 \mid \|P^t(x, \cdot) - \mu\|_1 \leq 1/\mathrm{e} \right\}$$

- **What makes a family of distributions easy/hard to sample? New algorithms?**

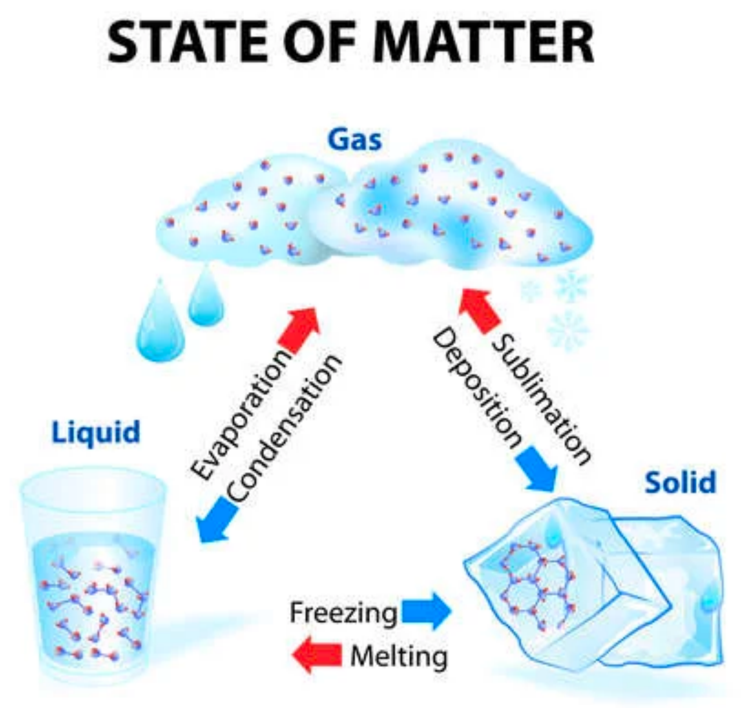
Outline

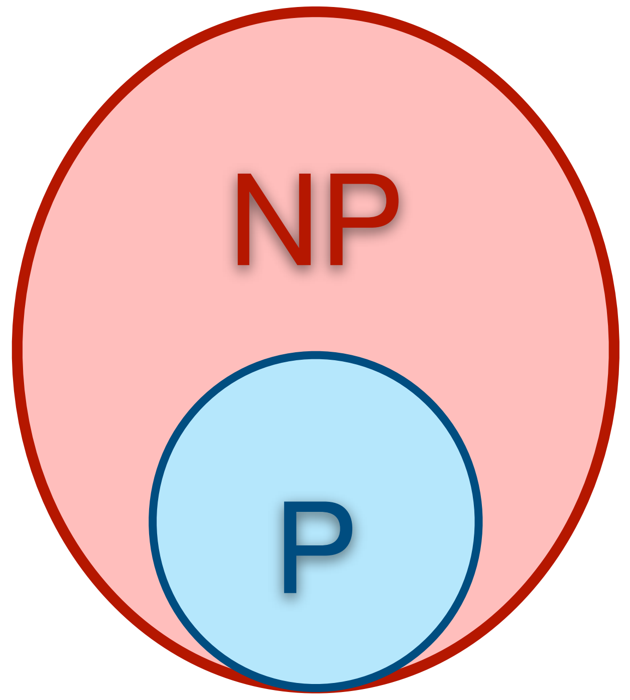
- **Computational Phase Transition of Sampling**
  - Phase transition of probabilistic graphical models
  - Phase transition of sampling constraint satisfaction solutions (a.k.a. *a sampling Lovász local lemma*)
- **Network Algorithms for Gibbs Sampling**
  - Parallel/Distributed/Dynamic sampling algorithms
- **Application:** *Network Reliability Estimation*

Computational Phase Transition of Sampling

Computational Phase Transition

Physical
Phase Transition

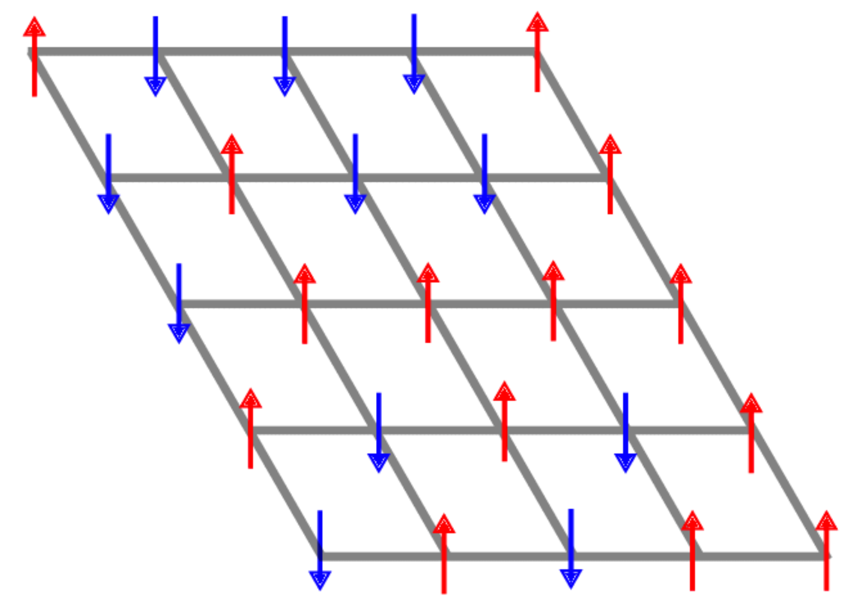




Computational
Complexity

- **Gibbs distribution:**

$$\forall \boldsymbol{x} \in \Omega^n: \quad \mu(\boldsymbol{x}) \propto \prod_{(f,S)\in\mathscr{C}} f(\boldsymbol{x}_S)$$



- **locally constrained random variables $\Longleftrightarrow$ locally interacting particle states**

- Continuous change of strength of local interaction $\Longrightarrow$ sharp transition of global state (state of matter / computational complexity)

Hardcore Model

Gibbs sampler:

maintain an independent set $I \subseteq V$:

- pick an $v \in V$ uniformly at random;
- if $I \cup \{v\}$ is independent set then

$$I \leftarrow \begin{cases} I \cup \{v\} & \text{with prob. } \frac{\lambda}{1+\lambda} \\ I \setminus \{v\} & \text{with prob. } \frac{1}{1+\lambda} \end{cases}$$

- Given a graph $G(V, E)$ and a parameter $\lambda > 0$:

  $\forall$ independent set $I \subseteq V$ of G:

$$\mu(I) \propto \lambda^{|I|}$$

- **Critical threshold** (for phase transition of *hardcore gas with fugacity λ on Δ-degree Bethe lattice*):

$$\lambda_c(\Delta) \triangleq \frac{(\Delta-1)^{\Delta-1}}{(\Delta-2)^{\Delta}} \approx \frac{\mathrm{e}}{\Delta-2}$$

