Markov-chain Monte Carlo algorithms for studying cycle spaces, with some applications to graph colouring

Tim Garoni

Department of Mathematics and Statistics
University of Melbourne (?)

April 20, 2011
Monash Discrete Maths Seminar

Collaborators/References

1. Qingquan Liu, Youjin Deng, TG
   Loop models in three dimensions,
   in preparation.
2. Qingquan Liu, Youjin Deng, TG, and Jesús Salas
   Irreducible Markov-chain Monte Carlo algorithm for zero-temperature Potts antiferromagnets,
   in preparation.
3. Qingquan Liu, Youjin Deng, and TG,
   Worm Monte Carlo study of the honeycomb-lattice loop model,
   Nucl. Phys. B 846, 283-315 (2011).
4. Wei Zhang, TG, and Youjin Deng,
   A worm algorithm for the fully-packed loop model,
   Nucl. Phys. B 814, 461-484 (2009).
5. Youjin Deng, TG, and Alan D. Sokal,
   Dynamic Critical Behavior of the Worm Algorithm for the Ising Model,
   Phys. Rev. Lett. 99, 110601 (2007).

Outline

- Loops & Worms

Outline

- Loops & Worms
  - $O(n)$-loop models

Outline

- Loops & Worms
  - $O(n)$-loop models
  - Basics of the Markov-chain Monte Carlo method

Outline

- Loops & Worms
  - $O(n)$-loop models
  - Basics of the Markov-chain Monte Carlo method
  - Worm algorithm for $O(n)$-loop models

Outline

- Loops & Worms
  - $O(n)$-loop models
  - Basics of the Markov-chain Monte Carlo method
  - Worm algorithm for $O(n)$-loop models
  - Proof of irreducibility in fully-packed limit

Outline

- Loops & Worms
    - $O(n)$-loop models
    - Basics of the Markov-chain Monte Carlo method
    - Worm algorithm for $O(n)$-loop models
    - Proof of irreducibility in fully-packed limit
- Antiferromagnetic Potts models at zero temperature

Outline

- Loops & Worms
  - $O(n)$-loop models
  - Basics of the Markov-chain Monte Carlo method
  - Worm algorithm for $O(n)$-loop models
  - Proof of irreducibility in fully-packed limit
- Antiferromagnetic Potts models at zero temperature
  - Review of Wang-Swendsen-Kotecký (WSK) algorithm

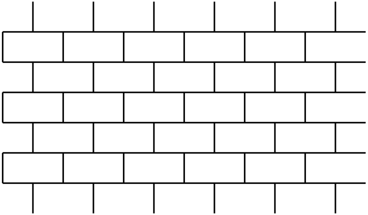
Outline

- Loops & Worms
  - $O(n)$-loop models
  - Basics of the Markov-chain Monte Carlo method
  - Worm algorithm for $O(n)$-loop models
  - Proof of irreducibility in fully-packed limit
- Antiferromagnetic Potts models at zero temperature
  - Review of Wang-Swendsen-Kotecký (WSK) algorithm
  - (Non)-irreducibility of WSK algorithm at zero temperature

Outline

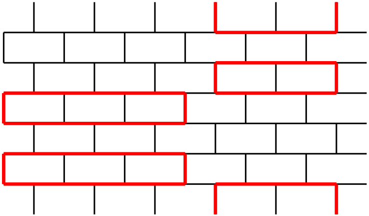
- Loops & Worms
  - $O(n)$-loop models
  - Basics of the Markov-chain Monte Carlo method
  - Worm algorithm for $O(n)$-loop models
  - Proof of irreducibility in fully-packed limit
- Antiferromagnetic Potts models at zero temperature
  - Review of Wang-Swendsen-Kotecký (WSK) algorithm
  - (Non)-irreducibility of WSK algorithm at zero temperature
  - Using worm algorithm instead

Loop models



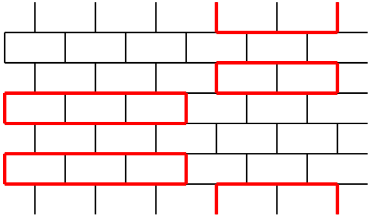
- Consider a finite graph G
  - Physicists love the honeycomb lattice
  - (Regular map of type $\{6, 3\}$ on the torus)

Loop models



- Consider a finite graph G
  - Physicists love the honeycomb lattice
  - (Regular map of type $\{6,3\}$ on the torus)
- Take state space to be $\mathcal{C}(G)$
  - Cycle space of G
  - $\{A \subset E : (V,A) \text{ has no odd vertices}\}$
  - = all disjoint unions of loops on G when $\Delta(G) \leq 3$

Loop models

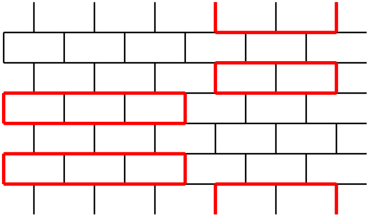


- Consider a finite graph G
  - Physicists love the honeycomb lattice
  - (Regular map of type $\{6, 3\}$ on the torus)
- Take state space to be $\mathcal{C}(G)$
  - Cycle space of G
  - $\{A \subset E : (V, A) \text{ has no odd vertices}\}$
  - = all disjoint unions of loops on G when $\Delta(G) \leq 3$
- Assign probabilities via

$$\phi_{G,n,x}(A) = \frac{n^{c(A)} x^{|A|}}{Z_{G,n,x}}, \qquad A \in \mathcal{C}(G)$$

- $c(A)$ is cyclomatic number, $|A|$ the number of bonds

Loop models

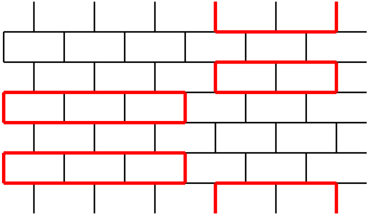


- Consider a finite graph G
  - Physicists love the honeycomb lattice
  - (Regular map of type $\{6,3\}$ on the torus)
- Take state space to be $\mathcal{C}(G)$
  - Cycle space of G
  - $\{A \subset E : \ (V,A)$ has no odd vertices$\}$
  - = all disjoint unions of loops on G when $\Delta(G) \leq 3$
- Assign probabilities via

$$\phi_{G,n,x}(A) = \frac{n^{c(A)}\, x^{|A|}}{Z_{G,n,x}}, \qquad A \in \mathcal{C}(G)$$

- $c(A)$ is cyclomatic number, $|A|$ the number of bonds
- Mathematical physics want to compute expectations wrt $\phi_{G,n,x}(\cdot)$

Loop models

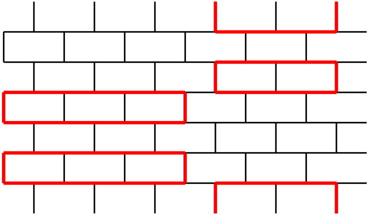


- Consider a finite graph G
  - Physicists love the honeycomb lattice
  - (Regular map of type $\{6, 3\}$ on the torus)
- Take state space to be $\mathcal{C}(G)$
  - Cycle space of G
  - $\{A \subset E : (V, A) \text{ has no odd vertices}\}$
  - = all disjoint unions of loops on G when $\Delta(G) \leq 3$
- Assign probabilities via

$$\phi_{G,n,x}(A) = \frac{n^{c(A)} x^{|A|}}{Z_{G,n,x}}, \qquad A \in \mathcal{C}(G)$$

- $c(A)$ is cyclomatic number, $|A|$ the number of bonds
- Mathematical physics want to compute expectations wrt $\phi_{G,n,x}(\cdot)$
  - As $|V| \to \infty$ such models display *critical phenomena*: correlations on all scales, fractals,...

Loop models

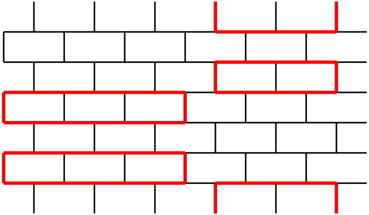


- Consider a finite graph G
  - Physicists love the honeycomb lattice
  - (Regular map of type $\{6, 3\}$ on the torus)
- Take state space to be $\mathcal{C}(G)$
  - Cycle space of G
  - $\{A \subset E : (V, A) \text{ has no odd vertices}\}$
  - = all disjoint unions of loops on G when $\Delta(G) \leq 3$
- Assign probabilities via

$$\phi_{G,n,x}(A) = \frac{n^{c(A)} x^{|A|}}{Z_{G,n,x}}, \qquad A \in \mathcal{C}(G)$$

- $c(A)$ is cyclomatic number, $|A|$ the number of bonds
- Mathematical physics want to compute expectations wrt $\phi_{G,n,x}(\cdot)$
  - As $|V| \to \infty$ such models display *critical phenomena*: correlations on all scales, fractals,...
- If G is large its infeasible to calculate $Z_{G,n,x} = \sum_{A \in \mathcal{C}(G)} n^{c(A)} x^{|A|}$

Loop models



- Consider a finite graph G
  - Physicists love the honeycomb lattice
  - (Regular map of type $\{6, 3\}$ on the torus)
- Take state space to be $\mathcal{C}(G)$
  - Cycle space of G
  - $\{A \subset E : (V, A) \text{ has no odd vertices}\}$
  - = all disjoint unions of loops on G when $\Delta(G) \leq 3$
- Assign probabilities via

$$\phi_{G,n,x}(A) = \frac{n^{c(A)} x^{|A|}}{Z_{G,n,x}}, \qquad A \in \mathcal{C}(G)$$

- $c(A)$ is cyclomatic number, $|A|$ the number of bonds
- Mathematical physics want to compute expectations wrt $\phi_{G,n,x}(\cdot)$
  - As $|V| \to \infty$ such models display *critical phenomena*: correlations on all scales, fractals,...
- If G is large its infeasible to calculate $Z_{G,n,x} = \sum_{A \in \mathcal{C}(G)} n^{c(A)} x^{|A|}$
- So how can we sample from $\phi_{G,n,x}(\cdot)$?

