Distributed Algorithms *for* MCMC Sampling

Yitong Yin
Nanjing University

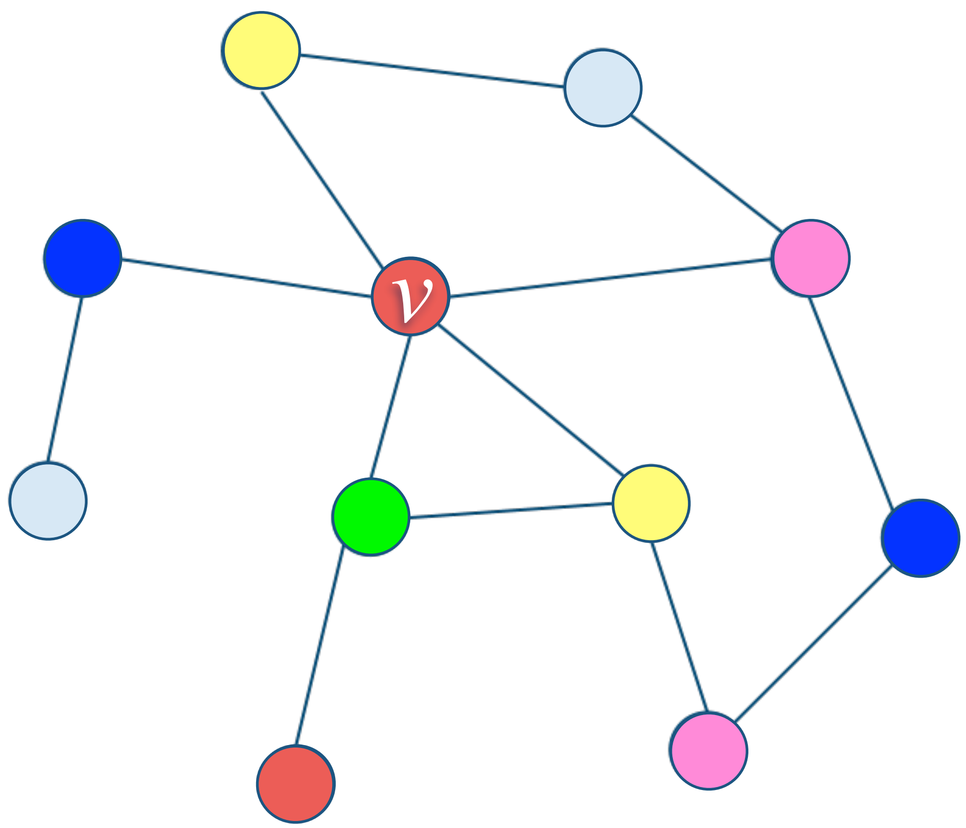
Shonan Meeting No. 162: Distributed Graph Algorithms

Outline

- Distributed Sampling Problem
  - Gibbs Distribution (distribution defined by local constraints)
- Algorithmic Ideas
  - MCMC *Local Metropolis Algorithm*
  - *LOCAL Jerrum-Valiant-Vazirani*
  - *Local Rejection Sampling*
- MCMC Distributed Simulation of Metropolis (with ideal parallelism)

MCMC: Markov chain Monte Carlo

Single-Site Markov Chain



Start from an arbitrary coloring $\in[q]^V$

at each step:

for a uniform random vertex v

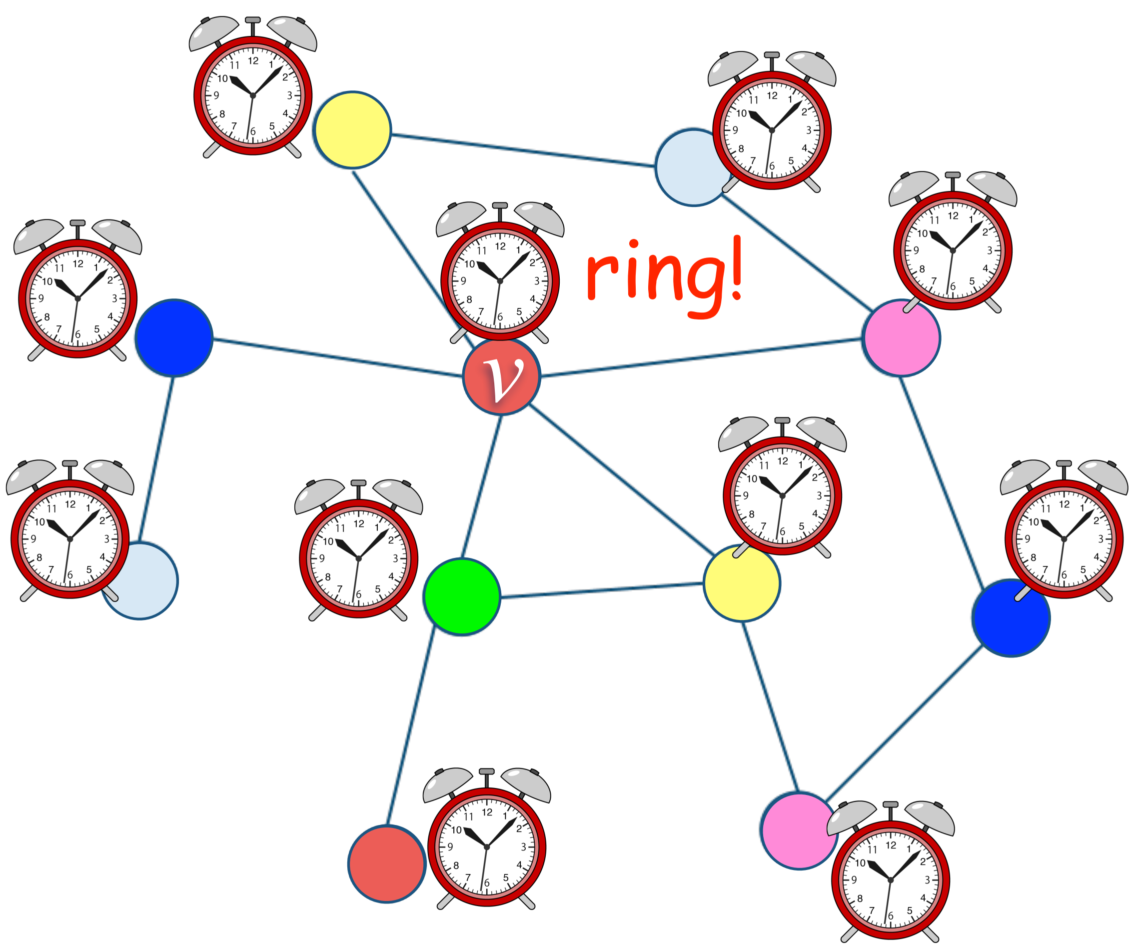
propose a random color $c\in[q]$;

change v's color to c if it's proper;

Metropolis Algorithm
(q-coloring)

Single-Site Markov Chain in 1960s

Each vertex holds an independent rate-1 Poisson clock.



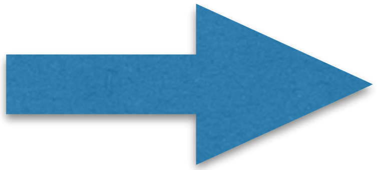
When the clock at v rings:

propose a random color $c \in [q]$;

change v's color to c if it's proper;

Metropolis Algorithm
(q-coloring)

continuous time T

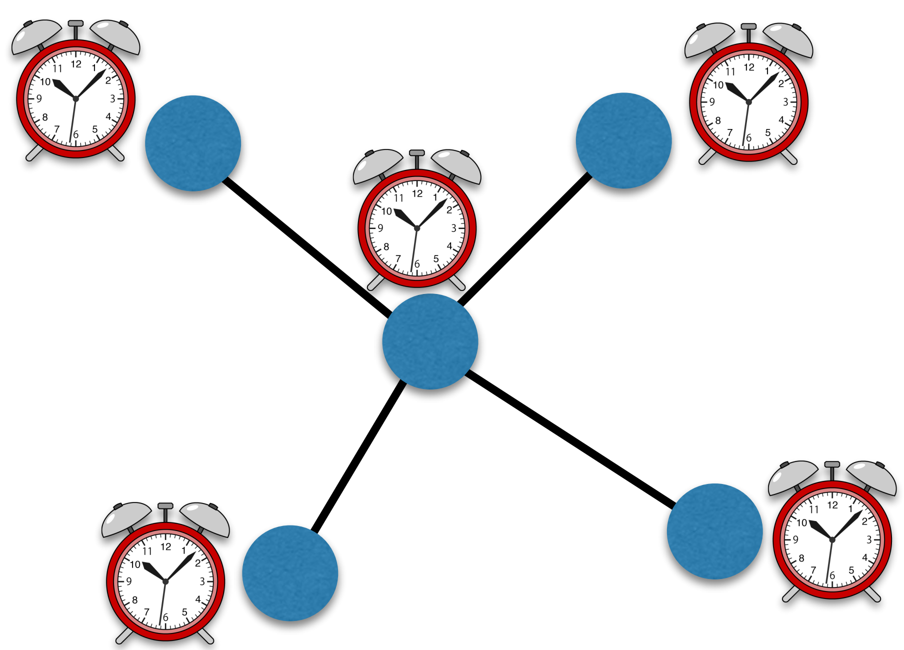


discrete time
$\theta(nT)$ sequential steps

Distributed Simulation of Continuous-Time Process

Goal: Give a distributed algorithm that perfect simulates the time T continuous Markov chain.