- $\lambda > \lambda_c(\Delta) \implies$ sampling is **NP-hard** [Sly, FOCS 2010 best paper]
- $\lambda < \lambda_c(\Delta)$:

strong spatial mixing (SSM)
high-dimensional expander (HDX)
local-to-global argument
modified log-Sobolev inequality
field dynamics
… …

Gibbs sampler

$$n^{O(\log \Delta)} \longrightarrow n^{f(\lambda)} \longrightarrow \Delta^{O(\Delta^2)} n \log n \longrightarrow O(n^2 \log n) \longrightarrow O(n \log n) \text{ optimal}$$

[Weitz STOC '07] — [Anari, Liu, Oveis Gharan FOCS '20] — [Chen, Liu, Vigoda STOC '21] — [Chen, Feng, **Y.**, Zhang FOCS '21] — [Chen, Feng, **Y.**, Zhang, FOCS '22] [Chen, Eldan, FOCS '22]

for all Gibbs distributions with pairwise repulsive constraints on Boolean variables (*anti-ferromagnetic two-state spin systems*)

Constraint Satisfaction Solutions

- For **hard constraints** (Boolean decisions) $f : \Omega^S \to \{0,1\}$

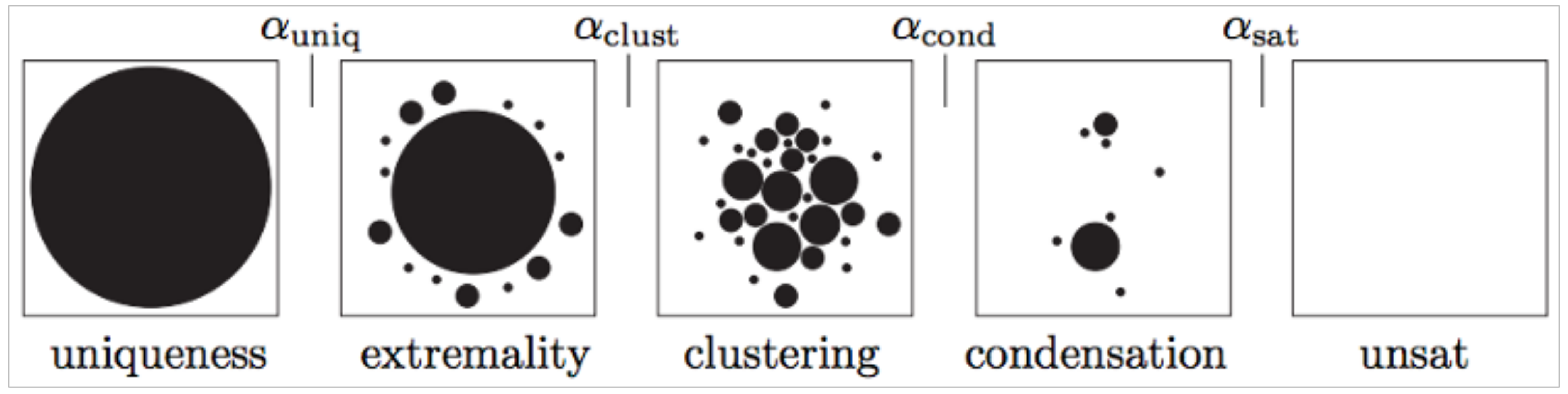
$$\forall \boldsymbol{x} \in \Omega^n: \qquad \mu(\boldsymbol{x}) \propto \prod_{(f,S)\in\mathscr{C}} f(\boldsymbol{x}_S)$$

  μ is the uniform distribution over all **constraint satisfaction solutions**

- **Example:** *k*-SAT with variable degree d

  CNF formula $(x_1 \vee \neg x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee x_4) \wedge (x_3 \vee \neg x_4 \vee \neg x_5)$

- **Barrier**: classic sampling algorithms rely on ***connectivity*** of solution space



Satisfying solution exists when $k \gtrsim \log d$
(*Lovász local lemma*)

Sampling?

Overcome the Connectivity Barrier

Projected Markov chain:

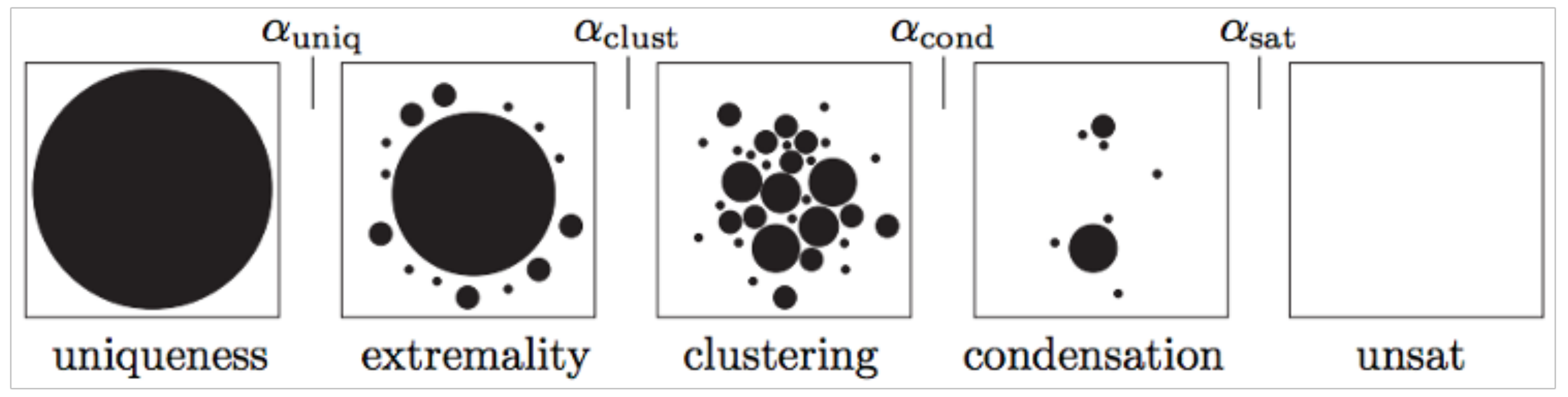
Properly construct a subset $U \subseteq V$ of variables;

Sample $\boldsymbol{x}_U \sim \mu_U$ by simulating Gibbs sampler on μ_U;

Extend $\boldsymbol{x}_U$ to a satisfying solution $\boldsymbol{x} \sim \mu$;