Elementary theory of finite Markov chains

- Consider:
  - A finite set S (state space)
  - A stochastic matrix $P : S \times S \to [0,1]$ (transition matrix)
  - A probability vector $\alpha : S \to [0,1]$ (initial distribution)

Elementary theory of finite Markov chains

- Consider:
  - A finite set S (state space)
  - A stochastic matrix $P : S \times S \to [0, 1]$ (transition matrix)
  - A probability vector $\alpha : S \to [0, 1]$ (initial distribution)
- Defines a Markov chain $X_1, X_2, \ldots$ on S
  - $\mathbb{P}(X_0 = s) = \alpha(s)$
  - $\mathbb{P}(X_{n+1} = s' | X_n = s) = P(s \to s')$

Elementary theory of finite Markov chains

- Consider:
  - A finite set S (state space)
  - A stochastic matrix $P : S \times S \to [0,1]$ (transition matrix)
  - A probability vector $\alpha : S \to [0,1]$ (initial distribution)
- Defines a Markov chain $X_1, X_2, \ldots$ on S
  - $\mathbb{P}(X_0 = s) = \alpha(s)$
  - $\mathbb{P}(X_{n+1} = s' | X_n = s) = P(s \to s')$
- Probability vector π is a stationary distribution if $\pi P = \pi$

Elementary theory of finite Markov chains

- Consider:
  - A finite set S (state space)
  - A stochastic matrix $P : S \times S \to [0,1]$ (transition matrix)
  - A probability vector $\alpha : S \to [0,1]$ (initial distribution)
- Defines a Markov chain $X_1, X_2, \ldots$ on S
  - $\mathbb{P}(X_0 = s) = \alpha(s)$
  - $\mathbb{P}(X_{n+1} = s' | X_n = s) = P(s \to s')$
- Probability vector π is a stationary distribution if $\pi P = \pi$
  - Don't need to know Z to prove stationarity

Elementary theory of finite Markov chains

- Consider:
  - A finite set S (state space)
  - A stochastic matrix $P : S \times S \to [0,1]$ (transition matrix)
  - A probability vector $\alpha : S \to [0,1]$ (initial distribution)
- Defines a Markov chain $X_1, X_2, \ldots$ on S
  - $\mathbb{P}(X_0 = s) = \alpha(s)$
  - $\mathbb{P}(X_{n+1} = s' | X_n = s) = P(s \to s')$
- Probability vector π is a stationary distribution if $\pi P = \pi$
  - Don't need to know Z to prove stationarity
- Detailed balance ($\pi_s P_{ss'} = \pi_{s'} P_{s's}$) implies stationarity

Elementary theory of finite Markov chains

- Consider:
  - A finite set S (state space)
  - A stochastic matrix $P : S \times S \to [0,1]$ (transition matrix)
  - A probability vector $\alpha : S \to [0,1]$ (initial distribution)
- Defines a Markov chain $X_1, X_2, \ldots$ on S
  - $\mathbb{P}(X_0 = s) = \alpha(s)$
  - $\mathbb{P}(X_{n+1} = s' | X_n = s) = P(s \to s')$
- Probability vector π is a stationary distribution if $\pi P = \pi$
  - Don't need to know Z to prove stationarity
- Detailed balance ($\pi_s P_{ss'} = \pi_{s'} P_{s's}$) implies stationarity
- P is irreducible if $s \leftrightarrow s'$ for all $s, s' \in S$

Elementary theory of finite Markov chains

- Consider:
  - A finite set S (state space)
  - A stochastic matrix $P : S \times S \to [0,1]$ (transition matrix)
  - A probability vector $\alpha : S \to [0,1]$ (initial distribution)
- Defines a Markov chain $X_1, X_2, \ldots$ on S
  - $\mathbb{P}(X_0 = s) = \alpha(s)$
  - $\mathbb{P}(X_{n+1} = s' | X_n = s) = P(s \to s')$
- Probability vector π is a stationary distribution if $\pi P = \pi$
  - Don't need to know Z to prove stationarity
- Detailed balance ($\pi_s P_{ss'} = \pi_{s'} P_{s's}$) implies stationarity
- P is irreducible if $s \leftrightarrow s'$ for all $s, s' \in S$
- If P is irreducible it has a unique stationary distribution π

Elementary theory of finite Markov chains

- Consider:
  - A finite set S (state space)
  - A stochastic matrix $P : S \times S \to [0, 1]$ (transition matrix)
  - A probability vector $\alpha : S \to [0, 1]$ (initial distribution)
- Defines a Markov chain $X_1, X_2, \ldots$ on S
  - $\mathbb{P}(X_0 = s) = \alpha(s)$
  - $\mathbb{P}(X_{n+1} = s' | X_n = s) = P(s \to s')$
- Probability vector π is a stationary distribution if $\pi P = \pi$
  - Don't need to know Z to prove stationarity
- Detailed balance ($\pi_s P_{ss'} = \pi_{s'} P_{s's}$) implies stationarity
- P is irreducible if $s \leftrightarrow s'$ for all $s, s' \in S$
- If P is irreducible it has a unique stationary distribution π

Theorem (Ergodic Theorem)

If P is irreducible with stationary distribution π then for any initial distribution

$$\frac{1}{N}\sum_{n=1}^{N} f(X_n) \xrightarrow[N\to\infty]{} \sum_{s\in S} \pi(s) f(s) \qquad a.s.$$

Metropolized Chains

- We want a transition matrix P in detailed balance with π

Metropolized Chains

- We want a transition matrix P in detailed balance with π
- Suppose we have a symmetric transition matrix Ψ

Metropolized Chains

- We want a transition matrix P in detailed balance with π
- Suppose we have a symmetric transition matrix Ψ
- Construct P from Ψ by introducing acceptance probabilities
  - Accept proposed transitions $\Psi(s \to s')$ with probability $a(s \to s')$

Metropolized Chains

- We want a transition matrix P in detailed balance with π
- Suppose we have a symmetric transition matrix Ψ
- Construct P from Ψ by introducing acceptance probabilities
  - Accept proposed transitions $\Psi(s \to s')$ with probability $a(s \to s')$
  - $P(s \to s') = a(s \to s')\,\Psi(s \to s')$

Metropolized Chains

- We want a transition matrix P in detailed balance with π
- Suppose we have a symmetric transition matrix Ψ
- Construct P from Ψ by introducing acceptance probabilities
  - Accept proposed transitions $\Psi(s \to s')$ with probability $a(s \to s')$
  - $P(s \to s') = a(s \to s')\,\Psi(s \to s')$
  - Demanding P be in detailed balance with π implies

$$\pi(s)\,a(s \to s')\,\Psi(s \to s') = \pi(s')\,a(s' \to s)\,\Psi(s' \to s)$$

Metropolized Chains

- We want a transition matrix P in detailed balance with π
- Suppose we have a symmetric transition matrix Ψ
- Construct P from Ψ by introducing acceptance probabilities
  - Accept proposed transitions $\Psi(s \to s')$ with probability $a(s \to s')$
  - $P(s \to s') = a(s \to s')\,\Psi(s \to s')$
  - Demanding P be in detailed balance with π implies

$$\pi(s)\,a(s \to s')\,\Psi(s \to s') = \pi(s')\,a(s' \to s)\,\Psi(s' \to s)$$

  - $a(s \to s') = \min(1, \pi(s')/\pi(s))$ is maximal solution

Metropolized Chains

- We want a transition matrix P in detailed balance with π
- Suppose we have a symmetric transition matrix Ψ
- Construct P from Ψ by introducing acceptance probabilities
  - Accept proposed transitions $\Psi(s \to s')$ with probability $a(s \to s')$
  - $P(s \to s') = a(s \to s')\,\Psi(s \to s')$
  - Demanding P be in detailed balance with π implies

$$\pi(s)\,a(s \to s')\,\Psi(s \to s') = \pi(s')\,a(s' \to s)\,\Psi(s' \to s)$$

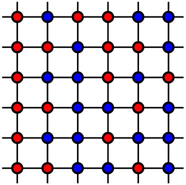
  - $a(s \to s') = \min(1, \pi(s')/\pi(s))$ is maximal solution
- Irreducibility of P needs to be checked case-by-case

Metropolized Chains

- We want a transition matrix P in detailed balance with π
- Suppose we have a symmetric transition matrix Ψ
- Construct P from Ψ by introducing acceptance probabilities
  - Accept proposed transitions $\Psi(s \to s')$ with probability $a(s \to s')$
  - $P(s \to s') = a(s \to s')\,\Psi(s \to s')$
  - Demanding P be in detailed balance with π implies

$$\pi(s)\,a(s \to s')\,\Psi(s \to s') = \pi(s')\,a(s' \to s)\,\Psi(s' \to s)$$

  - $a(s \to s') = \min(1, \pi(s')/\pi(s))$ is maximal solution
- Irreducibility of P needs to be checked case-by-case



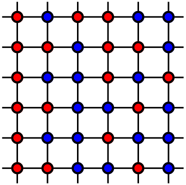
- E.g. Ising model on $G = (V, E)$
  - $\mathcal{S} = \{-1, 1\}^V$
  - $\pi(\sigma) \propto e^{-H(\sigma)}$

Metropolized Chains

- We want a transition matrix P in detailed balance with π
- Suppose we have a symmetric transition matrix Ψ
- Construct P from Ψ by introducing acceptance probabilities
  - Accept proposed transitions $\Psi(s \to s')$ with probability $a(s \to s')$
  - $P(s \to s') = a(s \to s')\,\Psi(s \to s')$
  - Demanding P be in detailed balance with π implies

$$\pi(s)\,a(s \to s')\,\Psi(s \to s') = \pi(s')\,a(s' \to s)\,\Psi(s' \to s)$$

  - $a(s \to s') = \min(1, \pi(s')/\pi(s))$ is maximal solution
- Irreducibility of P needs to be checked case-by-case



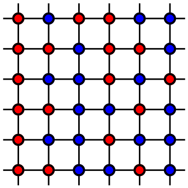
- E.g. Ising model on $G = (V, E)$
  - $\mathcal{S} = \{-1, 1\}^V$
  - $\pi(\sigma) \propto e^{-H(\sigma)}$
- Simplest Ψ for Ising model is

Metropolized Chains

- We want a transition matrix P in detailed balance with π
- Suppose we have a symmetric transition matrix Ψ
- Construct P from Ψ by introducing acceptance probabilities
  - Accept proposed transitions $\Psi(s \to s')$ with probability $a(s \to s')$
  - $P(s \to s') = a(s \to s')\,\Psi(s \to s')$
  - Demanding P be in detailed balance with π implies

$$\pi(s)\,a(s \to s')\,\Psi(s \to s') = \pi(s')\,a(s' \to s)\,\Psi(s' \to s)$$

  - $a(s \to s') = \min(1, \pi(s')/\pi(s))$ is maximal solution
- Irreducibility of P needs to be checked case-by-case



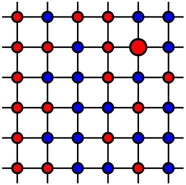
- E.g. Ising model on $G = (V, E)$
  - $\mathcal{S} = \{-1, 1\}^V$
  - $\pi(\sigma) \propto e^{-H(\sigma)}$
- Simplest Ψ for Ising model is
  - Pick $v \in V$ uniformly at random

Metropolized Chains

- We want a transition matrix P in detailed balance with π
- Suppose we have a symmetric transition matrix Ψ
- Construct P from Ψ by introducing acceptance probabilities
  - Accept proposed transitions $\Psi(s \to s')$ with probability $a(s \to s')$
  - $P(s \to s') = a(s \to s')\,\Psi(s \to s')$
  - Demanding P be in detailed balance with π implies

$$\pi(s)\,a(s \to s')\,\Psi(s \to s') = \pi(s')\,a(s' \to s)\,\Psi(s' \to s)$$

  - $a(s \to s') = \min(1, \pi(s')/\pi(s))$ is maximal solution
- Irreducibility of P needs to be checked case-by-case



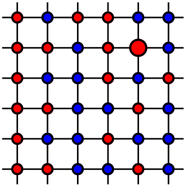
- E.g. Ising model on $G = (V, E)$
  - $\mathcal{S} = \{-1, 1\}^V$
  - $\pi(\sigma) \propto e^{-H(\sigma)}$
- Simplest Ψ for Ising model is
  - Pick $v \in V$ uniformly at random