(Have the same behavior given the same random bits.)



do NOT allow adjacent vertices update their colors in the same round:

$\Rightarrow$ $O(\Delta T)$ rounds

[Feng, Hayes, Y. '19]:

$O(T + \log n)$ rounds w.h.p.
(under some mild condition)

2-Phase Paradigm

for each vertex $v \in V$:

Phase I:

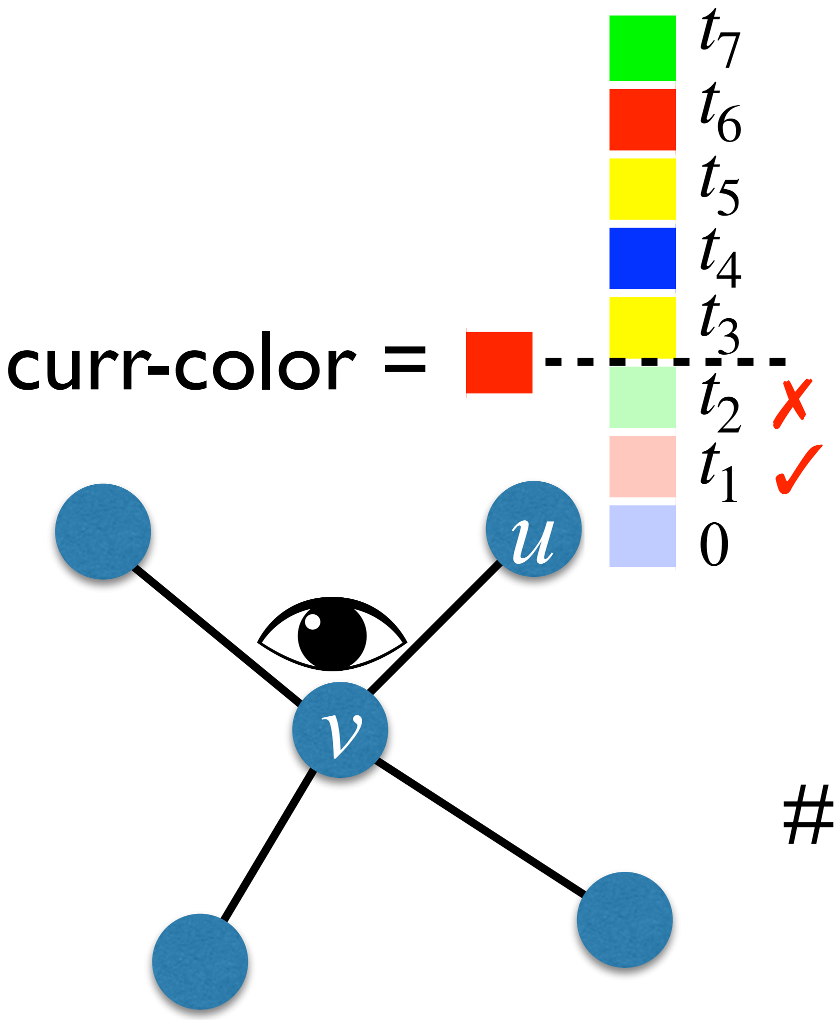
- locally generate all update times $0 < t_1 < t_2 < \cdots < t_{M_v} < T$ and proposed colors $c_1, c_2, \ldots, c_{M_v} \in [q]$;
- send the initial color and all $(t_i, c_i)_{1 \le i \le M_v}$ to all neighbors;

Phase II:

- For $i = 1, 2, \ldots, M_v$ do:
  once having received *enough information*:
  resolve the i-th update of v and send the result
  ("Accept / Reject") to all neighbors;

for each vertex $v \in V$:

- For $i = 1,2,\ldots,M_v$ do:
  once having received *enough information*:
  resolve the i-th update of v and send the result
  ("Accept / Reject") to all neighbors;



"enough info" to resolve
the i-th update at v: (t_i^v, c_i^v)

all adjacent updates before t_i^v
have been resolved and received by v

∃ a path $v_1, v_2, \ldots, v_L$

#rounds > L ⇒ $T > t_{i_1}^{v_1} > t_{i_2}^{v_2} > \cdots > t_{i_L}^{v_L} > 0$

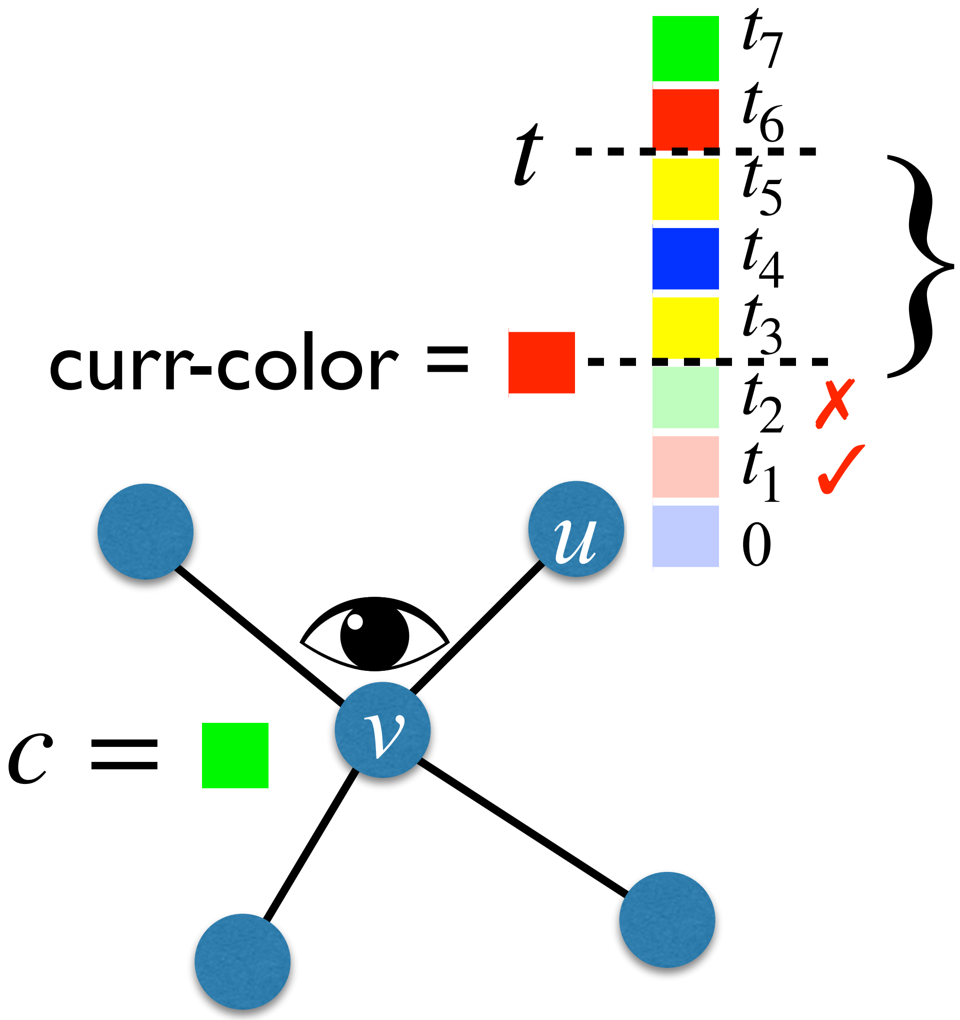
which occurs w.p. <$(eT/L)^L$

⇒ #rounds = $O(\Delta T + \log n)$ w.h.p.

Resolve Update In Advance

"enough info" to resolve the i-th update at v: (t, c)

If $c \notin \bigcup_{u \sim v} S_u(t)$:"Accept!"

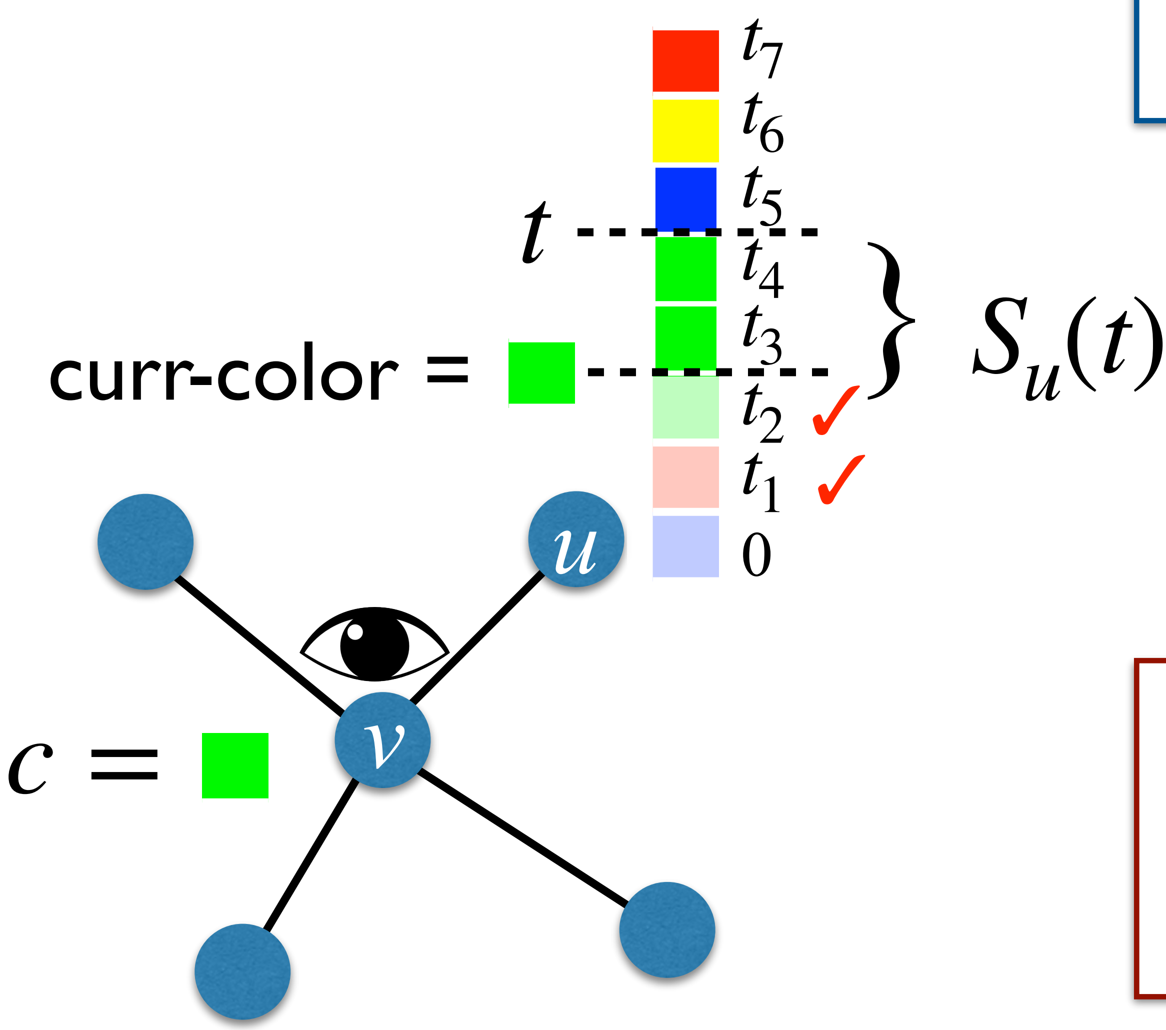


$S_u(t)$: set of possible colors of u at time t

Resolve Update In Advance

“enough info” to resolve the i-th update at v: (t, c)

If $c \notin \bigcup_{u \sim v} S_u(t)$:“Accept!”