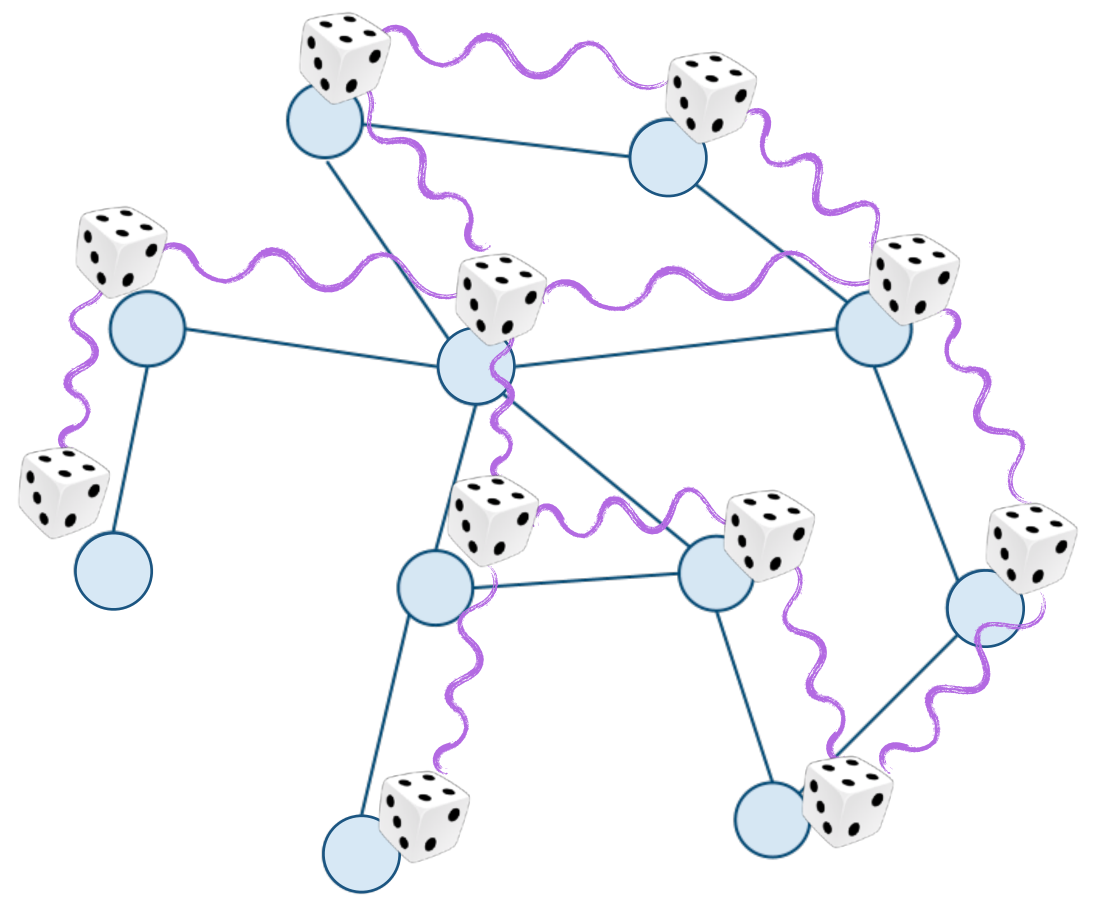
Idea: project onto lower dimension to improve connectivity

- Efficiently construct a "good" subspace $U \subseteq V$:
  - Gibbs sampler for μ_U is fast-convergent (the subspace is well-connected) and efficient to implement
  - it is efficient to extend a random partial solution $\boldsymbol{x}_U \sim \mu_U$ to a uniform satisfying solution $\boldsymbol{x} \sim \mu$
- **Fast sampler in near-linear time** (under *Lovász local lemma like condition*):
  - SAT [Feng, Guo, Y., Zhang, STOC 2020]
  - CSP with *atomic* constraints [Feng, He, Y., STOC 2021]
  - *general* CSP (*constraint satisfaction problem*) [He, Wang, Y., FOCS 2022]



Network Algorithms for Gibbs Sampling

Distributed Gibbs Sampling



- **Generate high-dimensional sample in a network:**
  - Each node $v \in \{1,2,\ldots,n\}$ generates a random X_v
  - Altogether it follows the correct joint distribution

$$\boldsymbol{X} = (X_1, X_2, \ldots, X_n) \sim \mu$$

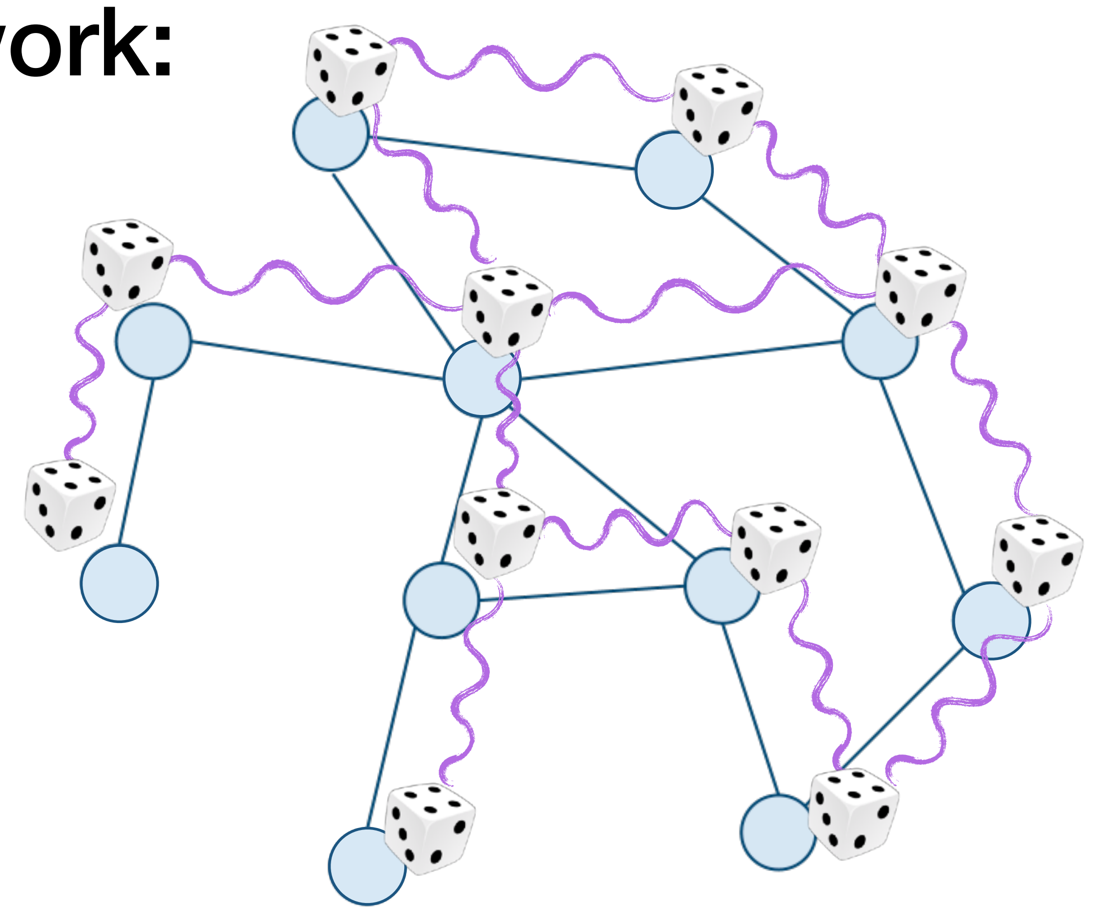
Distributed Gibbs Sampling

Gibbs sampler for μ:

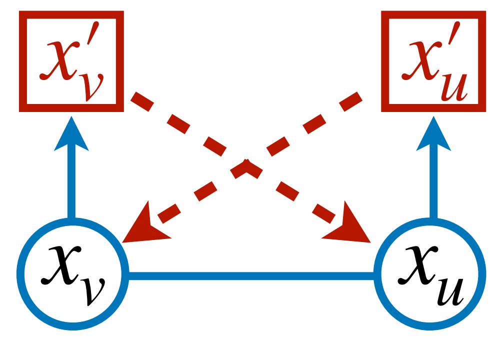
maintain an $\boldsymbol{x} \in \Omega^n$, at each step:

- pick a random $v \in \{1,2,\ldots,n\}$;
- update x_v according to $\mu_v(\cdot \mid x_{N(v)})$;

Generate high-dimensional samples in a network:



- Classic sampling algorithms are intrinsically *sequential*.
- Barrier for parallelization: update of variable depends on neighbors' states



  - concurrent updates of adjacent variables $\Longrightarrow$ fault
  - correct parallelization: $O(\Delta)$ **overhead!**

- **Is it possible to correctly parallelize the Markov chain with linear speedup?**

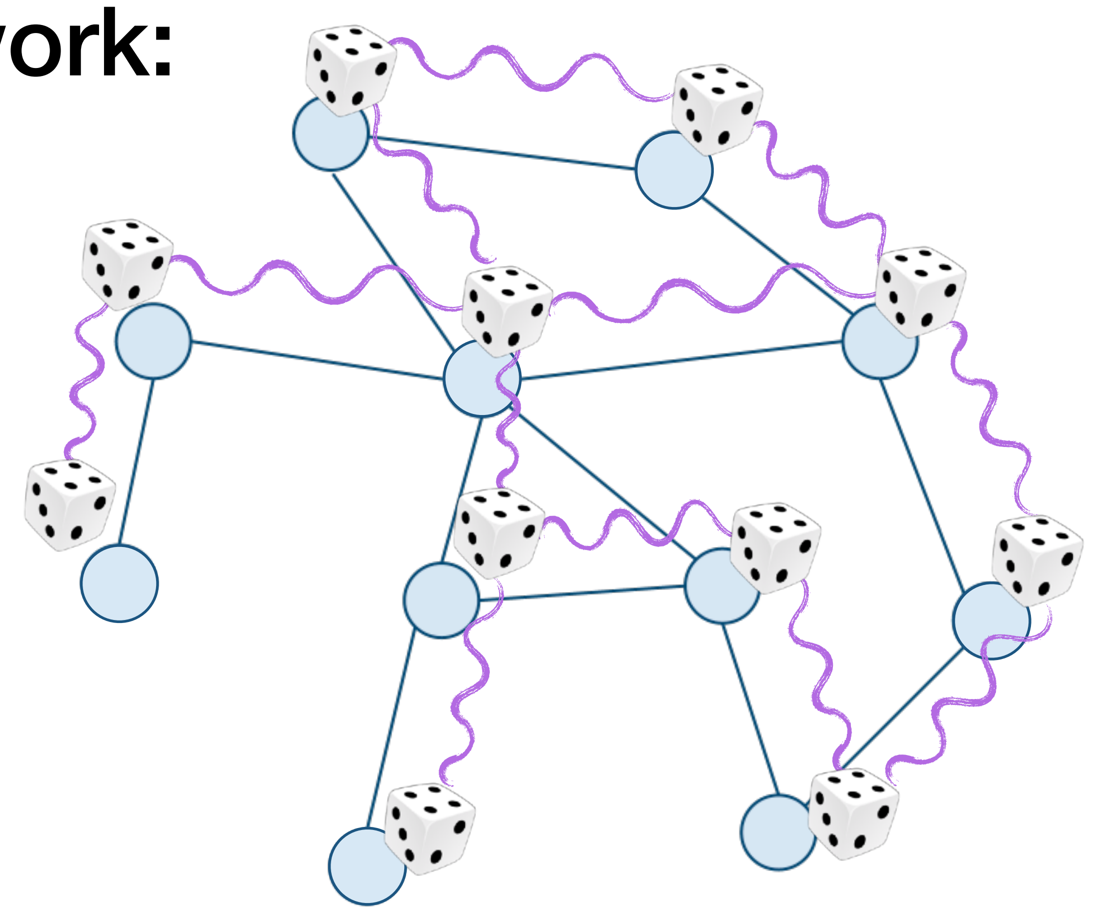
Distributed Gibbs Sampling

Gibbs sampler for μ:

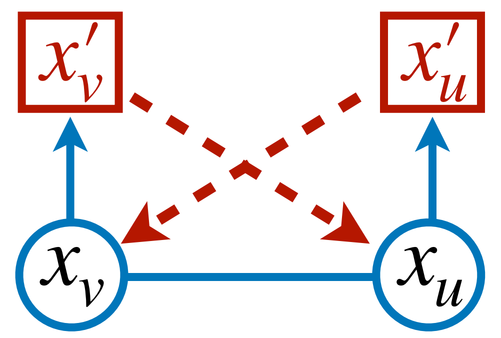
maintain an $\boldsymbol{x} \in \Omega^n$, at each step:

- pick a random $v \in \{1,2,\ldots,n\}$;
- update x_v according to $\mu_v(\,\cdot \mid x_{N(v)})$;

Generate high-dimensional samples in a network:



- Is it possible to correctly parallelize the Markov chain with linear speedup?



- a parallel chain called *LocalMetropolis* [Feng, Sun, Y., PODC 2017]
- sampling by *network decomposition* [Feng, Y., PODC 2018]
- parallelize *Metropolis algorithm* [Feng, Hayes, Y., SODA 2021]

An Idealized Parallel "Sampling Algorithm"