Metropolized Chains

- We want a transition matrix P in detailed balance with π
- Suppose we have a symmetric transition matrix Ψ
- Construct P from Ψ by introducing acceptance probabilities
  - Accept proposed transitions $\Psi(s \to s')$ with probability $a(s \to s')$
  - $P(s \to s') = a(s \to s')\,\Psi(s \to s')$
  - Demanding P be in detailed balance with π implies

$$\pi(s)\,a(s \to s')\,\Psi(s \to s') = \pi(s')\,a(s' \to s)\,\Psi(s' \to s)$$

  - $a(s \to s') = \min(1, \pi(s')/\pi(s))$ is maximal solution
- Irreducibility of P needs to be checked case-by-case



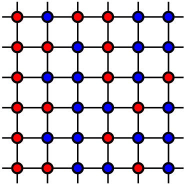
- E.g. Ising model on $G = (V, E)$
  - $\mathcal{S} = \{-1, 1\}^V$
  - $\pi(\sigma) \propto e^{-H(\sigma)}$
- Simplest Ψ for Ising model is
  - Pick $v \in V$ uniformly at random
  - Propose $\sigma_v \to -\sigma_v$

Metropolized Chains

- We want a transition matrix P in detailed balance with π
- Suppose we have a symmetric transition matrix Ψ
- Construct P from Ψ by introducing acceptance probabilities
  - Accept proposed transitions $\Psi(s \to s')$ with probability $a(s \to s')$
  - $P(s \to s') = a(s \to s')\,\Psi(s \to s')$
  - Demanding P be in detailed balance with π implies

$$\pi(s)\,a(s \to s')\,\Psi(s \to s') = \pi(s')\,a(s' \to s)\,\Psi(s' \to s)$$

  - $a(s \to s') = \min(1, \pi(s')/\pi(s))$ is maximal solution
- Irreducibility of P needs to be checked case-by-case



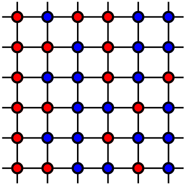
- E.g. Ising model on $G = (V, E)$
  - $\mathcal{S} = \{-1, 1\}^V$
  - $\pi(\sigma) \propto e^{-H(\sigma)}$
- Simplest Ψ for Ising model is
  - Pick $v \in V$ uniformly at random
  - Propose $\sigma_v \to -\sigma_v$

Metropolized Chains

- We want a transition matrix P in detailed balance with π
- Suppose we have a symmetric transition matrix Ψ
- Construct P from Ψ by introducing acceptance probabilities
  - Accept proposed transitions $\Psi(s \to s')$ with probability $a(s \to s')$
  - $P(s \to s') = a(s \to s')\,\Psi(s \to s')$
  - Demanding P be in detailed balance with π implies

$$\pi(s)\,a(s \to s')\,\Psi(s \to s') = \pi(s')\,a(s' \to s)\,\Psi(s' \to s)$$

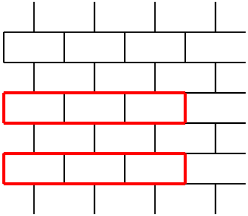
  - $a(s \to s') = \min(1, \pi(s')/\pi(s))$ is maximal solution
- Irreducibility of P needs to be checked case-by-case



- E.g. Ising model on $G = (V, E)$
  - $\mathcal{S} = \{-1, 1\}^V$
  - $\pi(\sigma) \propto e^{-H(\sigma)}$
- Simplest Ψ for Ising model is
  - Pick $v \in V$ uniformly at random
  - Propose $\sigma_v \to -\sigma_v$
- Local moves from $\mathcal{S} \to \mathcal{S}$ easy to construct

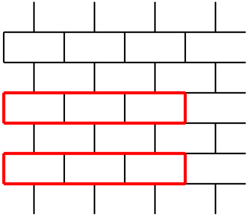
Worm chains

- Local moves on $\mathcal{C}(G)$ not so obvious



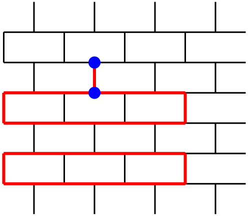
Worm chains

- Local moves on $\mathcal{C}(G)$ not so obvious
- Enlarge $\mathcal{C}(G)$ to include two defects (odd vertices)



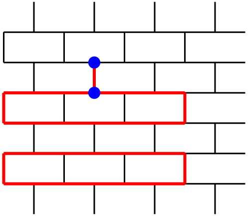
Worm chains

- Local moves on $\mathcal{C}(G)$ not so obvious
- Enlarge $\mathcal{C}(G)$ to include two defects (odd vertices)



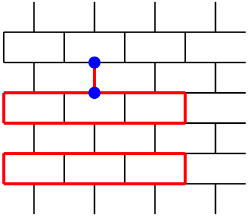
Worm chains

- Local moves on $\mathcal{C}(G)$ not so obvious
- Enlarge $\mathcal{C}(G)$ to include two defects (odd vertices)
- Move the defects via random walk



Worm chains

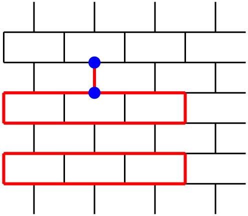
- Local moves on $\mathcal{C}(G)$ not so obvious
- Enlarge $\mathcal{C}(G)$ to include two defects (odd vertices)
- Move the defects via random walk



- Let ∂A be the set of all odd vertices in (V, A)

Worm chains

- Local moves on $\mathcal{C}(G)$ not so obvious
- Enlarge $\mathcal{C}(G)$ to include two defects (odd vertices)
- Move the defects via random walk

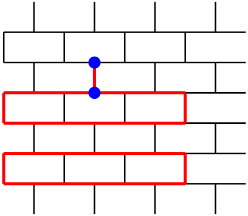


- Let ∂A be the set of all odd vertices in (V, A)
- State space of worm algorithm is

$$\mathcal{S}(G) = \{(A, u, v) : \partial A = \{u, v\}\}$$

Worm chains

- Local moves on $\mathcal{C}(G)$ not so obvious
- Enlarge $\mathcal{C}(G)$ to include two defects (odd vertices)
- Move the defects via random walk



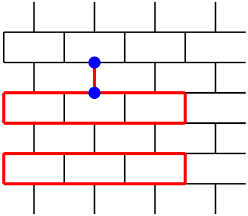
- Let ∂A be the set of all odd vertices in (V, A)
- State space of worm algorithm is

$$\mathcal{S}(G) = \{(A, u, v) : \partial A = \{u, v\}\}$$

- When $u = v$ we take $\partial A = \emptyset$

Worm chains

- Local moves on $\mathcal{C}(G)$ not so obvious
- Enlarge $\mathcal{C}(G)$ to include two defects (odd vertices)
- Move the defects via random walk



- Let ∂A be the set of all odd vertices in (V, A)
- State space of worm algorithm is

$$\mathcal{S}(G) = \{(A, u, v) : \partial A = \{u, v\}\}$$

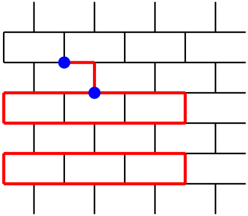
- When $u = v$ we take $\partial A = \emptyset$

- Proposal matrix

$$\Psi_{n,x}[(A, u, v) \to (A \triangle uu', u', v)] = \frac{1}{2\, d(u)}$$

Worm chains

- Local moves on $\mathcal{C}(G)$ not so obvious
- Enlarge $\mathcal{C}(G)$ to include two defects (odd vertices)
- Move the defects via random walk



- Let ∂A be the set of all odd vertices in (V, A)
- State space of worm algorithm is

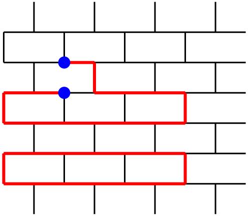
$$\mathcal{S}(G) = \{(A, u, v) : \partial A = \{u, v\}\}$$

- When $u = v$ we take $\partial A = \emptyset$
- Proposal matrix

$$\Psi_{n,x}[(A, u, v) \to (A \triangle uu', u', v)] = \frac{1}{2\,d(u)}$$

Worm chains

- Local moves on $\mathcal{C}(G)$ not so obvious
- Enlarge $\mathcal{C}(G)$ to include two defects (odd vertices)
- Move the defects via random walk



- Let ∂A be the set of all odd vertices in (V, A)
- State space of worm algorithm is

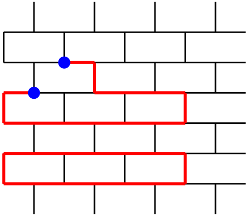
$$\mathcal{S}(G) = \{(A, u, v) : \partial A = \{u, v\}\}$$

- When $u = v$ we take $\partial A = \emptyset$
- Proposal matrix

$$\Psi_{n,x}[(A, u, v) \to (A \triangle uu', u', v)] = \frac{1}{2\, d(u)}$$

Worm chains

- Local moves on $\mathcal{C}(G)$ not so obvious
- Enlarge $\mathcal{C}(G)$ to include two defects (odd vertices)
- Move the defects via random walk



- Let ∂A be the set of all odd vertices in (V, A)
- State space of worm algorithm is

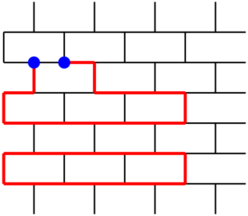
$$\mathcal{S}(G) = \{(A, u, v) : \partial A = \{u, v\}\}$$

- When $u = v$ we take $\partial A = \emptyset$
- Proposal matrix

$$\Psi_{n,x}[(A, u, v) \to (A \triangle uu', u', v)] = \frac{1}{2\, d(u)}$$

Worm chains

- Local moves on $\mathcal{C}(G)$ not so obvious
- Enlarge $\mathcal{C}(G)$ to include two defects (odd vertices)
- Move the defects via random walk



- Let ∂A be the set of all odd vertices in (V, A)
- State space of worm algorithm is

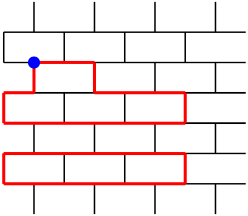
$$\mathcal{S}(G) = \{(A, u, v) : \partial A = \{u, v\}\}$$

- When $u = v$ we take $\partial A = \emptyset$
- Proposal matrix

$$\Psi_{n,x}[(A, u, v) \to (A \triangle uu', u', v)] = \frac{1}{2\, d(u)}$$

Worm chains

- Local moves on $\mathcal{C}(G)$ not so obvious
- Enlarge $\mathcal{C}(G)$ to include two defects (odd vertices)
- Move the defects via random walk



- Let ∂A be the set of all odd vertices in (V, A)
- State space of worm algorithm is

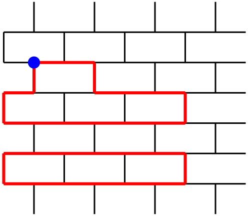
$$\mathcal{S}(G) = \{(A, u, v) : \partial A = \{u, v\}\}$$

- When $u = v$ we take $\partial A = \emptyset$
- Proposal matrix

$$\Psi_{n,x}[(A, u, v) \to (A \triangle uu', u', v)] = \frac{1}{2\,d(u)}$$

Worm chains

- Local moves on $\mathcal{C}(G)$ not so obvious
- Enlarge $\mathcal{C}(G)$ to include two defects (odd vertices)
- Move the defects via random walk