: set of possible colors of u at time t

If $\exists u \sim v$ s.t. $S_u(t) = \{c\}$:
“Reject!”

to resolve the i-th update at v: (t, c)

Construct $S_u(t)$ for every neighbor u of v;

upon $c \notin \bigcup_{u \sim v} S_u(t)$:

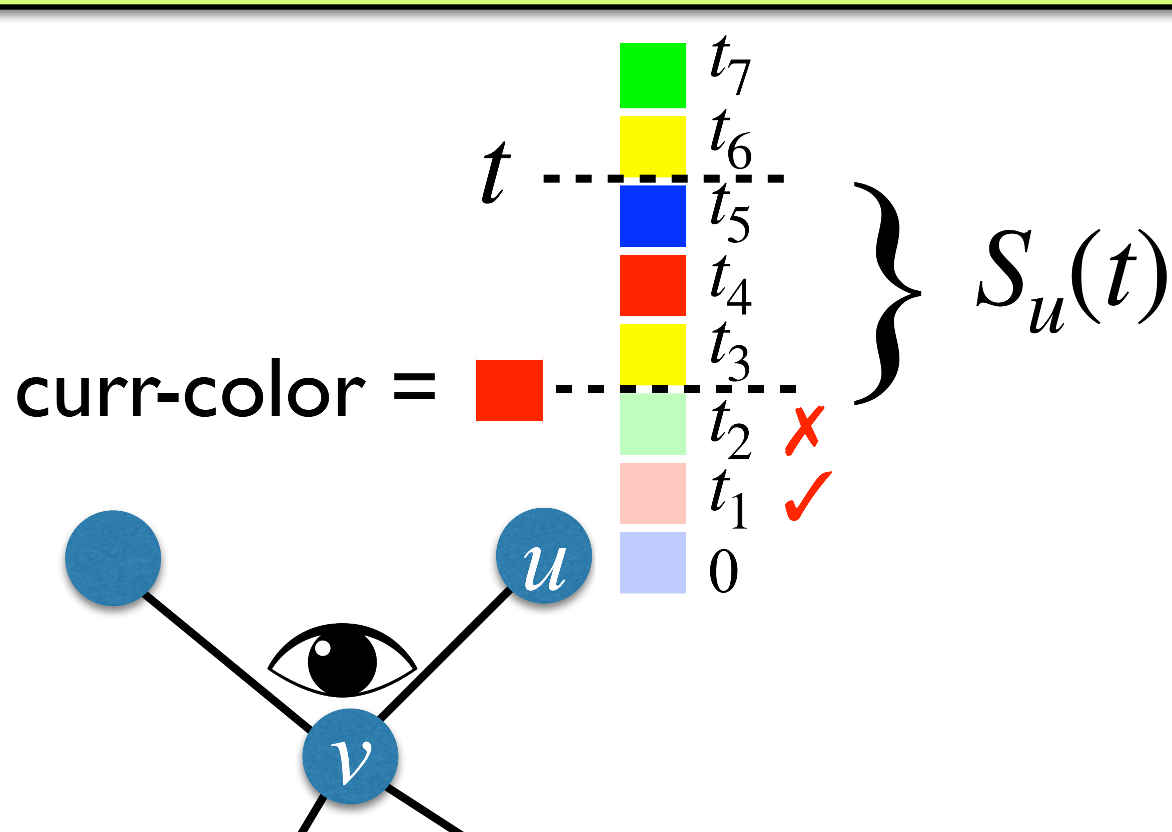
  send "Accept!" to all neighbors and $i{+}{+}$;

upon $\exists u \sim v$ s.t. $S_u(t) = \{c\}$:

  send "Reject!" to all neighbors and $i{+}{+}$;

upon receiving "Accept!" or "Reject!" from neighbor u:

  update $S_u(t)$ accordingly;



: current set of possible colors of u at time t

to resolve the i-th update at v: (t, c)

Construct $S_u(t)$ for every neighbor u of v;

upon $c \notin \bigcup_{u \sim v} S_u(t)$:

send "Accept!" to all neighbors and i++;

upon $\exists u \sim v$ s.t. $S_u(t) = \{c\}$:

send "Reject!" to all neighbors and i++;

upon receiving "Accept!" or "Reject!" from neighbor u:

update $S_u(t)$ accordingly;

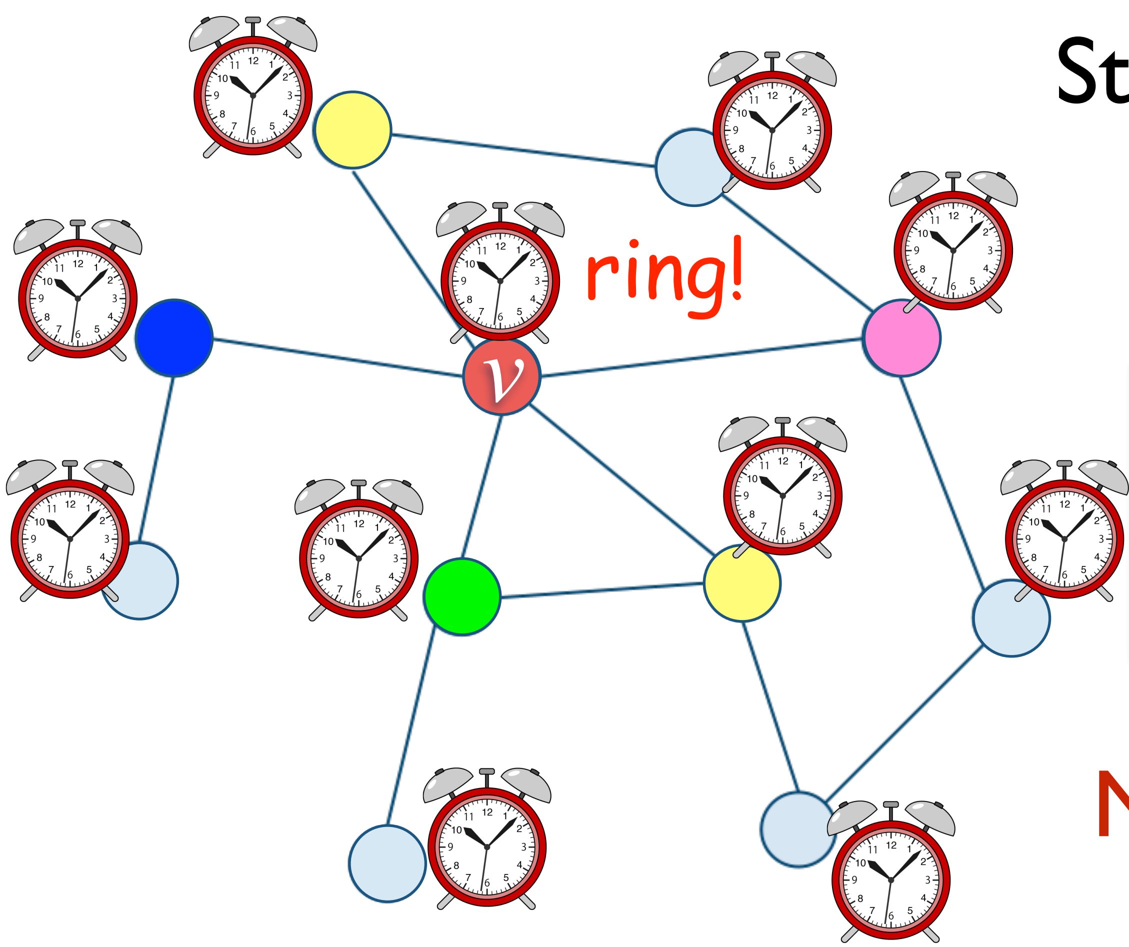
#round > L ⇒ ∃ a path $v_1, v_2, \ldots, v_L$: #paths $\leq \Delta^L$

$\Pr < O\left(\frac{T}{qL}\right)^L$ $\begin{cases} T > t_{i_1}^{v_1} > t_{i_2}^{v_2} > \cdots > t_{i_L}^{v_L} > 0 \\ \text{along the path: "good events" do not happen} \end{cases}$

$q > C\Delta$ for constant $C > 0$ ⇒ #rounds = O($T + \log n$) w.h.p.

The Metropolis Algorithm

Each vertex holds an independent rate-1 poisson clock.