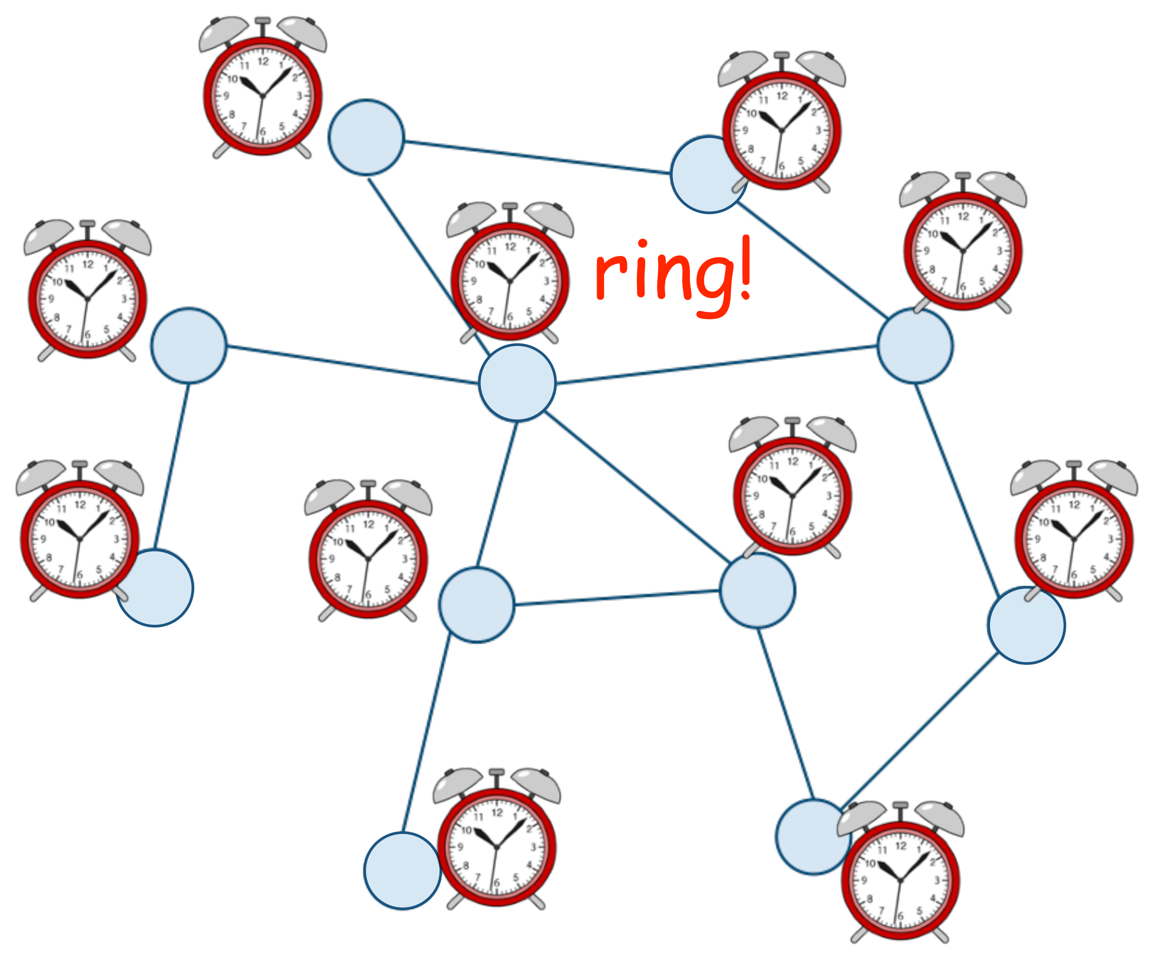
Continuous-time Gibbs sampler for μ:

each $v \in \{1,2,\ldots,n\}$ holds a Poisson clock;

when the clock at v rings:

- update x_v according to $\mu_v(\cdot \mid x_{N(v)})$;

atomic operation



$$O(T) \text{ continuous-time duration} \iff O(nT) \text{ discrete-time steps}$$

- This is the original definition of **Gibbs sampler** [Glauber 1963].
- An idealized (*continuous-time* with *atomic update operation*) process that models the evolution of physical world.
- **Simulate this idealized process on computer network with no overhead?**

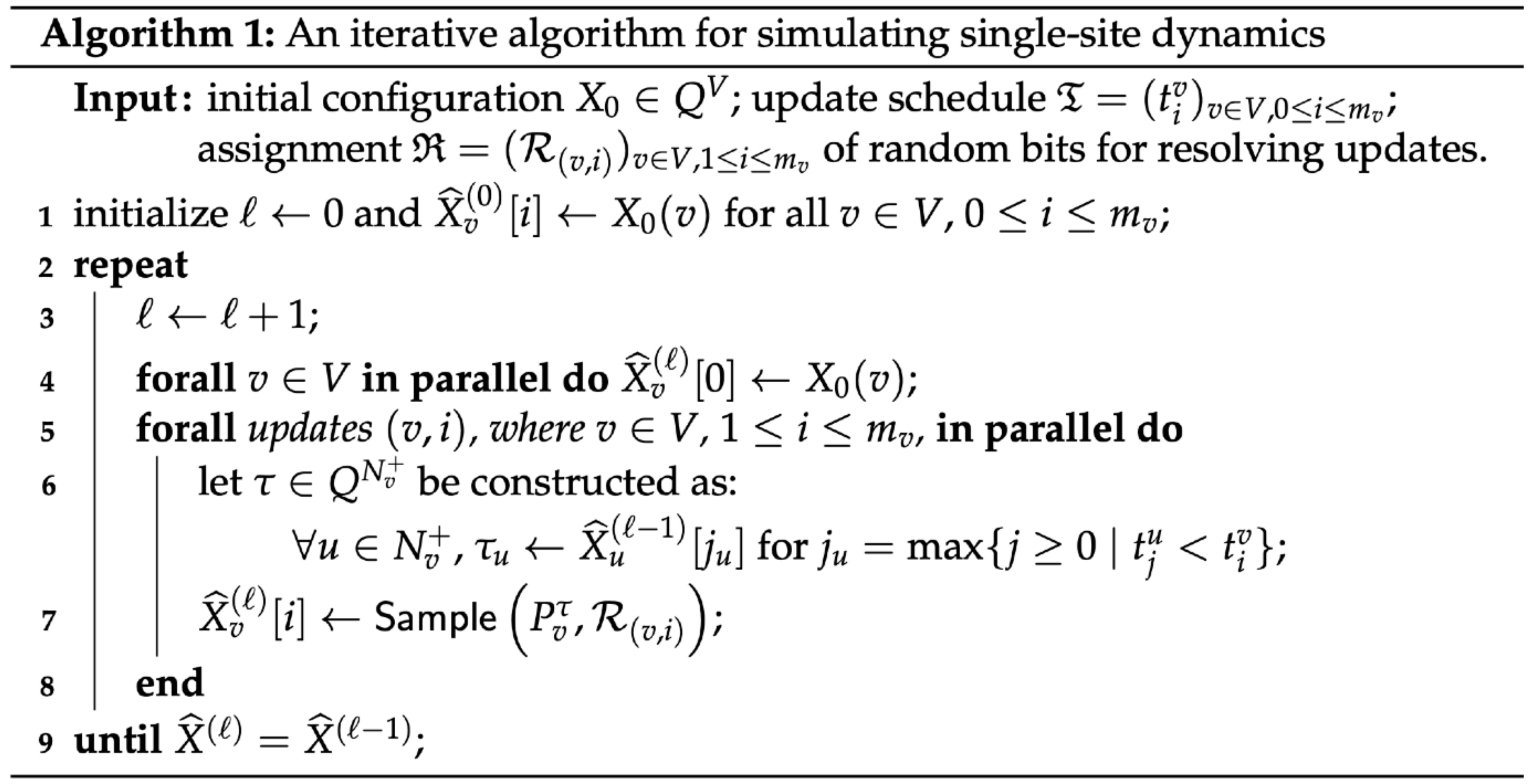
Parallelize the Gibbs Sampler

Continuous-time Gibbs sampler for μ:
[Glauber 1963]

each $v \in \{1,2,\ldots,n\}$ holds a Poisson clock;

when the clock at v rings:

- update x_v according to $\mu_v(\cdot \mid x_{N(v)})$;



Algorithm 1: An iterative algorithm for simulating single-site dynamics

```
Input: initial configuration $X_0 \in Q^V$; update schedule $\mathfrak{T} = (t_i^v)_{v\in V, 0\le i\le m_v}$;
       assignment $\mathfrak{R} = (\mathcal{R}_{(v,i)})_{v\in V, 1\le i\le m_v}$ of random bits for resolving updates.
initialize $\ell \leftarrow 0$ and $\widehat{X}_v^{(0)}[i] \leftarrow X_0(v)$ for all $v \in V, 0 \le i \le m_v$;
repeat
    $\ell \leftarrow \ell + 1$;
    forall $v \in V$ in parallel do $\widehat{X}_v^{(\ell)}[0] \leftarrow X_0(v)$;
    forall updates $(v,i)$, where $v \in V, 1 \le i \le m_v$, in parallel do
        let $\tau \in Q^{N_v^+}$ be constructed as:
            $\forall u \in N_v^+, \tau_u \leftarrow \widehat{X}_u^{(\ell-1)}[j_u]$ for $j_u = \max\{j \ge 0 \mid t_j^u < t_i^v\}$;
        $\widehat{X}_v^{(\ell)}[i] \leftarrow \mathsf{Sample}\left(P_v^\tau, \mathcal{R}_{(v,i)}\right)$;
    end
until $\widehat{X}^{(\ell)} = \widehat{X}^{(\ell-1)}$;
```

- Ideas:
  - Construct a dynamical system whose fixpoint corresponds to the correct evolution of the chain.
  - Simulate this dynamical system by a locally-iterative message passing algorithm on the network.
  - A universal coupling of random bits used in different iterations to ensure fast stabilization to fixpoint.

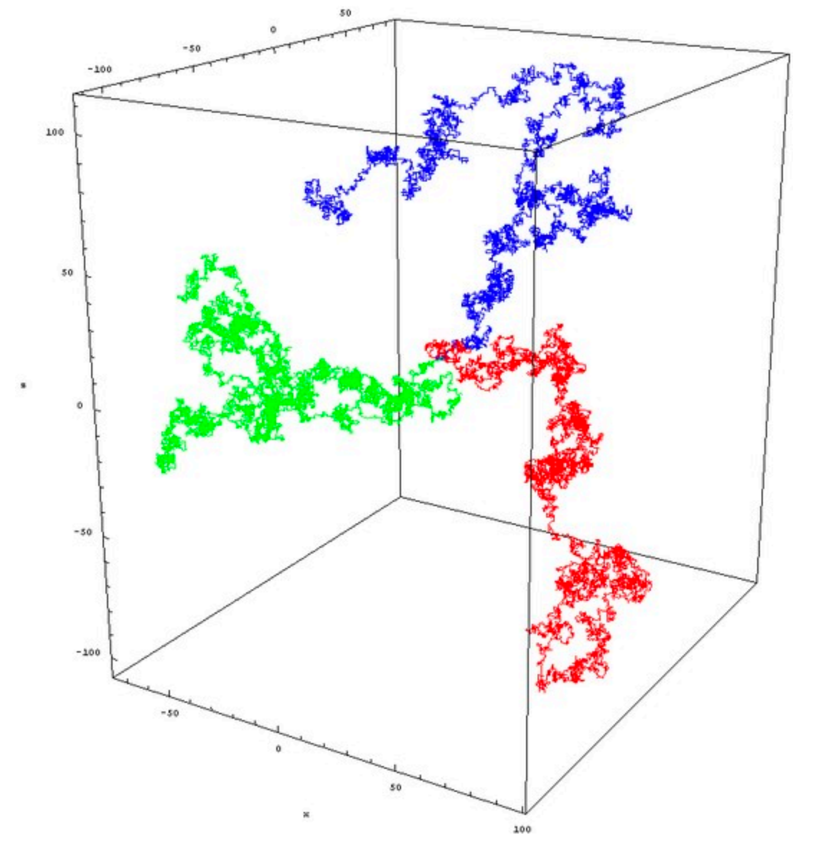
- A much weakened **Dobrushin's condition** (which is almost always satisfied)
  $\Longrightarrow$ **faithful parallel simulation of Gibbs sampler with linear speedup** [Liu, Y., STOC 2022]
  (all *single-site dynamics*)

Dynamic Sampling

Dynamic Sampling problem: for a dynamically changing graphical model $\mu \to \mu'$

$$X \sim \mu \xrightarrow[\text{with incremental cost}]{\text{dynamic update}} X' \sim \mu'$$

- Sampling/inference tasks on dynamically changing data:
  - Online data, data streams, network environment, etc.
- Dynamically changing graphical models generated in:
  - Locally-iterative algorithms for learning.
  - Self-reduction procedure in approximate counting.



Classic random walks fail on dynamic data

- **Algorithmic Lipschitz**: transform $X \sim \mu$ to $X' \sim \mu'$ with cost proportional to diff(μ, μ')

Dynamic Sampling

Dynamic Sampling problem: for a dynamically changing graphical model $\mu \to \mu'$

$$\boldsymbol{X} \sim \mu \xrightarrow[\text{with incremental cost}]{\text{dynamic update}} \boldsymbol{X}' \sim \mu'$$

```
Algorithm 1: Dynamic Sampler
  Input  : a graphical model $\mathcal{I}$ and a random sample $\boldsymbol{X} \sim \mu_{\mathcal{I}}$;
  Update: an update $(D, \Phi_D)$ which modifies $\mathcal{I}$ to $\mathcal{I}'$;
  Output: a random sample $\boldsymbol{X} \sim \mu_{\mathcal{I}'}$;
  $\mathcal{R} \leftarrow \mathsf{vbl}(D)$;
  while $\mathcal{R} \neq \emptyset$ do
      $(\boldsymbol{X}, \mathcal{R}) \leftarrow$ Local-Resample($\mathcal{I}'$, $\boldsymbol{X}$, $\mathcal{R}$);
  return $\boldsymbol{X}$;
```

```
Algorithm 2: Local-Resample($\mathcal{I}$, $\boldsymbol{X}$, $\mathcal{R}$)
  Input  : a graphical model $\mathcal{I} = (V, E, [q], \Phi)$, a configuration $\boldsymbol{X} \in [q]^V$ and a $\mathcal{R} \subseteq V$;
  Output: a new pair $(\boldsymbol{X}', \mathcal{R}')$ of configuration $\boldsymbol{X}' \in [q]^V$ and subset $\mathcal{R}' \subseteq V$;
  for each $e \in E^+(\mathcal{R})$, in parallel, compute $\kappa_e \triangleq \frac{1}{\phi_e(X_e)} \min_{x \in [q]^e : x_{e \cap \mathcal{R}} = X_{e \cap \mathcal{R}}} \phi_e(x)$;
  for each $v \in \mathcal{R}$, in parallel, resample $X_v \in [q]$ independently according to distribution $\phi_v$;
  for each $e \in E^+(\mathcal{R})$, in parallel, sample $F_e \in \{0,1\}$ ind. with $\Pr[F_e = 0] = \kappa_e \cdot \phi_e(X_e)$;
  $\boldsymbol{X}' \leftarrow \boldsymbol{X}$ and $\mathcal{R}' \leftarrow \bigcup_{e \in E : F_e = 1} e$;
  return $(\boldsymbol{X}', \mathcal{R}')$.
```