- Let ∂A be the set of all odd vertices in (V, A)
- State space of worm algorithm is

$$\mathcal{S}(G) = \{(A, u, v) : \partial A = \{u, v\}\}$$

- When $u = v$ we take $\partial A = \emptyset$
- Proposal matrix

$$\Psi_{n,x}[(A, u, v) \to (A \triangle uu', u', v)] = \frac{1}{2\, d(u)}$$

- Use Metropolis to construct $P_{n,x}$ in detailed balance with

$$\pi_{n,x}(A, u, v) \propto n^{c(A)}\, x^{|A|}$$

Worm transition matrix

$$P_{n,x}[(A,u,v) \to (A \triangle uu', u', v)] = P_{n,x}[(A,v,u) \to (A \triangle uu', v, u')]$$

$$= \frac{1}{2\,d(u)} \begin{cases} \min(1, x\,n) & uu' \notin A \text{ and } u \leftrightarrow u' \text{ in } (V,A) \\ \min(1, x) & uu' \notin A \text{ and } u \not\leftrightarrow u' \text{ in } (V,A) \\ \min(1, 1/n\,x) & uu' \in A \text{ and } u \leftrightarrow u' \text{ in } (V, A \setminus uu') \\ \min(1, 1/x) & uu' \in A \text{ and } u \not\leftrightarrow u' \text{ in } (V, A \setminus uu') \end{cases}$$

Worm transition matrix

$$P_{n,x}[(A,u,v) \to (A \triangle uu', u', v)] = P_{n,x}[(A,v,u) \to (A \triangle uu', v, u')]$$
$$= \frac{1}{2\,d(u)} \begin{cases} \min(1, x\,n) & uu' \notin A \text{ and } u \leftrightarrow u' \text{ in } (V,A) \\ \min(1, x) & uu' \notin A \text{ and } u \not\leftrightarrow u' \text{ in } (V,A) \\ \min(1, 1/n\,x) & uu' \in A \text{ and } u \leftrightarrow u' \text{ in } (V, A \setminus uu') \\ \min(1, 1/x) & uu' \in A \text{ and } u \not\leftrightarrow u' \text{ in } (V, A \setminus uu') \end{cases}$$

- Let $\overline{\pi}_{n,x}$ = restriction of $\pi_{n,x}$ to $\{(A,u,v) \in \mathcal{S} : u = v\} = \mathcal{C}(G) \times V$

Worm transition matrix

$$P_{n,x}[(A,u,v) \to (A \triangle uu', u', v)] = P_{n,x}[(A,v,u) \to (A \triangle uu', v, u')]$$

$$= \frac{1}{2\,d(u)} \begin{cases} \min(1, x\,n) & uu' \notin A \text{ and } u \leftrightarrow u' \text{ in } (V,A) \\ \min(1, x) & uu' \notin A \text{ and } u \not\leftrightarrow u' \text{ in } (V,A) \\ \min(1, 1/n\,x) & uu' \in A \text{ and } u \leftrightarrow u' \text{ in } (V, A \setminus uu') \\ \min(1, 1/x) & uu' \in A \text{ and } u \not\leftrightarrow u' \text{ in } (V, A \setminus uu') \end{cases}$$

- Let $\overline{\pi}_{n,x}$ = restriction of $\pi_{n,x}$ to $\{(A,u,v) \in \mathcal{S} : u = v\} = \mathcal{C}(G) \times V$
  - $\overline{\pi}_{n,x}(A,v,v) = \phi_{n,x}(A)/V$
  - So $\langle X \rangle_{\overline{\pi}_{n,x}} = \langle X \rangle_{\phi_{n,x}}$ for all $X : \mathcal{C}(G) \to \mathbb{R}$

Worm transition matrix

$$P_{n,x}[(A,u,v)\to(A\triangle uu',u',v)] = P_{n,x}[(A,v,u)\to(A\triangle uu',v,u')]$$
$$= \frac{1}{2\,d(u)}\begin{cases}\min(1,x\,n) & uu'\notin A \text{ and } u\leftrightarrow u' \text{ in } (V,A)\\ \min(1,x) & uu'\notin A \text{ and } u\not\leftrightarrow u' \text{ in } (V,A)\\ \min(1,1/n\,x) & uu'\in A \text{ and } u\leftrightarrow u' \text{ in } (V,A\setminus uu')\\ \min(1,1/x) & uu'\in A \text{ and } u\not\leftrightarrow u' \text{ in } (V,A\setminus uu')\end{cases}$$

- Let $\overline{\pi}_{n,x}$ = restriction of $\pi_{n,x}$ to $\{(A,u,v)\in\mathcal{S} : u=v\} = \mathcal{C}(G)\times V$
  - $\overline{\pi}_{n,x}(A,v,v) = \phi_{n,x}(A)/V$
  - So $\langle X\rangle_{\overline{\pi}_{n,x}} = \langle X\rangle_{\phi_{n,x}}$ for all $X : \mathcal{C}(G)\to\mathbb{R}$
- Only observe $P_{n,x}$ chain when $u=v$

Worm transition matrix

$$P_{n,x}[(A,u,v) \to (A \triangle uu', u', v)] = P_{n,x}[(A,v,u) \to (A \triangle uu', v, u')]$$

$$= \frac{1}{2\,d(u)} \begin{cases} \min(1, x\,n) & uu' \notin A \text{ and } u \leftrightarrow u' \text{ in } (V,A) \\ \min(1, x) & uu' \notin A \text{ and } u \not\leftrightarrow u' \text{ in } (V,A) \\ \min(1, 1/n\,x) & uu' \in A \text{ and } u \leftrightarrow u' \text{ in } (V, A \setminus uu') \\ \min(1, 1/x) & uu' \in A \text{ and } u \not\leftrightarrow u' \text{ in } (V, A \setminus uu') \end{cases}$$

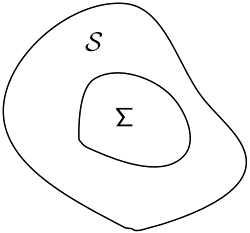
- Let $\overline{\pi}_{n,x}$ = restriction of $\pi_{n,x}$ to $\{(A,u,v) \in \mathcal{S} : u = v\} = \mathcal{C}(G) \times V$
  - $\overline{\pi}_{n,x}(A,v,v) = \phi_{n,x}(A)/V$
  - So $\langle X \rangle_{\overline{\pi}_{n,x}} = \langle X \rangle_{\phi_{n,x}}$ for all $X : \mathcal{C}(G) \to \mathbb{R}$
- Only observe $P_{n,x}$ chain when $u = v$
  - Get new chain, $\overline{P}_{n,x}$, in detailed balance with $\overline{\pi}_{n,x}$

Induced Markov chain on subspace

- Consider an irreducible P on a finite state space $\mathcal{S}$
- In detailed balance with π

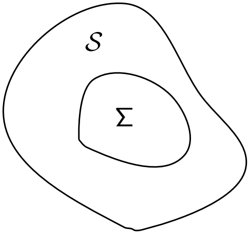
Induced Markov chain on subspace

- Consider an irreducible P on a finite state space $\mathcal{S}$
- In detailed balance with π
- Only observe the process when $s \in \Sigma \subseteq \mathcal{S}$



Induced Markov chain on subspace

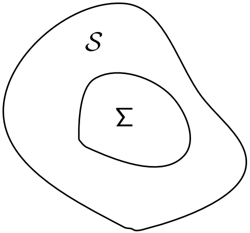
- Consider an irreducible P on a finite state space $\mathcal{S}$
- In detailed balance with π
- Only observe the process when $s \in \Sigma \subseteq \mathcal{S}$
- New process is irreducible Markov chain on Σ



Induced Markov chain on subspace

- Consider an irreducible P on a finite state space $\mathcal{S}$
- In detailed balance with π
- Only observe the process when $s \in \Sigma \subseteq \mathcal{S}$
- New process is irreducible Markov chain on Σ
- Transition matrix is

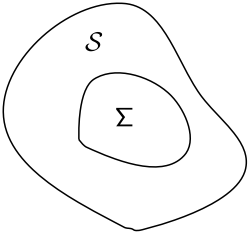
$$(\overline{P})_{ss'} := (P)_{ss'} + \sum_{n=0}^{\infty} \sum_{s_0, s_1, \ldots, s_n \in \overline{\Sigma}} (P)_{ss_0} \prod_{l=1}^{n} (P)_{s_{l-1}s_l} (P)_{s_n s'}$$



Induced Markov chain on subspace

- Consider an irreducible P on a finite state space $\mathcal{S}$
- In detailed balance with π
- Only observe the process when $s \in \Sigma \subseteq \mathcal{S}$
- New process is irreducible Markov chain on Σ
- Transition matrix is

$$(\overline{P})_{ss'} := (P)_{ss'} + \sum_{n=0}^{\infty} \sum_{s_0, s_1, \ldots, s_n \in \overline{\Sigma}} (P)_{ss_0} \prod_{l=1}^{n} (P)_{s_{l-1}s_l} (P)_{s_n s'}$$

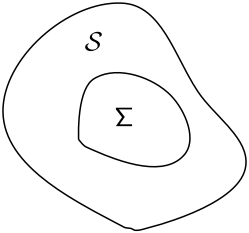


- $\overline{P}$ is in detailed balance with

$$\overline{\pi}_s = \frac{\pi_s}{\sum_{s' \in \Sigma} \pi_{s'}}, \qquad s \in \Sigma$$

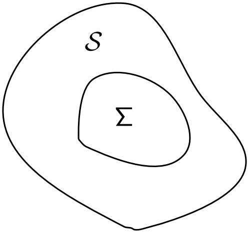
Rejection-free algorithm

- We don't care about states in $\overline{\Sigma} := \mathcal{S} \setminus \Sigma$



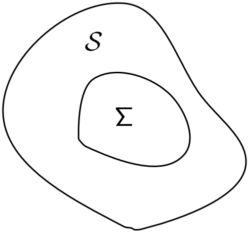
Rejection-free algorithm

- We don't care about states in $\overline{\Sigma} := \mathcal{S} \setminus \Sigma$
- Therefore we can safely avoid rejections when $(A, u, v) \in \overline{\Sigma}$



Rejection-free algorithm

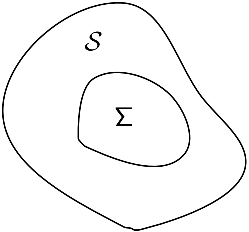
- We don't care about states in $\overline{\Sigma} := \mathcal{S} \setminus \Sigma$
- Therefore we can safely avoid rejections when $(A, u, v) \in \overline{\Sigma}$
  - Use rejection-free chain instead



$$(P')_{ss'} := \begin{cases} (P)_{ss'} & s \in \Sigma \\ \dfrac{(P)_{ss'}}{1-(P)_{ss}} & s \in \overline{\Sigma}, s' \neq s \\ 0 & s \in \overline{\Sigma}, s' = s \end{cases}$$

Rejection-free algorithm

- We don't care about states in $\overline{\Sigma} := \mathcal{S} \setminus \Sigma$
- Therefore we can safely avoid rejections when $(A, u, v) \in \overline{\Sigma}$