Start from an arbitrary $X \in [q]^V$

When the clock at v rings:

> let $b = X_v$ and propose a random $c \in [q]$;
>
> change X_v to c with prob. $f^v_{b,c}(X_{N(v)})$;

Metropolis filter:

$$f^v_{b,c} : [q]^{N(v)} \to [0,1]$$

$b \in [q]$: current color of v

$c \in [q]$: proposed color of v

2-Phase Paradigm

for each vertex $v \in V$:

Phase I:

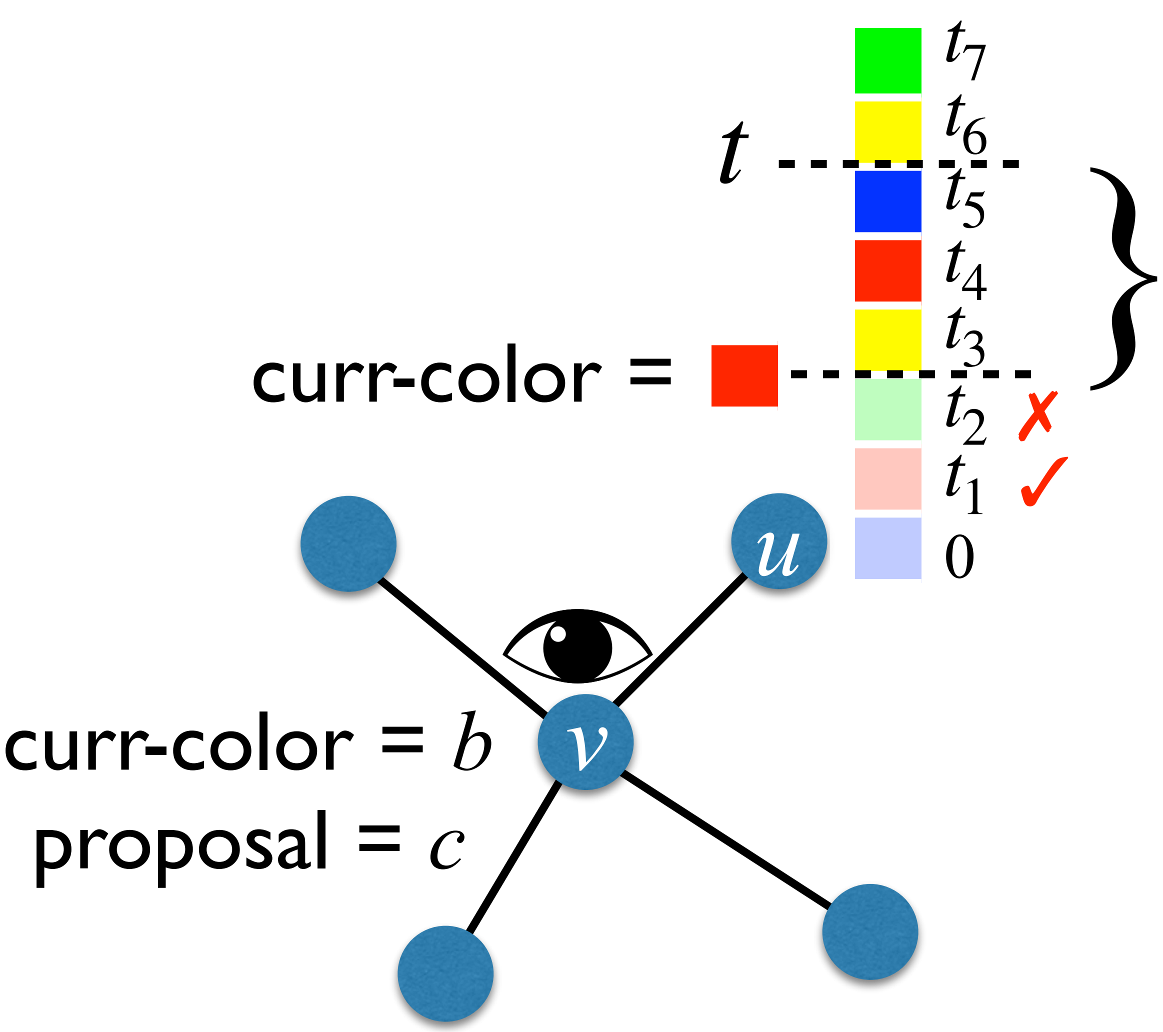
- locally generate all update times $0 < t_1 < t_2 < \cdots < t_{M_v} < T$ and proposed colors $c_1, c_2, \ldots, c_{M_v} \in [q]$;
- send the initial color and all $(t_i, c_i)_{1 \le i \le M_v}$ to all neighbors;

Phase II:

- For $i = 1,2,\ldots,M_v$ do:
  once having received *enough information*:
  resolve the i-th update of v and send the result
  ("Accept / Reject") to all neighbors;

to resolve the i-th update at v: (t, c)

- For $i = 1, 2, \ldots, M_v$ do:
  once having received enough information (to execute the Metropoli filter):
  resolve the i-th update of v and send the result
  ("Accept / Reject") to all neighbors;



$S_u(t)$: set of possible colors of u at time t

$$\forall \tau \in \bigotimes_{u \sim v} S_u(t)$$

$f^v_{b,c}(\tau)$ gives a biased coin

Idea: Couple all these coins!

to resolve the i-th update at v: (t, c)

Construct $S_u(t)$ for every neighbor u of v;

let b be v's current color and:

$$P_{\text{Acc}} \triangleq \min_{\tau \in \bigoplus_{u \sim v} S_u(t)} f_{b,c}(\tau);$$

$$P_{\text{Rej}} \triangleq 1 - \max_{\tau \in \bigoplus_{u \sim v} S_u(t)} f_{b,c}(\tau);$$

sample a uniform random $\beta \in [0,1]$;

upon $\beta \le P_{\text{Acc}}$:

send "Accept!" to all neighbors and $i{+}{+}$;

upon $\beta \ge 1 - P_{\text{Rej}}$:

send "Reject!" to all neighbors and $i{+}{+}$;

upon receiving "Accept!" or "Reject!" from neighbor u:

update $S_u(t)$ accordingly and recalculate P_{Acc} and P_{Rej};

Universal Distributed Simulation of Metropolis Algorithm

Metropolis Algorithm: continuous-time T

let $b = X_v$ and propose a random $c \in [q]$;

change X_v to c with prob. $f^v_{b,c}(X_{N(v)})$;

Lipschitz condition: $\exists$ constant $C > 0$:

$$\forall (u,v) \in E, \forall a, a', b \in [q] : \quad \mathbb{E}_c[\delta_{u,a,a'} f^v_{b,c}] < \frac{C}{\Delta}$$

where $$\delta_{u,a,a'} f^v_{b,c} \triangleq \max_{\substack{\sigma,\tau \text{ differ onl, at } u \\ \sigma_u = a, \tau_u = b}} |f^v_{b,c}(\sigma) - f^v_{b,c}(\tau)|$$

#rounds = $O(T + \log n)$ w.h.p.

model	Lipschitz condition	Necessary condition for mixing
q-coloring	∃ constant $C>0$ $q>C\Delta$	$q \geq \Delta+2$
Ising model with temperature *β*	∃ constant $C>0$ $1-\mathrm{e}^{-2\|\beta\|} < \frac{C}{\Delta}$	$1-\mathrm{e}^{-2\|\beta\|} < \frac{2}{\Delta}$
hardcore model with fugacity *λ*	∃ constant $C>0$ $\lambda < \frac{C}{\Delta}$	$\lambda < \frac{(\Delta-1)^{\Delta-1}}{(\Delta-2)^{\Delta}} \approx \frac{\mathrm{e}}{\Delta-2}$

Summary

- Universal distributed perfect simulation of Metropolis algorithms, with ideal parallelism under mild Lipschitz condition for Metropolis filter.
- **Open problem**: distributed simulation of general class of single-site Markov chains.

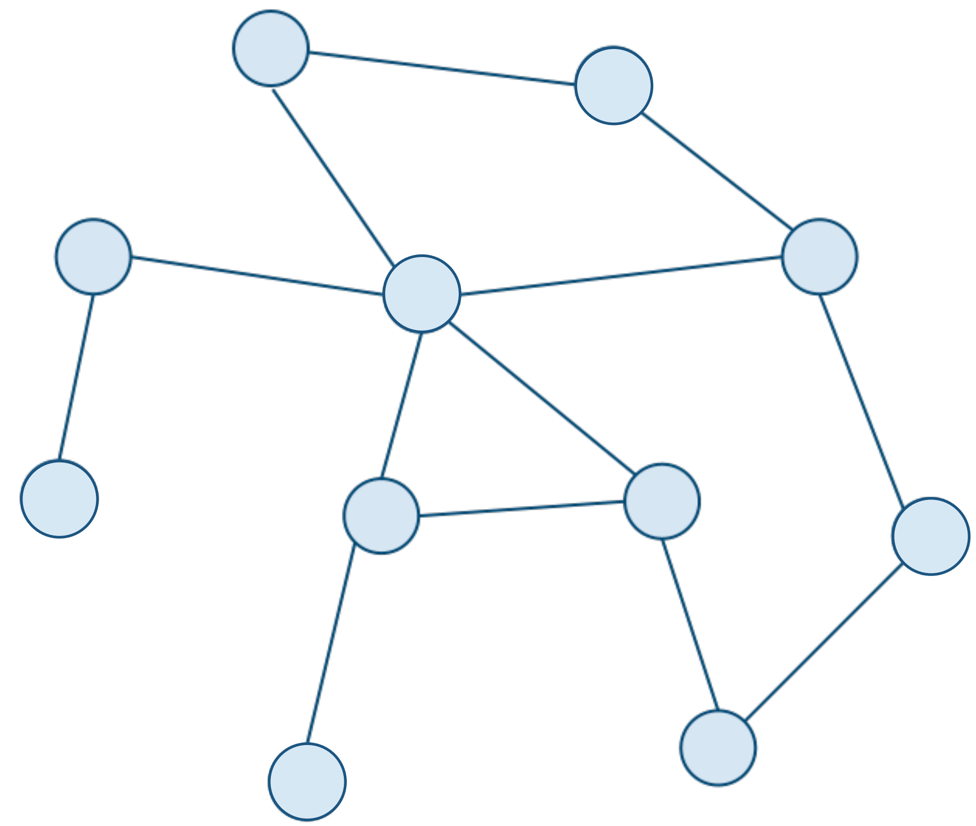
Outline

- Distributed Sampling Problem
  - Gibbs Distribution (distribution defined by local constraints)
- Algorithmic Ideas [Feng, Hayes, Y., '19]
  - *Local Metropolis Algorithm* [Feng, Sun, Y., PODC'17]
  - *LOCAL Jerrum-Valiant-Vazirani* [Feng, Y., PODC'18]
  - *Local Rejection Sampling* [Feng, Vishnoi, Y., STOC'19]
- Distributed Simulation of Metropolis