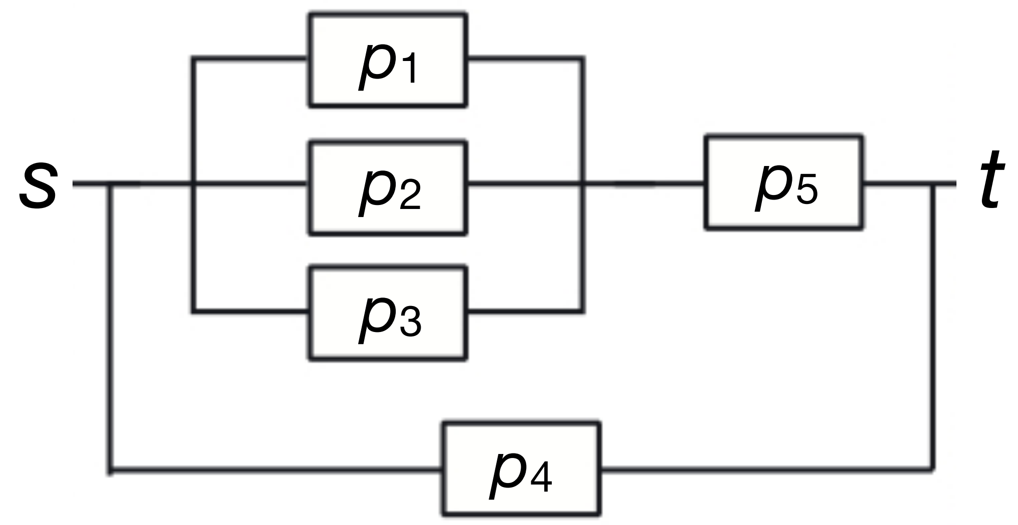
- A **dynamic sampling** algorithm: [Feng, Vishnoi, Y., STOC 2019]
  - correct and efficient on *dynamic* data
  - *parallel*, *distributed*, *communication-efficient*
  - *Las Vegas* algorithm for *perfect* sampling
- Based on **Partial Rejection Sampling** [Guo, Jerrum, Liu, STOC 2017]
  - very different from Markov chains (random walks).

Application:
Network Reliability Estimation

Network Reliability

[Valiant 1979]

- Given an undirected graph (a network) $G(V, E)$, and parameters $\vec{p} \in [0,1]^E$:
  - each edge $e \in E$ **fails** independently with prob. p_e
  - let $G(\vec{p})$ denote the resulting network
- (all-terminal) network reliability:



$$\boxed{\text{the probability that } G(\vec{p}) \text{ is connected}}$$

$$\mathrm{R}_{\vec{p}}(G) = \sum_{\boxed{\substack{C \subseteq E \text{ that} \\ \text{connects } V}}} \prod_{e \in C} (1 - p_e) \prod_{e \notin C} p_e$$

enumerating all connected subgraphs

Computational Complexity of Counting

- Let $A = \{a_{ij}\} \in \mathbb{R}^{n\times n}$ be a square matrix.
- **Determinant:** can be computed as fast as matrix multiplication

$$\sum_{\pi\in S_n} \mathrm{sgn}(\pi) \prod_{i=1}^{n} a_{i,\pi(i)}$$

- **Permanent:** is **#P-complete** [Valiant 1979]

$$\sum_{\pi\in S_n} \prod_{i=1}^{n} a_{i,\pi(i)}$$

solvable in polynomial-time $\Longrightarrow$ the polynomial hierarchy (**PH**) collapses
$\Longrightarrow$ **NP=P**

Network Reliability

- Given an undirected graph (a network) $G(V,E)$, and a parameter $\vec{p} \in [0,1]^E$:

(all-terminal) network reliability: $$\mathsf{R}_{\vec{p}}(G) = \sum_{\substack{C \subseteq E \text{ that} \\ \text{connects } V}} \prod_{e \in C}(1-p_e) \prod_{e \notin C} p_e$$

$$\boxed{\begin{array}{c}\text{the probability that the network remains connected} \\ \text{when each edge } e \in E \text{ fails independently with prob. } p_e\end{array}}$$

- The problem is **#P-complete [Valiant 1979] [Jerrum 1981]**:
    - $\mathsf{R}_{\vec{p}}(G)$ cannot be evaluated precisely in polynomial time unless **NP**=**P**

- **Approximation** by Monte Carlo method: return an estimation $\widehat{\mathsf{R}_{\vec{p}}(G)}$

$$\Pr\left[(1-\epsilon)\mathsf{R}_{\vec{p}}(G) \le \widehat{\mathsf{R}_{\vec{p}}(G)} \le (1+\epsilon)\mathsf{R}_{\vec{p}}(G)\right] \ge 1 - o(1)$$

Network Reliability by Sampling

$G(\vec{p})$: a subgraph of G obtained by removing each $e \in E$ independently with prob. p_e

- A naïve Monte Carlo estimation of **network reliability** $\mathrm{R}_{\vec{p}}(G)$:

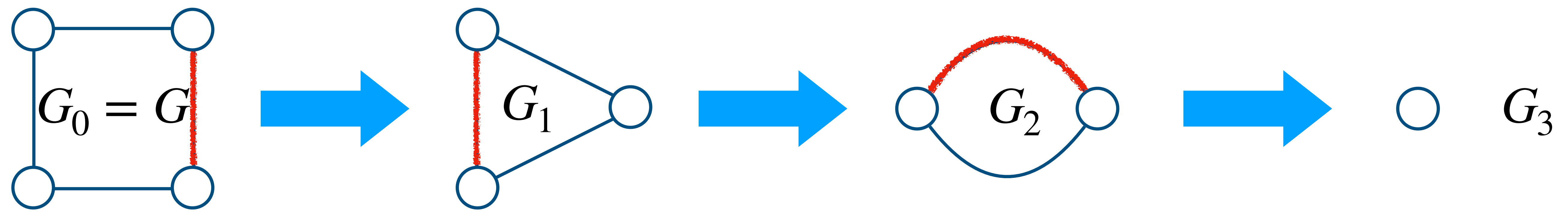
```
for $j = 1,2,\ldots,k$ for a large enough $k$:
    generate a $G^{(j)} \sim G(\vec{p})$ by removing each $e \in E$ independently with prob. $p_e$;
return $\frac{1}{k}\sum_{i=1}^{k} \mathbf{1}\left[G^{(j)} \text{ is connected}\right]$;
```

- Requires too many samples $G^{(j)} \sim G(\vec{p})$ if $\mathrm{R}_{\vec{p}}(G)$ is close to 0 (unreliable network).
- Monte Carlo method based on self-reduction:
  - Drawing samples $C \sim G(\vec{p})$ conditioned on C being **connected** on V.