- Use rejection-free chain instead

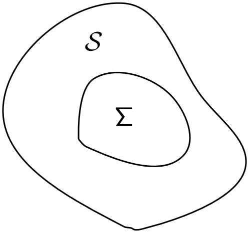
$$(P')_{ss'} := \begin{cases} (P)_{ss'} & s \in \Sigma \\ \dfrac{(P)_{ss'}}{1-(P)_{ss}} & s \in \overline{\Sigma}, s' \neq s \\ 0 & s \in \overline{\Sigma}, s' = s \end{cases}$$

- P' is in detailed balance with

$$\pi'_s = \begin{cases} \pi_s & s \in \Sigma \\ (1-(P)_{ss})\,\pi_s & s \in \overline{\Sigma} \end{cases}$$

Rejection-free algorithm

- We don't care about states in $\overline{\Sigma} := \mathcal{S} \setminus \Sigma$
- Therefore we can safely avoid rejections when $(A, u, v) \in \overline{\Sigma}$



- Use rejection-free chain instead

$$(P')_{ss'} := \begin{cases} (P)_{ss'} & s \in \Sigma \\ \dfrac{(P)_{ss'}}{1-(P)_{ss}} & s \in \overline{\Sigma}, s' \neq s \\ 0 & s \in \overline{\Sigma}, s' = s \end{cases}$$

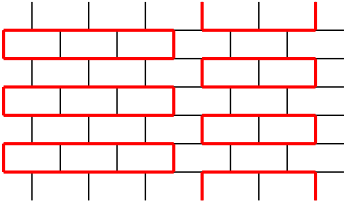
- P' is in detailed balance with

$$\pi'_s = \begin{cases} \pi_s & s \in \Sigma \\ (1-(P)_{ss})\,\pi_s & s \in \overline{\Sigma} \end{cases}$$

- $\overline{P'}$ is in detailed balance with $\overline{\pi'}_s = \overline{\pi}_s = \frac{\pi_s}{\sum_{s' \in \Sigma} \pi_{s'}}$, $s \in \Sigma$

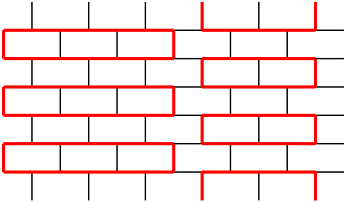
Fully-packed loops

Let G be bicubic and take $x \to \infty$



Fully-packed loops

Let G be bicubic and take $x \to \infty$



- State space is

$$\mathcal{F}(G) := \{A \in \mathcal{C}(G) : |A| = |V|\}$$

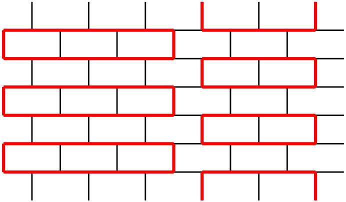
- Assign probabilities via

$$\phi_{G,n}(A) \propto n^{c(A)}, \qquad A \in \mathcal{F}(G)$$

- $c(A)$ is cyclomatic number

Fully-packed loops

Let G be bicubic and take $x \to \infty$



- State space is

$$\mathcal{F}(G) := \{A \in \mathcal{C}(G) : |A| = |V|\}$$

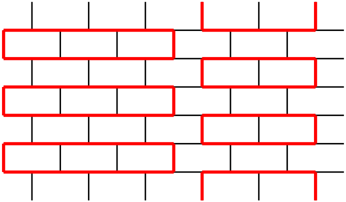
- Assign probabilities via

$$\phi_{G,n}(A) \propto n^{c(A)}, \qquad A \in \mathcal{F}(G)$$

- $c(A)$ is cyclomatic number
- $\mathcal{F}(G)$ is subset of cycle space with maximal $|A|$ $(= |V|)$

Fully-packed loops

Let G be bicubic and take $x \to \infty$



- State space is

$$\mathcal{F}(G) := \{A \in \mathcal{C}(G) : |A| = |V|\}$$

- Assign probabilities via

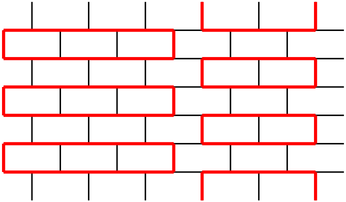
$$\phi_{G,n}(A) \propto n^{c(A)}, \qquad A \in \mathcal{F}(G)$$

- $c(A)$ is cyclomatic number

- $\mathcal{F}(G)$ is subset of cycle space with maximal $|A|$ $(= |V|)$
  - $A \in \mathcal{F}(G)$ iff (V, A) is a 2-factor iff $E \setminus A$ is a perfect matching

Fully-packed loops

Let G be bicubic and take $x \to \infty$



- State space is

$$\mathcal{F}(G) := \{A \in \mathcal{C}(G) : |A| = |V|\}$$

- Assign probabilities via

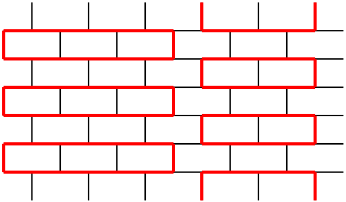
$$\phi_{G,n}(A) \propto n^{c(A)}, \qquad A \in \mathcal{F}(G)$$

- $c(A)$ is cyclomatic number

- $\mathcal{F}(G)$ is subset of cycle space with maximal $|A|$ $(= |V|)$
  - $A \in \mathcal{F}(G)$ iff (V, A) is a 2-factor iff $E \setminus A$ is a perfect matching
  - $n = 0$ corresponds to uniformly sampling Hamiltonian cycles

Fully-packed loops

Let G be bicubic and take $x \to \infty$



- State space is

$$\mathcal{F}(G) := \{A \in \mathcal{C}(G) : |A| = |V|\}$$

- Assign probabilities via

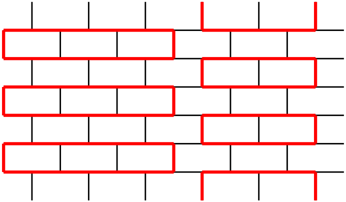
$$\phi_{G,n}(A) \propto n^{c(A)}, \qquad A \in \mathcal{F}(G)$$

- $c(A)$ is cyclomatic number

- $\mathcal{F}(G)$ is subset of cycle space with maximal $|A|$ $(= |V|)$
  - $A \in \mathcal{F}(G)$ iff (V, A) is a 2-factor iff $E \setminus A$ is a perfect matching
  - $n = 0$ corresponds to uniformly sampling Hamiltonian cycles
  - $n = 1$ corresponds to uniformly sampling dimer coverings

Fully-packed loops

Let G be bicubic and take $x \to \infty$



- State space is

$$\mathcal{F}(G) := \{A \in \mathcal{C}(G) : |A| = |V|\}$$

- Assign probabilities via

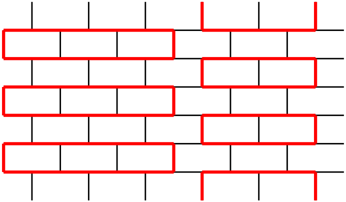
$$\phi_{G,n}(A) \propto n^{c(A)}, \qquad A \in \mathcal{F}(G)$$

- $c(A)$ is cyclomatic number

- $\mathcal{F}(G)$ is subset of cycle space with maximal $|A|$ $(= |V|)$
  - $A \in \mathcal{F}(G)$ iff (V, A) is a 2-factor iff $E \setminus A$ is a perfect matching
  - $n = 0$ corresponds to uniformly sampling Hamiltonian cycles
  - $n = 1$ corresponds to uniformly sampling dimer coverings
  - $n = 1$ corresponds to dual Ising model

Fully-packed loops

Let G be bicubic and take $x \to \infty$



- State space is

$$\mathcal{F}(G) := \{A \in \mathcal{C}(G) : |A| = |V|\}$$

- Assign probabilities via

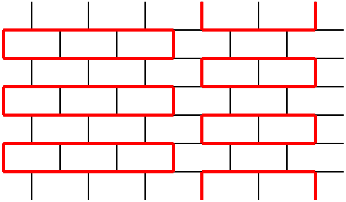
$$\phi_{G,n}(A) \propto n^{c(A)}, \qquad A \in \mathcal{F}(G)$$

- $c(A)$ is cyclomatic number

- $\mathcal{F}(G)$ is subset of cycle space with maximal $|A|$ ($= |V|$)
  - $A \in \mathcal{F}(G)$ iff (V, A) is a 2-factor iff $E \setminus A$ is a perfect matching
  - $n = 0$ corresponds to uniformly sampling Hamiltonian cycles
  - $n = 1$ corresponds to uniformly sampling dimer coverings
  - $n = 1$ corresponds to dual Ising model
  - $n = 2$ related to $q > 2$ Potts models (more later...)

Fully-packed loops

Let G be bicubic and take $x \to \infty$



- State space is

$$\mathcal{F}(G) := \{A \in \mathcal{C}(G) : |A| = |V|\}$$

- Assign probabilities via

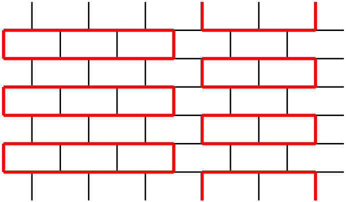
$$\phi_{G,n}(A) \propto n^{c(A)}, \qquad A \in \mathcal{F}(G)$$

- $c(A)$ is cyclomatic number

- $\mathcal{F}(G)$ is subset of cycle space with maximal $|A|$ $(= |V|)$
  - $A \in \mathcal{F}(G)$ iff (V, A) is a 2-factor iff $E \setminus A$ is a perfect matching
  - $n = 0$ corresponds to uniformly sampling Hamiltonian cycles
  - $n = 1$ corresponds to uniformly sampling dimer coverings
  - $n = 1$ corresponds to dual Ising model
  - $n = 2$ related to $q > 2$ Potts models (more later. . .)
- The simple worm algorithm is absorbing when $x = +\infty$

Fully-packed loops

Let G be bicubic and take $x \to \infty$



- State space is

$$\mathcal{F}(G) := \{A \in \mathcal{C}(G) : |A| = |V|\}$$

- Assign probabilities via

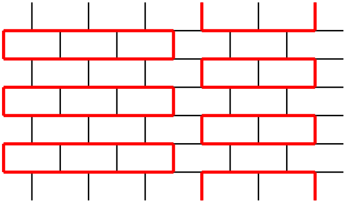
$$\phi_{G,n}(A) \propto n^{c(A)}, \qquad A \in \mathcal{F}(G)$$

- $c(A)$ is cyclomatic number

- $\mathcal{F}(G)$ is subset of cycle space with maximal $|A|$ $(= |V|)$
  - $A \in \mathcal{F}(G)$ iff (V, A) is a 2-factor iff $E \setminus A$ is a perfect matching
  - $n = 0$ corresponds to uniformly sampling Hamiltonian cycles
  - $n = 1$ corresponds to uniformly sampling dimer coverings
  - $n = 1$ corresponds to dual Ising model
  - $n = 2$ related to $q > 2$ Potts models (more later...)
- The simple worm algorithm is absorbing when $x = +\infty$
- Rejection-free algorithm remains valid