Local Computation

Locally Checkable Labeling (*LCL*) problems:

- CSPs with local constraints.
- Construct a feasible solution: vertex/edge coloring, Lovász local lemma
  - Find local optimum: MIS, MM
  - Approximate global optimum: maximum matching, minimum vertex cover, minimum dominating set

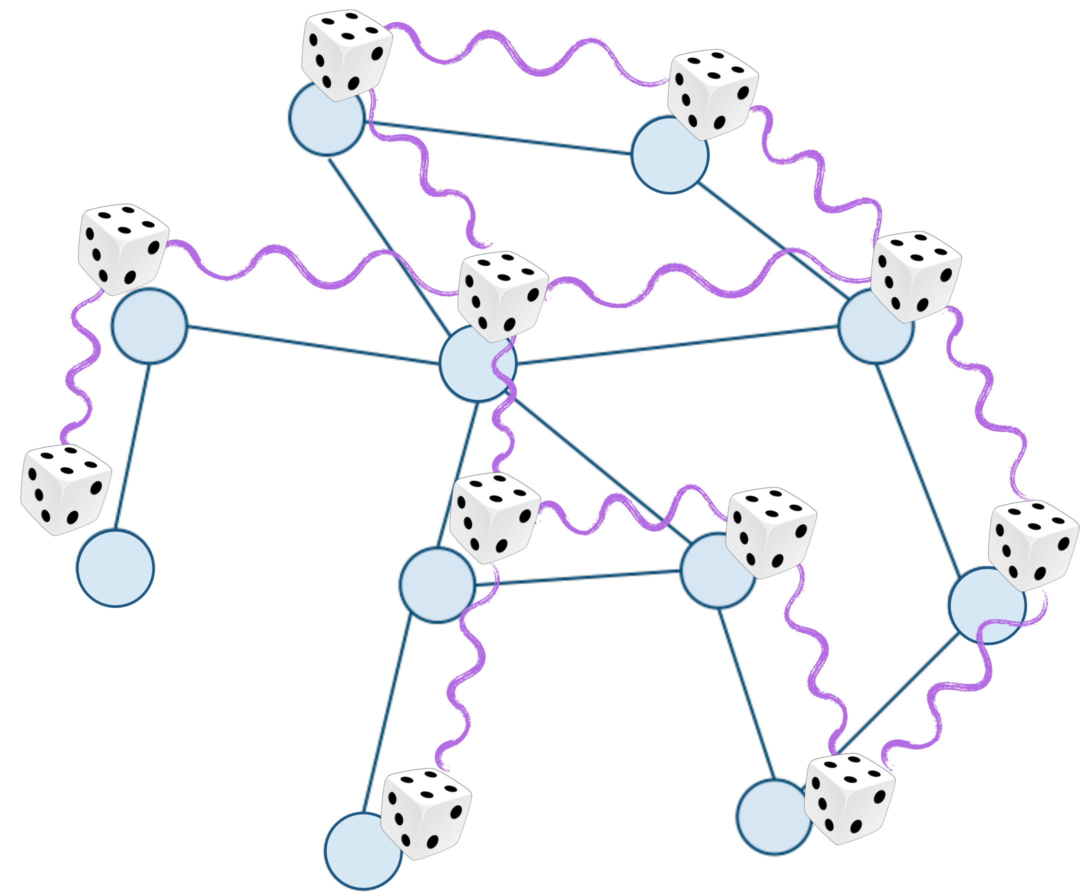


network $G(V,E)$

Quest: "Find a solution to the locally defined problem."

“What can be *sampled* locally?”

- CSP with local constraints.
- Sample a uniform random solution.
- Distribution μ (over solutions) described by local rules.
  - uniform LCL solution
  - Ising model / RBM / tensor network…



network $G(V,E)$

Quest: “Generate a sample from the locally defined distribution.”

Markov Random Fields

- Each vertex corresponds to a variable with finite domain $[q]$.
- Each edge $(u,v) \in E$ imposes a binary constraint:

$$A_{u,v} : [q]^2 \rightarrow \{0,1\}$$

- Gibbs distribution μ :

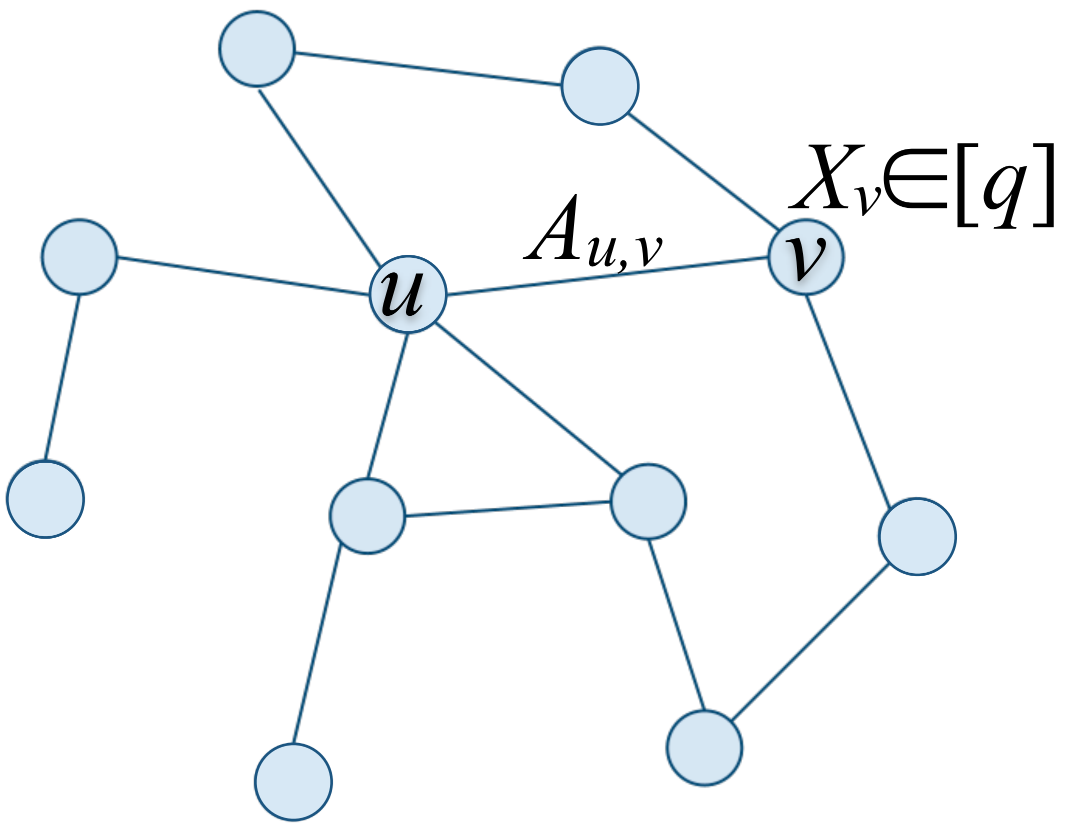
$\forall \sigma \in [q]^V$:

$$\mu(\sigma) \propto \prod_{(u,v)\in E} A_{u,v}(\sigma_u, \sigma_v)$$

- local conflict colorings:

[Fraigniaud, Heinrich, Kosowski '16]

network $G(V,E)$:



$\vec{X} \in [q]^V$ follows μ

Markov Random Fields

- Gibbs distribution μ :

$$\forall \sigma \in [q]^V : \quad \mu(\sigma) \propto \prod_{(u,v)\in E} A_{u,v}(\sigma_u, \sigma_v)$$

- vertex q-coloring:

$$A_{u,v} = \begin{bmatrix} 0 & & 1 \\ & 0 & \\ 1 & \ddots & \\ & & 0 \end{bmatrix}$$