Network Reliability by Sampling

$G(\vec{p})$: a subgraph of G obtained by removing each $e \in E$ independently with prob. p_e

- Monte Carlo method based on self-reduction:
  - Drawing samples $C \sim G(\vec{p})$ conditioned on C being **connected** on V.
- **Edge-contraction:**



  - Telescopic product: $\mathsf{R}_{\vec{p}}(G) = \frac{\mathsf{R}_{\vec{p}}(G_0)}{\mathsf{R}_{\vec{p}}(G_1)} \cdot \frac{\mathsf{R}_{\vec{p}}(G_1)}{\mathsf{R}_{\vec{p}}(G_2)} \cdot \frac{\mathsf{R}_{\vec{p}}(G_2)}{\mathsf{R}_{\vec{p}}(G_3)} \cdot \mathsf{R}_{\vec{p}}(G_3)$
  - $\frac{\mathsf{R}_{\vec{p}}(G_i)}{\mathsf{R}_{\vec{p}}(G_{i+1})}$ can be estimated by sampling $C \sim G_{i+1}(\vec{p})|$**connected.**

Markov Chain for Connected Subgraphs

$G(\vec{p})$: a subgraph of G obtained by removing each $e \in E$ independently with prob. p_e

- Monte Carlo method based on self-reduction:
    - Drawing samples $C \sim G(\vec{p})$ conditioned on C being **connected** on V.
- A natural Markov chain (Gibbs sampler) for connected subgraphs:

```
start with $C_0 = E$;  and for each step $t = 0,1,2\ldots$:
    pick an edge $e \in E$ uniformly at random;
    if $C_t - \{e\}$ disconnects $V$ then $C_{t+1} = C_t$; otherwise
        $C_{t+1} = \begin{cases} C_t \cup \{e\} & \text{with prob. } 1 - p_e \\ C_t - \{e\} & \text{with prob. } p_e \end{cases}$
```

m: number of edges

n: number of vertices

- The chain mixes (*converges*) to $G(\vec{p})|$**connected** in $O(m^2 \log n)$ steps.
- Conjecture: the chain mixes in $O(m \log n)$ steps.

Markov Chain for Connected Subgraphs

$G(\vec{p})$: a subgraph of G obtained by removing each $e \in E$ independently with prob. p_e

- Monte Carlo method based on self-reduction:
  - Drawing samples $C \sim G(\vec{p})$ conditioned on C being **connected** on V.
- A substantially more complicated Markov chain (matroid basis exchange) for connected subgraphs:
  - Each step (**matroid basis exchange**) requires $O(m)$ computation.
- The chain mixes (*converges*) to $G(\vec{p})|$**connected** in $O(m \log n)$ steps **[Anari, Liu, Oveis Gharan, Vinzant, STOC 2019 best paper]**
  - *Strongly log-concave distribution* **and** *high-dimension expander* **(HDX)**
- Markov chain comparison $\Longrightarrow$ the Gibbs sampler mixes in $O(m^2 \log n)$ steps

Markov Chain for Connected Subgraphs

$G(\vec{p})$: a subgraph of G obtained by removing each $e \in E$ independently with prob. p_e

- Monte Carlo method based on self-reduction:
    - Drawing samples $C \sim G(\vec{p})$ conditioned on C being **connected** on V.
- The Gibbs sampler converges in $O(m^2 \log n)$ steps.

$\times O(n)$
cost

- The matroid basis exchange chain converges in $O(m \log n)$ steps **[Anari, Liu, Oveis Gharan, Vinzant, STOC 2019 best paper]**
    - Each step (matroid basis exchange) requires $O(m)$ computation.
- Fastest estimation of network reliability runs in $O(mn^2 \log n)$ **[Guo, He, 2020]**
    - based on an ingenious reduction to sampling root-connected subgraph via partial rejection sampling [Guo, Jerrum, Liu, '17] / dynamic sampling [Feng, Nisheeth, Y. '19]

Network Reliability Estimation

- Given an undirected graph (a network) $G(V,E)$, and a parameter $\vec{p} \in [0,1]^E$:

  (all-terminal) network reliability: $\mathrm{R}_{\vec{p}}(G) = \sum_{\substack{C \subseteq E \text{ that} \\ \text{connects } V}} \prod_{e \in C} (1 - p_e) \prod_{e \notin C} p_e$

  the probability that the network remains connected when each edge $e \in E$ fails independently with prob. p_e

- Precisely evaluating $\mathrm{R}_{\vec{p}}(G)$ is **#P-complete**
- **Approximation** by Monte Carlo method in $\tilde{O}(mn^2)$ time
- Open problems:
  - estimating network reliability in $\tilde{O}(mn)$ time or less
  - network algorithms for network reliability (on going project …)

Computational Phase Transition of Sampling

Network Algorithms for Gibbs Sampling

Application: *Network Reliability Estimation*

Thank you!

- [Chen, Feng, **Y.**, Zhang '22]: Optimal mixing for two-state anti-ferromagnetic spin systems. **FOCS** '22.
- [He, Wang, **Y.** '22]: Sampling Lovász local lemma for general constraint satisfaction solutions in near-linear time. **FOCS** '22.
- [Liu, **Y.** '22]: Simple parallel algorithms for single-site dynamics. **STOC** '22.
- [Chen, Feng, **Y.**, Zhang '21]: Rapid mixing of Glauber dynamics via spectral independence for all degrees. **FOCS** '21.
- [Feng, He, **Y.** '21]: Sampling constraint satisfaction solutions in the local lemma regime. **STOC** '21.
- [Feng, Hayes, **Y.** '21]: Distributed Metropolis sampler with optimal parallelism. **SODA** '21.
- [Feng, Guo, **Y.**, Zhang '20]: Fast sampling and counting *k*-SAT solutions in the local lemma regime. **STOC** '20. **JACM** '21.
- [Feng, Vishnoi, **Y.** '19]: Dynamic sampling from graphical models. **STOC** '19. **SICOMP** '21.
- [Feng, **Y.** '18]: On local distributed sampling and counting. **PODC** '18.
- [Feng, Sun, **Y.** '17]: What can be sampled locally? **PODC** '17.