Fully-packed loops

Let G be bicubic and take $x \to \infty$



- State space is

$$\mathcal{F}(G) := \{A \in \mathcal{C}(G) : |A| = |V|\}$$

- Assign probabilities via

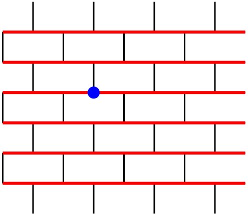
$$\phi_{G,n}(A) \propto n^{c(A)}, \qquad A \in \mathcal{F}(G)$$

- $c(A)$ is cyclomatic number

- $\mathcal{F}(G)$ is subset of cycle space with maximal $|A|$ $(= |V|)$
  - $A \in \mathcal{F}(G)$ iff (V, A) is a 2-factor iff $E \setminus A$ is a perfect matching
  - $n = 0$ corresponds to uniformly sampling Hamiltonian cycles
  - $n = 1$ corresponds to uniformly sampling dimer coverings
  - $n = 1$ corresponds to dual Ising model
  - $n = 2$ related to $q > 2$ Potts models (more later...)
- The simple worm algorithm is absorbing when $x = +\infty$
- Rejection-free algorithm remains valid
  - $n = 1$ similar to Jerrum & Sinclair's perfect matching algorithm

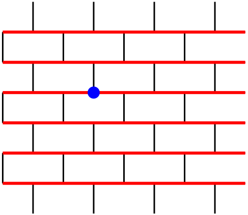
Worm algorithm for fully-packed loops

$P'_{n,\infty}$ allows the following transitions:



Worm algorithm for fully-packed loops

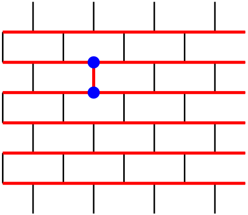
$P'_{n,\infty}$ allows the following transitions:



- If $d_A(u) = d_A(v) = 2$ then
  - Move one of the defects across $uu' \notin A$

Worm algorithm for fully-packed loops

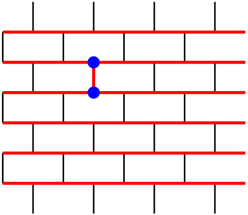
$P'_{n,\infty}$ allows the following transitions:



- If $d_A(u) = d_A(v) = 2$ then
  - Move one of the defects across $uu' \notin A$

Worm algorithm for fully-packed loops

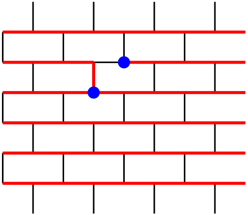
$P'_{n,\infty}$ allows the following transitions:



- If $d_A(u) = d_A(v) = 2$ then
  - Move one of the defects across $uu' \notin A$
- If $d_A(u) = d_A(v) = 3$ then
  - Delete one of the occupied edges at u or v

Worm algorithm for fully-packed loops

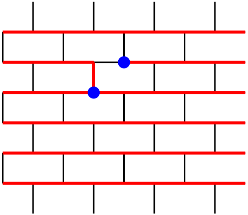
$P'_{n,\infty}$ allows the following transitions:



- If $d_A(u) = d_A(v) = 2$ then
  - Move one of the defects across $uu' \notin A$
- If $d_A(u) = d_A(v) = 3$ then
  - Delete one of the occupied edges at u or v

Worm algorithm for fully-packed loops

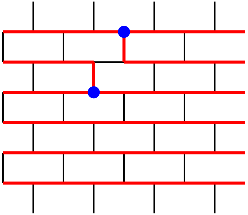
$P'_{n,\infty}$ allows the following transitions:



- If $d_A(u) = d_A(v) = 2$ then
  - Move one of the defects across $uu' \notin A$
- If $d_A(u) = d_A(v) = 3$ then
  - Delete one of the occupied edges at u or v
- If $d_A(u) = 1$ and $d_A(v) = 3$ then
  - Add one of the two vacant edges at u

Worm algorithm for fully-packed loops

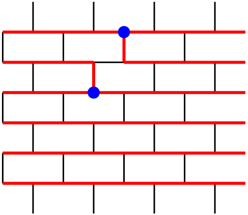
$P'_{n,\infty}$ allows the following transitions:



- If $d_A(u) = d_A(v) = 2$ then
  - Move one of the defects across $uu' \notin A$
- If $d_A(u) = d_A(v) = 3$ then
  - Delete one of the occupied edges at u or v
- If $d_A(u) = 1$ and $d_A(v) = 3$ then
  - Add one of the two vacant edges at u

Worm algorithm for fully-packed loops

$P'_{n,\infty}$ allows the following transitions:

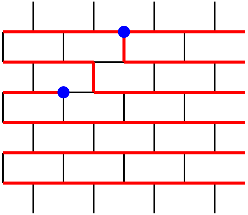


- If $d_A(u) = d_A(v) = 2$ then
  - Move one of the defects across $uu' \notin A$
- If $d_A(u) = d_A(v) = 3$ then
  - Delete one of the occupied edges at u or v
- If $d_A(u) = 1$ and $d_A(v) = 3$ then
  - Add one of the two vacant edges at u

- $\mathcal{R}(G) = \{(A, u, v) \in \mathcal{S}(G) \;:\; |A| \geq |V|\}$ is closed under $P'_{n,\infty}$

Worm algorithm for fully-packed loops

$P'_{n,\infty}$ allows the following transitions:

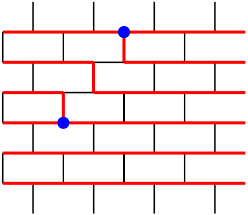


- If $d_A(u) = d_A(v) = 2$ then
  - Move one of the defects across $uu' \notin A$
- If $d_A(u) = d_A(v) = 3$ then
  - Delete one of the occupied edges at u or v
- If $d_A(u) = 1$ and $d_A(v) = 3$ then
  - Add one of the two vacant edges at u

- $\mathcal{R}(G) = \{(A, u, v) \in \mathcal{S}(G) \ : \ |A| \geq |V|\}$ is closed under $P'_{n,\infty}$

Worm algorithm for fully-packed loops

$P'_{n,\infty}$ allows the following transitions:

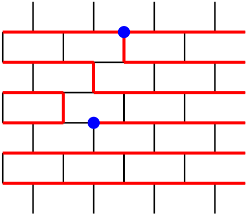


- If $d_A(u) = d_A(v) = 2$ then
  - Move one of the defects across $uu' \notin A$
- If $d_A(u) = d_A(v) = 3$ then
  - Delete one of the occupied edges at u or v
- If $d_A(u) = 1$ and $d_A(v) = 3$ then
  - Add one of the two vacant edges at u

- $\mathcal{R}(G) = \{(A, u, v) \in \mathcal{S}(G) \ : \ |A| \geq |V|\}$ is closed under $P'_{n,\infty}$

Worm algorithm for fully-packed loops

$P'_{n,\infty}$ allows the following transitions:

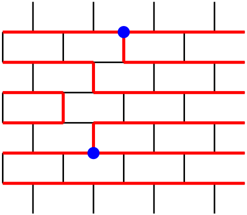


- If $d_A(u) = d_A(v) = 2$ then
  - Move one of the defects across $uu' \notin A$
- If $d_A(u) = d_A(v) = 3$ then
  - Delete one of the occupied edges at u or v
- If $d_A(u) = 1$ and $d_A(v) = 3$ then
  - Add one of the two vacant edges at u

- $\mathcal{R}(G) = \{(A, u, v) \in \mathcal{S}(G) \; : \; |A| \geq |V|\}$ is closed under $P'_{n,\infty}$

Worm algorithm for fully-packed loops

$P'_{n,\infty}$ allows the following transitions:

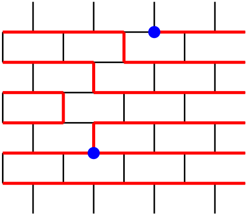


- If $d_A(u) = d_A(v) = 2$ then
  - Move one of the defects across $uu' \notin A$
- If $d_A(u) = d_A(v) = 3$ then
  - Delete one of the occupied edges at u or v
- If $d_A(u) = 1$ and $d_A(v) = 3$ then
  - Add one of the two vacant edges at u

- $\mathcal{R}(G) = \{(A, u, v) \in \mathcal{S}(G) \,:\, |A| \geq |V|\}$ is closed under $P'_{n,\infty}$

Worm algorithm for fully-packed loops

$P'_{n,\infty}$ allows the following transitions:

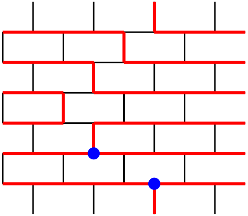


- If $d_A(u) = d_A(v) = 2$ then
  - Move one of the defects across $uu' \notin A$
- If $d_A(u) = d_A(v) = 3$ then
  - Delete one of the occupied edges at u or v
- If $d_A(u) = 1$ and $d_A(v) = 3$ then
  - Add one of the two vacant edges at u

- $\mathcal{R}(G) = \{(A, u, v) \in \mathcal{S}(G) \,:\, |A| \geq |V|\}$ is closed under $P'_{n,\infty}$

Worm algorithm for fully-packed loops

$P'_{n,\infty}$ allows the following transitions:

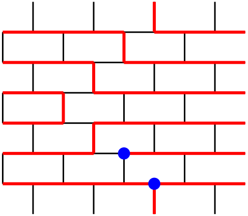


- If $d_A(u) = d_A(v) = 2$ then
  - Move one of the defects across $uu' \notin A$
- If $d_A(u) = d_A(v) = 3$ then
  - Delete one of the occupied edges at u or v
- If $d_A(u) = 1$ and $d_A(v) = 3$ then
  - Add one of the two vacant edges at u

- $\mathcal{R}(G) = \{(A, u, v) \in \mathcal{S}(G) \,:\, |A| \geq |V|\}$ is closed under $P'_{n,\infty}$

Worm algorithm for fully-packed loops

$P'_{n,\infty}$ allows the following transitions:

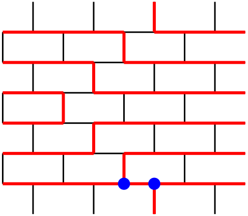


- If $d_A(u) = d_A(v) = 2$ then
  - Move one of the defects across $uu' \notin A$
- If $d_A(u) = d_A(v) = 3$ then
  - Delete one of the occupied edges at u or v
- If $d_A(u) = 1$ and $d_A(v) = 3$ then
  - Add one of the two vacant edges at u

- $\mathcal{R}(G) = \{(A, u, v) \in \mathcal{S}(G) \,:\, |A| \geq |V|\}$ is closed under $P'_{n,\infty}$

Worm algorithm for fully-packed loops

$P'_{n,\infty}$ allows the following transitions:

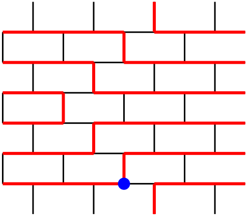


- If $d_A(u) = d_A(v) = 2$ then
  - Move one of the defects across $uu' \notin A$
- If $d_A(u) = d_A(v) = 3$ then
  - Delete one of the occupied edges at u or v
- If $d_A(u) = 1$ and $d_A(v) = 3$ then
  - Add one of the two vacant edges at u