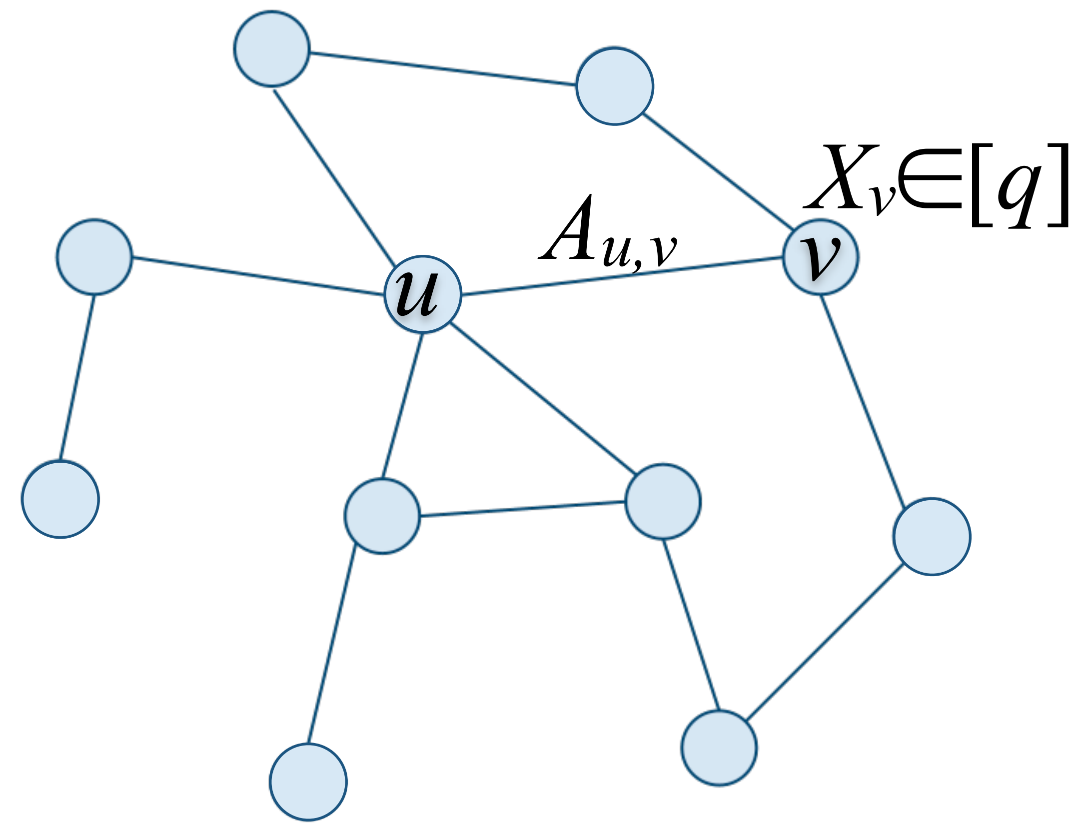
- independent set:

$$A_{u,v} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

- local conflict colorings: $A_{u,v} \in \{0,1\}^{q\times q}$

[Fraigniaud, Heinrich, Kosowski '16]

network $G(V,E)$:



$\vec{X} \in [q]^V$ follows μ

Markov Random Fields

- Each vertex corresponds to a variable with finite domain $[q]$.
- Each edge $(u,v)\in E$ imposes a binary constraint:

$$A_{u,v} : [q]^2 \to [0,1]$$

"soft" constraint

- Gibbs distribution μ :

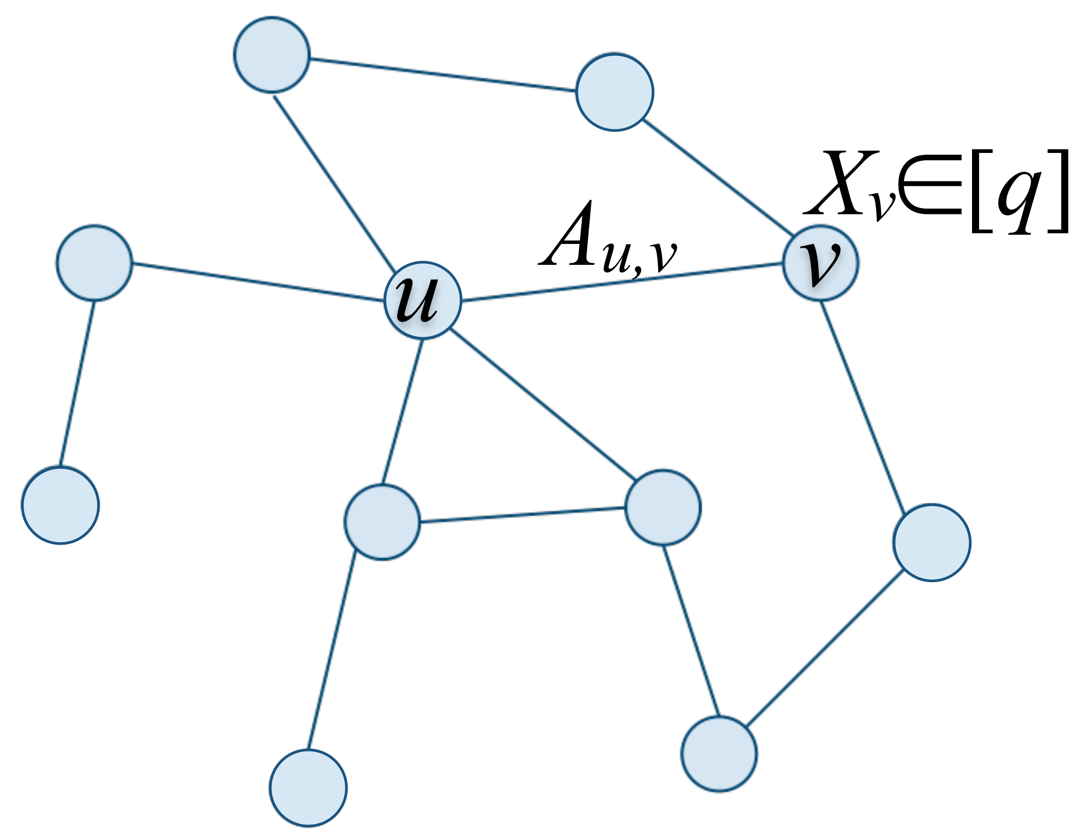
$\forall \sigma \in [q]^V$:

$$\mu(\sigma) \propto \prod_{(u,v)\in E} A_{u,v}(\sigma_u, \sigma_v)$$

- local conflict colorings:

[Fraigniaud, Heinrich, Kosowski '16]

network $G(V,E)$:



$\vec{X} \in [q]^V$ follows μ

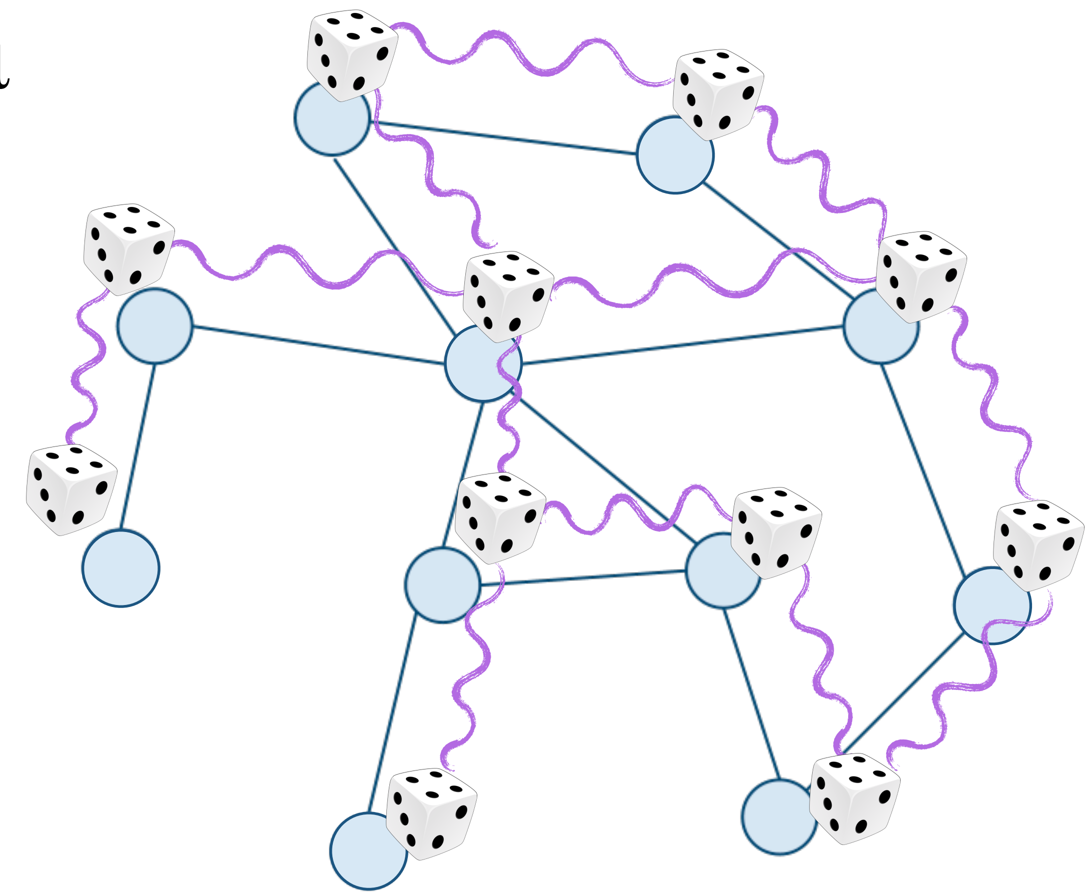
Distributed Sampling

- Instance: a Gibbs distribution μ
- Output: random $\boldsymbol{Y} \in [q]^V$
  - approx. sampling:

    $$d_{\mathrm{TV}}(\boldsymbol{Y}, \mu) \le \epsilon$$

  - perfect sampling:

    $$\boldsymbol{Y} \sim \mu$$



network $G(V,E)$

Empirical studies in machine learning:

[Kandasamy, *et al*, **AISTAT**'18]
[Dasklakis, *et al*, **NIPS**'18]
[De Sa, *et al*, **ICML**'16 **best paper**]
[De Sa, *et al*, **NIPS**'15]
[Ahmed, *et al*, **WSDM**'12]
[Gonzalez, *et al*, **AISTAT**'11]
[Yan, *et al*, **NIPS**'09]
[Smyth, *et al*, **NIPS**'09]
[Doshi-Velez, *et al*, **NIPS**'09]
[Newman, *et al*, **NIPS**'08]