- $\mathcal{R}(G) = \{(A, u, v) \in \mathcal{S}(G) \,:\, |A| \geq |V|\}$ is closed under $P'_{n,\infty}$

Worm algorithm for fully-packed loops

$P'_{n,\infty}$ allows the following transitions:

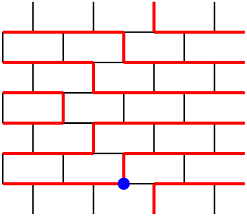


- If $d_A(u) = d_A(v) = 2$ then
  - Move one of the defects across $uu' \notin A$
- If $d_A(u) = d_A(v) = 3$ then
  - Delete one of the occupied edges at u or v
- If $d_A(u) = 1$ and $d_A(v) = 3$ then
  - Add one of the two vacant edges at u

- $\mathcal{R}(G) = \{(A, u, v) \in \mathcal{S}(G) : |A| \geq |V|\}$ is closed under $P'_{n,\infty}$

Worm algorithm for fully-packed loops

$P'_{n,\infty}$ allows the following transitions:

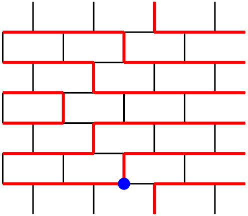


- If $d_A(u) = d_A(v) = 2$ then
  - Move one of the defects across $uu' \notin A$
- If $d_A(u) = d_A(v) = 3$ then
  - Delete one of the occupied edges at u or v
- If $d_A(u) = 1$ and $d_A(v) = 3$ then
  - Add one of the two vacant edges at u

- $\mathcal{R}(G) = \{(A, u, v) \in \mathcal{S}(G) \,:\, |A| \geq |V|\}$ is closed under $P'_{n,\infty}$
  - If $(A, v, v) \in \mathcal{R}(G)$ then $A \in \mathcal{F}(G)$

Worm algorithm for fully-packed loops

$P'_{n,\infty}$ allows the following transitions:

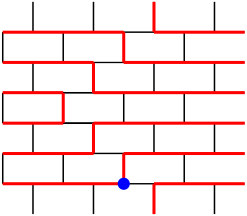


- If $d_A(u) = d_A(v) = 2$ then
  - Move one of the defects across $uu' \notin A$
- If $d_A(u) = d_A(v) = 3$ then
  - Delete one of the occupied edges at u or v
- If $d_A(u) = 1$ and $d_A(v) = 3$ then
  - Add one of the two vacant edges at u

- $\mathcal{R}(G) = \{(A, u, v) \in \mathcal{S}(G) \;:\; |A| \geq |V|\}$ is closed under $P'_{n,\infty}$
  - If $(A, v, v) \in \mathcal{R}(G)$ then $A \in \mathcal{F}(G)$
  - Let P_n be the restriction of $P'_{n,\infty}$ to $\mathcal{R}(G)$

Worm algorithm for fully-packed loops

$P'_{n,\infty}$ allows the following transitions:

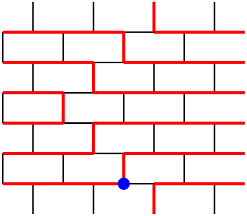


- If $d_A(u) = d_A(v) = 2$ then
  - Move one of the defects across $uu' \notin A$
- If $d_A(u) = d_A(v) = 3$ then
  - Delete one of the occupied edges at u or v
- If $d_A(u) = 1$ and $d_A(v) = 3$ then
  - Add one of the two vacant edges at u

- $\mathcal{R}(G) = \{(A, u, v) \in \mathcal{S}(G) \; : \; |A| \geq |V|\}$ is closed under $P'_{n,\infty}$
  - If $(A, v, v) \in \mathcal{R}(G)$ then $A \in \mathcal{F}(G)$
  - Let P_n be the restriction of $P'_{n,\infty}$ to $\mathcal{R}(G)$
- $\overline{P_n}$ is in detailed balance with $\phi_n(A) \propto n^{c(A)}$

Worm algorithm for fully-packed loops

$P'_{n,\infty}$ allows the following transitions:



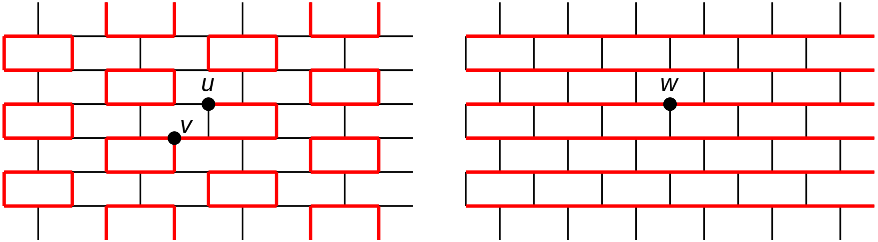
- If $d_A(u) = d_A(v) = 2$ then
  - Move one of the defects across $uu' \notin A$
- If $d_A(u) = d_A(v) = 3$ then
  - Delete one of the occupied edges at u or v
- If $d_A(u) = 1$ and $d_A(v) = 3$ then
  - Add one of the two vacant edges at u

- $\mathcal{R}(G) = \{(A, u, v) \in \mathcal{S}(G) \, : \, |A| \geq |V|\}$ is closed under $P'_{n,\infty}$
  - If $(A, v, v) \in \mathcal{R}(G)$ then $A \in \mathcal{F}(G)$
  - Let P_n be the restriction of $P'_{n,\infty}$ to $\mathcal{R}(G)$
- $\overline{P_n}$ is in detailed balance with $\phi_n(A) \propto n^{c(A)}$
- P_n is irreducible

P_n is irreducible

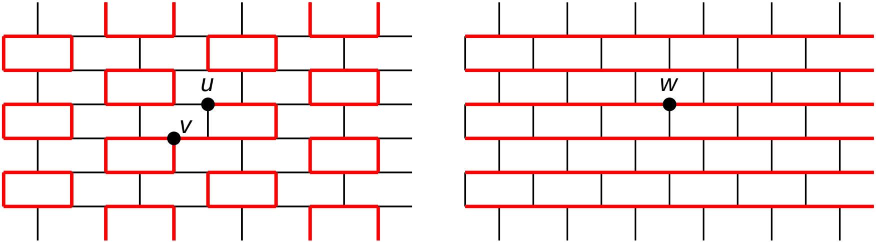
P_n is irreducible

We show $(A, u, v) \leftrightarrow (B, w, w)$ for any fixed $B \in \mathcal{F}(G)$ and $w \in V$



P_n is irreducible

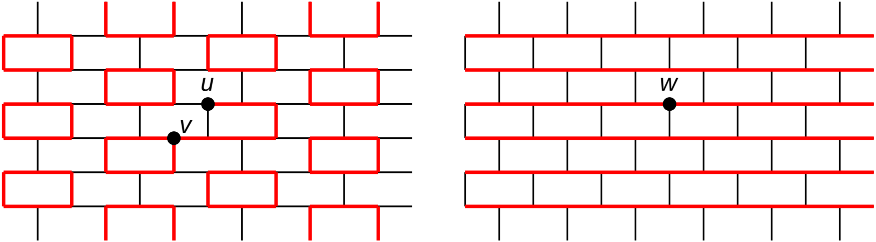
We show $(A, u, v) \leftrightarrow (B, w, w)$ for any fixed $B \in \mathcal{F}(G)$ and $w \in V$



- If $u \neq v$ then P_n always lets us to do the following:

P_n is irreducible

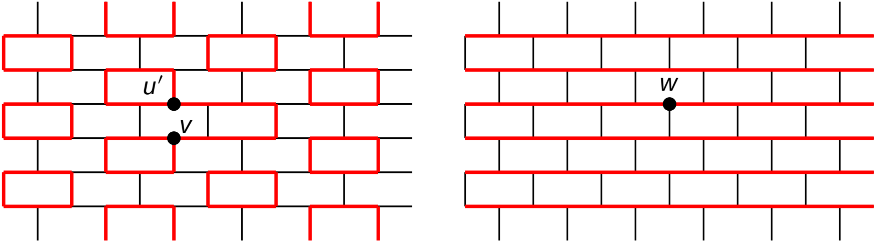
We show $(A, u, v) \leftrightarrow (B, w, w)$ for any fixed $B \in \mathcal{F}(G)$ and $w \in V$



- If $u \neq v$ then P_n always lets us to do the following:
  - If $d_u(A) = 1$ add one of the missing B-edges at u

P_n is irreducible

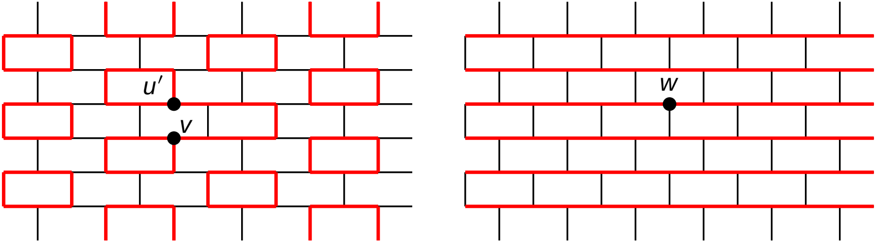
We show $(A, u, v) \leftrightarrow (B, w, w)$ for any fixed $B \in \mathcal{F}(G)$ and $w \in V$



- If $u \neq v$ then P_n always lets us to do the following:
  - If $d_u(A) = 1$ add one of the missing B-edges at u

P_n is irreducible

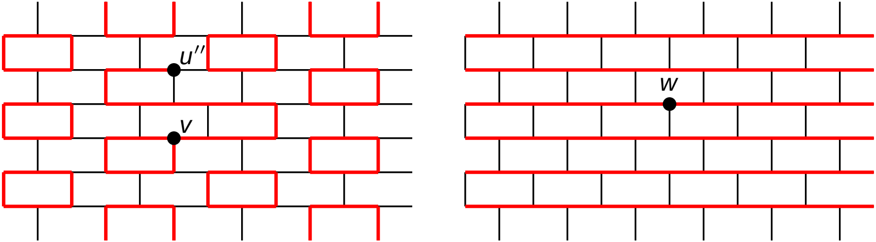
We show $(A, u, v) \leftrightarrow (B, w, w)$ for any fixed $B \in \mathcal{F}(G)$ and $w \in V$



- If $u \neq v$ then P_n always lets us to do the following:
  - If $d_u(A) = 1$ add one of the missing B-edges at u
  - If $d_{u'}(A) = d_v(A) = 3$ delete the occupied non-B-edge at u'

P_n is irreducible

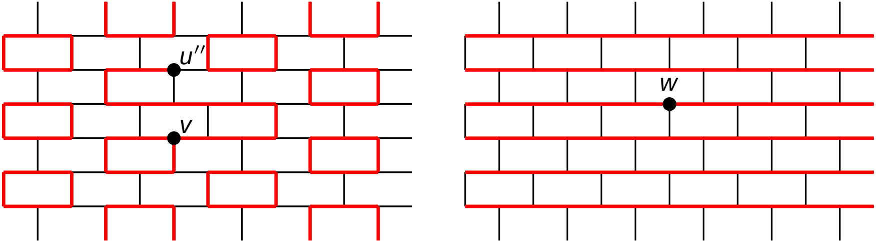
We show $(A, u, v) \leftrightarrow (B, w, w)$ for any fixed $B \in \mathcal{F}(G)$ and $w \in V$