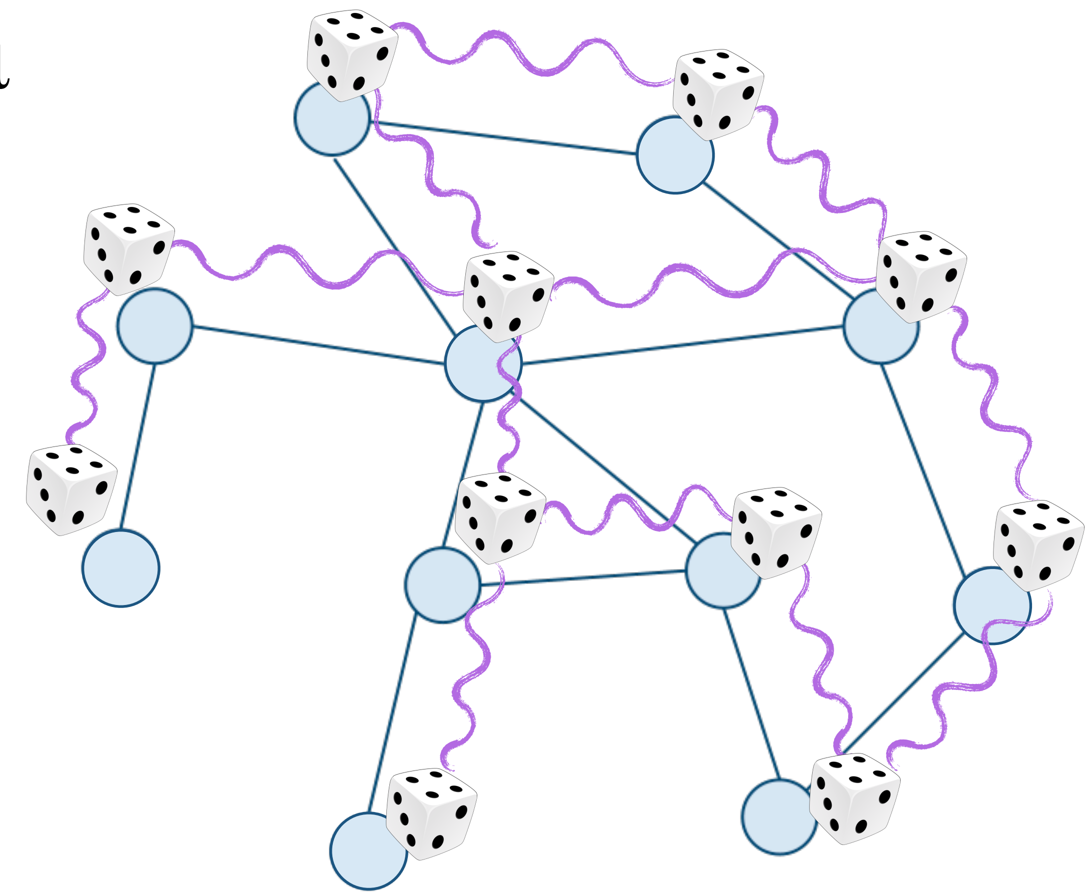
Distributed Sampling

- Instance: a Gibbs distribution μ
- Output: random $\boldsymbol{Y} \in [q]^V$
  - approx. sampling:

$$d_{\mathrm{TV}}(\boldsymbol{Y}, \mu) \leq \epsilon$$

  - perfect sampling:

$$\boldsymbol{Y} \sim \mu$$

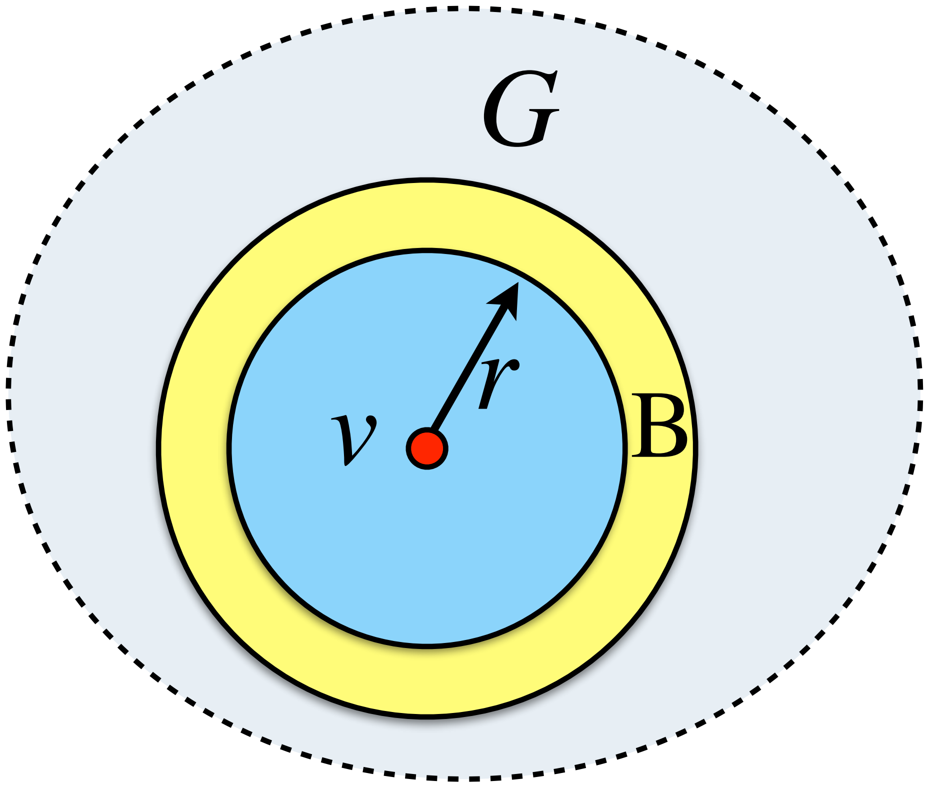


network $G(V,E)$

[Feng, Sun, Y. '17]:

Easy regime	Hard regime
• $O(\Delta \log n)$-round is easy • $O(\log n)$-round is possible • $\Omega(\log n)$-round is necessary	• can be $\Omega(Diam)$-hard when $Diam = n^{\Omega(1)}$

Phase Transition



Corerelation decay:

$$\forall \sigma_B, \tau_B \in [q]^B :$$

$$d_{\mathrm{TV}}(\mu_v(\cdot \mid \sigma_B), \mu_v(\cdot \mid \tau_B))$$

$$\leq \exp(-\Omega(r))$$

Hard regime: there is long-range correlation

- (Δ-1)-coloring on triangle-free graph
- independent set when Δ=6 or higher

} $\Omega(Diam)$-hard

Easy regime: various forms of correlation decays

- Dobrushin-Shlosman condition
- Uniqueness condition (spatial mixing)
- …

Outline

- Distributed Sampling Problem
  - Gibbs Distribution (distribution defined by local constraints)
- Algorithmic Ideas
  - *Local Metropolis Algorithm* [Feng, Sun, Y., PODC'17]
  - *LOCAL Jerrum-Valiant-Vazirani* [Feng, Y., PODC'18]
  - *Local Rejection Sampling* [Feng, Vishnoi, Y., STOC'19]
- Distributed Simulation of Metropolis

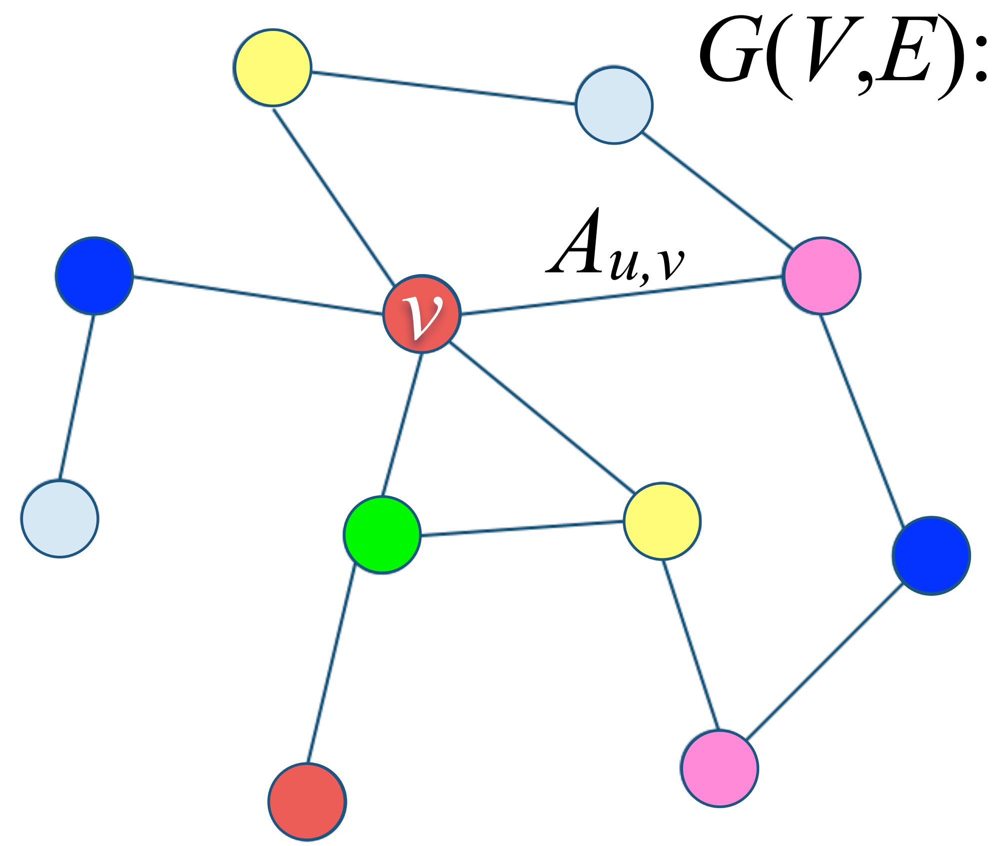
Single-Site Markov Chain

Metropolis for *q*-coloring:

starting from an arbitrary $X \in [q]^V$

at each step :

> pick a uniform random vertex v;
>
> propose a random color $c \in [q]$;
>
> change $X(v)$ to c if it's proper;

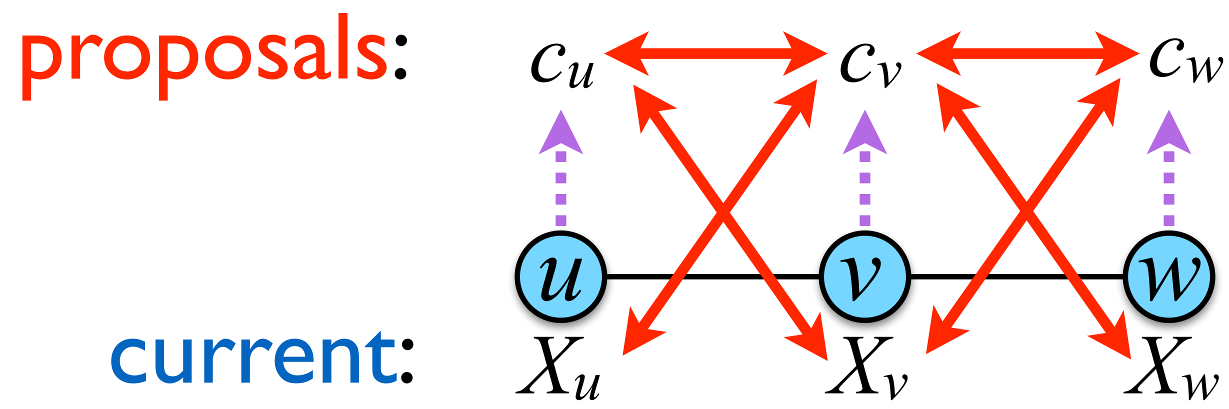


Metropolis for general MRF:

> pick a uniform random vertex v;
>
> propose to change $X(v)$ to a random color $c \in [q]$;
>
> accept the change with probability $\min\left\{1, \frac{\mu(X')}{\mu(X)}\right\} = \min\left\{1, \prod_{u \in N(v)} \frac{A_{u,v}(X(u), c)}{A_{u,v}(X(u), X(v))}\right\}$

[Bubley, Dyer, 97]: path-coupling works ➡ *mixing* in $O(n \log n)$ steps

The *Local Metropolis* Algorithm



starting from an arbitrary $X \in [q]^V$, at each step:

each vertex $v \in V$ independently proposes a random $c_v \in [q]$;

each edge $(u,v) \in E$ passes its test independently with probability:

$$A_{u,v}(X_u, c_v) \cdot A_{u,v}(c_u, X_v) \cdot A_{u,v}(c_u, c_v) \ ;$$

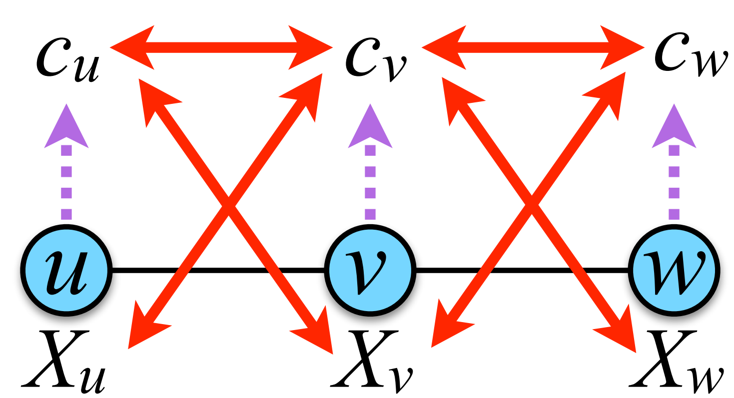
each vertex $v \in V$ accepts to change to its proposed value c_v
if *all incident edges pass their test*;

- converge to the **correct** Gibbs distribution μ. [Feng, Sun, Y. '17]

The *Local Metropolis* Algorithm

proposals:

current:



For q-coloring, at each step:

each vertex $v \in V$ independently proposes a random color $c_v \in [q]$;
each vertex $v \in V$ accepts to change to its proposed color c_v if:

$$X_u \neq c_v \wedge c_u \neq X_v \wedge c_u \neq c_v;$$

[Feng, Sun, Y. '17], [Fischer, Ghaffari '18], [Feng, Hayes, Yin '18]:

- Converges in $O(\log n)$ rounds when:

path-coupling works for (sequential) Metropolis chain ← Dobrushin-Shlosman condition

$(2+\delta)\Delta$-coloring

LOCAL Jerrum-Valiant-Vazirani

[Jerrum, Valiant, Vazirani '86]: (for self-reducible problems)

approximate counting → (Poly-time TM) → perfect sampling

LOCAL JVV [Feng, Y. '18]: (for self-reducible problems)

correlation decay ("strong spatial mixing") → (unbounded msg/comput.) → LOCAL approx. inference → (local JVV reduction) → SLOCAL perfect sampling → (network decomposition) → LOCAL perfect sampling

$O(\log^3 n)$ rounds

- $(2+\delta)\Delta$-coloring; 1.733Δ-coloring on triangle-free graph;
- **Conjecture**: $(1+\delta)\Delta$-coloring

Local Rejection Sampling

$\forall \sigma \in [q]^V$: $\mu(\sigma) \propto \prod_{e=(u,v)\in E} A_u(\sigma_u, \sigma_v)$ where $A_e : [q]^2 \to [0,1]$

a Moser-Tardos style algorithm [Feng, Vishnoi, Y. '19]:

each $v \in V$ ind. samples a random $\sigma_v \in [q]$;

each $e=(u,v) \in E$ samples $F_e \in \{0,1\}$ ind. with $\Pr[F_e = 0] = A_e(\sigma_u,\sigma_v)$;

while $\exists e \in E$ s.t. $F_e = 1$ do:

resample σ_v for all $v \in R \triangleq \bigcup_{e\in E:F_e=1} e$;

for each $e=(u,v) \in E$ that $e \cap R \neq \emptyset$, resample $F_e \in \{0,1\}$ ind. as:

$$\Pr[F_e = 0] = \begin{cases} A_e(\sigma_u, \sigma_v) & u, v \in R \quad \textbf{\textit{(internal edge)}} \\ \dfrac{A_e(\sigma_u, \sigma_v)}{A_e(\sigma_u, \sigma_v^{\text{old}})} \min A_e(\sigma_u, \cdot\,) & u \notin R, v \in R \quad \textbf{\textit{(boundary edge)}} \end{cases}$$

each $v \in V$ returns σ_v;

Local Rejection Sampling

[Feng, Vishnoi, Y. '19], [Feng, Guo, Y. '19]

a Moser-Tardos style algorithm:

- perfect sampling, Las Vegas
- parallel/distributed (CONGEST)
- $O(\log n)$-round when converge
- works for dynamic input
- require stronger types of correlation decay:
  - $O(\Delta^2)$-coloring (for a variant of the algorithm)

each $v \in V$ ind. samples a random $\sigma_v \in [q]$;
each $e=(u,v) \in E$ samples $F_e \in \{0,1\}$ ind. with $\Pr[F_e = 0] = A_e(\sigma_u, \sigma_v)$;
while $\exists e \in E$ s.t. $F_e = 1$ do:
    resample σ_v for all $v \in R \triangleq \bigcup_{e \in E: F_e = 1} e$;
    for each $e=(u,v) \in E$ that $e \cap R \neq \emptyset$, resample $F_e \in \{0,1\}$ ind. as:

$$\Pr[F_e = 0] = \begin{cases} A_e(\sigma_u, \sigma_v) & u, v \in R \quad \textbf{\textit{(internal edge)}} \\ \frac{A_e(\sigma_u, \sigma_v)}{A_e(\sigma_u, \sigma_v^{\text{old}})} \min A_e(\sigma_u, \cdot\,) & u \notin R, v \in R \quad \textbf{\textit{(boundary edge)}} \end{cases}$$

each $v \in V$ returns σ_v;

	Features/Limitations	Fast regimes
Local Metropolis	• synchronous parallel Markov chain • Monte Carlo sampling • CONGEST model	• path-coupling works for sequential process (Dobrushin-Shlosman cond.) • $(2+\delta)\Delta$-coloring
LOCAL JVV	• perfect sampling • abuses LOCAL model • $O(\log^3 n)$ rounds	• needs only necessary correlation decay • **conjecture**: $(1+\delta)\Delta$-coloring
Local Rejection Sampling	• Moser-Tardos style • Las Vegas, perfect sampling • CONGEST model • works on dynamic input	• requires faster correlation decay • $O(\Delta^2)$-coloring

	Features/Limitations	Fast regimes
Universal Simulation of Metropolis	• Monte Carlo sampling • CONGEST model	• as long as sequential Metropolis algorithm has $O(n \log n)$ mixing time
LOCAL JVV	• perfect sampling • abuses LOCAL model • $O(\log^3 n)$ rounds	• needs only necessary correlation decay • **conjecture**: $(1+\delta)\Delta$-coloring
Local Rejection Sampling	• Moser-Tardos style • Las Vegas, perfect sampling • CONGEST model • works on dynamic input	• requires faster correlation decay • $O(\Delta^2)$-coloring

Thank you!

Feng, Guo, Y. **Perfect sampling from spatial mixing.** arXiv:1907.06033.

Feng, Hayes, Y. **Distributed Metropolis Sampler with Optimal Parallelism.** arxiv:1904.00943

Feng, Hayes, Y. **Distributed Sampling Almost-Uniform Graph Coloring with Fewer Colors.** arxiv: 1802.06953.

Feng, Vishnoi, Y. **Dynamic Sampling from graphical models.** STOC'19. arxiv: 1807.06481.

Feng, Y. **On local distributed sampling and counting.** PODC'18. arxiv: 1802.06686.

Feng, Sun, Y. **What can be sampled locally?** PODC'17. arxiv: 1702.00142.