- If $u \neq v$ then P_n always lets us to do the following:
  - If $d_u(A) = 1$ add one of the missing B-edges at u
  - If $d_{u'}(A) = d_v(A) = 3$ delete the occupied non-B-edge at u'

P_n is irreducible

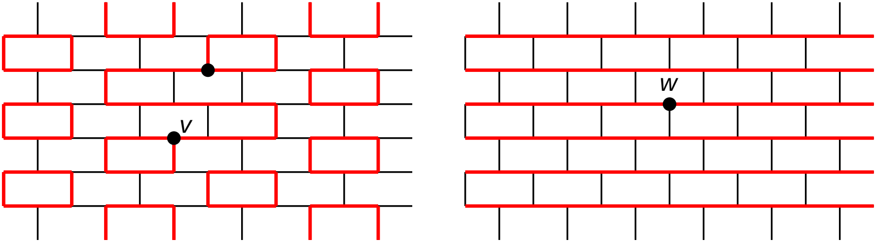
We show $(A, u, v) \leftrightarrow (B, w, w)$ for any fixed $B \in \mathcal{F}(G)$ and $w \in V$



- If $u \neq v$ then P_n always lets us to do the following:
  - If $d_u(A) = 1$ add one of the missing B-edges at u
  - If $d_{u'}(A) = d_v(A) = 3$ delete the occupied non-B-edge at u'
- New state has either one more B-edge or one less non-B-edge

P_n is irreducible

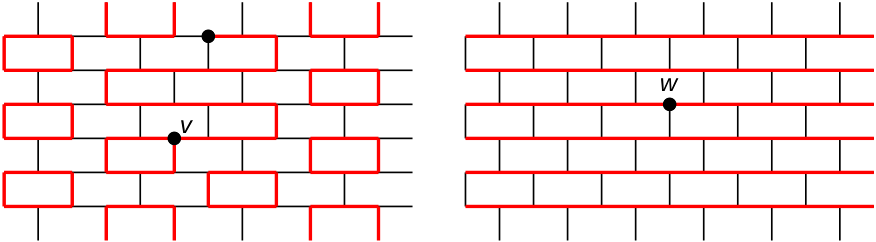
We show $(A, u, v) \leftrightarrow (B, w, w)$ for any fixed $B \in \mathcal{F}(G)$ and $w \in V$



- If $u \neq v$ then P_n always lets us to do the following:
  - If $d_u(A) = 1$ add one of the missing B-edges at u
  - If $d_{u'}(A) = d_v(A) = 3$ delete the occupied non-B-edge at u'
- New state has either one more B-edge or one less non-B-edge

P_n is irreducible

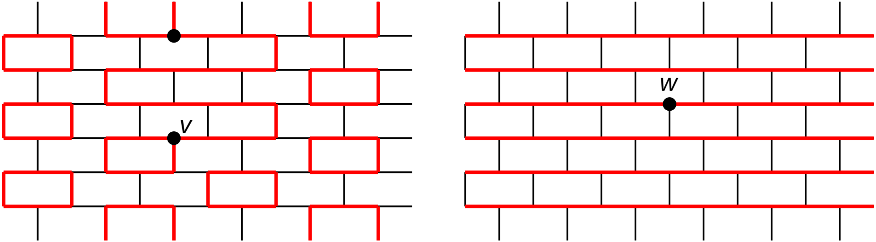
We show $(A, u, v) \leftrightarrow (B, w, w)$ for any fixed $B \in \mathcal{F}(G)$ and $w \in V$



- If $u \neq v$ then P_n always lets us to do the following:
  - If $d_u(A) = 1$ add one of the missing B-edges at u
  - If $d_{u'}(A) = d_v(A) = 3$ delete the occupied non-B-edge at u'
- New state has either one more B-edge or one less non-B-edge

P_n is irreducible

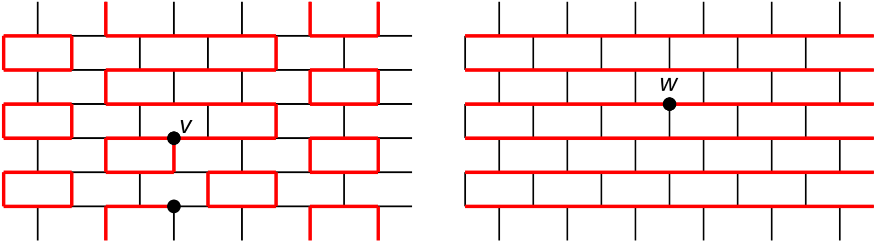
We show $(A, u, v) \leftrightarrow (B, w, w)$ for any fixed $B \in \mathcal{F}(G)$ and $w \in V$



- If $u \neq v$ then P_n always lets us to do the following:
  - If $d_u(A) = 1$ add one of the missing B-edges at u
  - If $d_{u'}(A) = d_v(A) = 3$ delete the occupied non-B-edge at u'
- New state has either one more B-edge or one less non-B-edge

P_n is irreducible

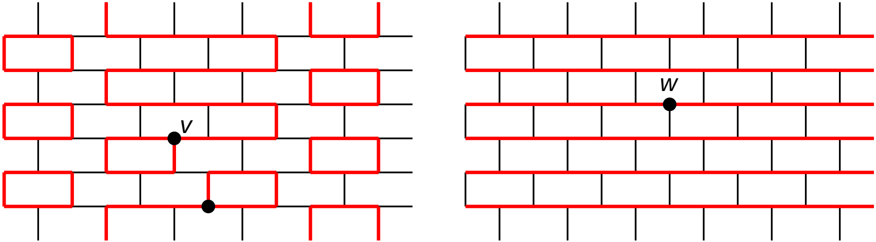
We show $(A, u, v) \leftrightarrow (B, w, w)$ for any fixed $B \in \mathcal{F}(G)$ and $w \in V$



- If $u \neq v$ then P_n always lets us to do the following:
  - If $d_u(A) = 1$ add one of the missing B-edges at u
  - If $d_{u'}(A) = d_v(A) = 3$ delete the occupied non-B-edge at u'
- New state has either one more B-edge or one less non-B-edge

P_n is irreducible

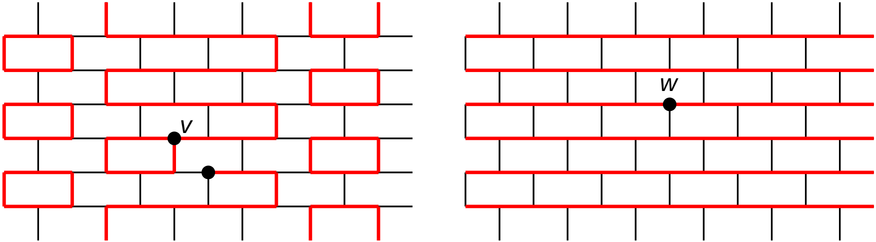
We show $(A, u, v) \leftrightarrow (B, w, w)$ for any fixed $B \in \mathcal{F}(G)$ and $w \in V$



- If $u \neq v$ then P_n always lets us to do the following:
  - If $d_u(A) = 1$ add one of the missing B-edges at u
  - If $d_{u'}(A) = d_v(A) = 3$ delete the occupied non-B-edge at u'
- New state has either one more B-edge or one less non-B-edge

P_n is irreducible

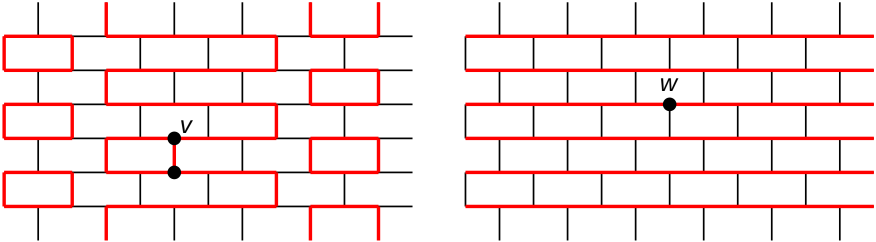
We show $(A, u, v) \leftrightarrow (B, w, w)$ for any fixed $B \in \mathcal{F}(G)$ and $w \in V$



- If $u \neq v$ then P_n always lets us to do the following:
  - If $d_u(A) = 1$ add one of the missing B-edges at u
  - If $d_{u'}(A) = d_v(A) = 3$ delete the occupied non-B-edge at u'
- New state has either one more B-edge or one less non-B-edge

P_n is irreducible

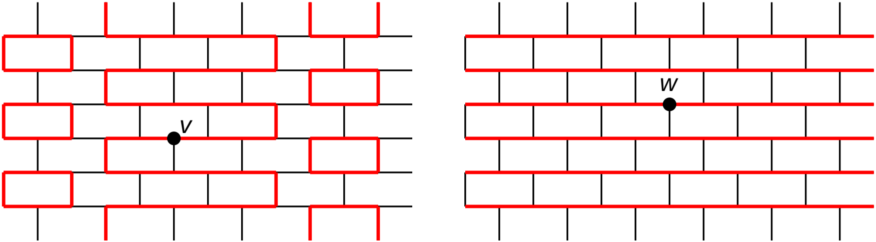
We show $(A, u, v) \leftrightarrow (B, w, w)$ for any fixed $B \in \mathcal{F}(G)$ and $w \in V$



- If $u \neq v$ then P_n always lets us to do the following:
  - If $d_u(A) = 1$ add one of the missing B-edges at u
  - If $d_{u'}(A) = d_v(A) = 3$ delete the occupied non-B-edge at u'
- New state has either one more B-edge or one less non-B-edge

P_n is irreducible

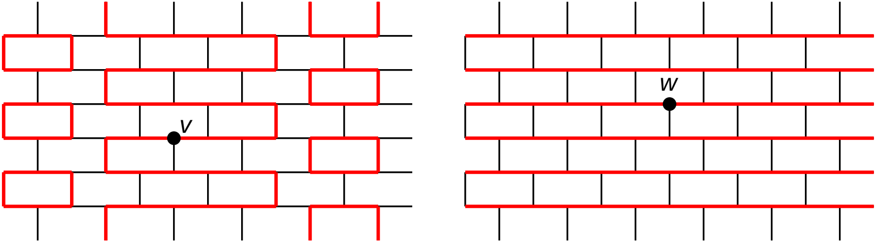
We show $(A, u, v) \leftrightarrow (B, w, w)$ for any fixed $B \in \mathcal{F}(G)$ and $w \in V$



- If $u \neq v$ then P_n always lets us to do the following:
  - If $d_u(A) = 1$ add one of the missing B-edges at u
  - If $d_{u'}(A) = d_v(A) = 3$ delete the occupied non-B-edge at u'
- New state has either one more B-edge or one less non-B-edge

P_n is irreducible

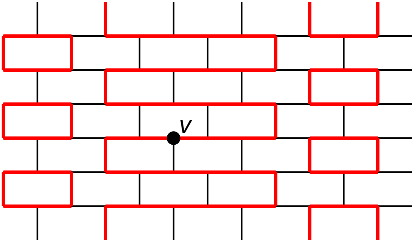
We show $(A, u, v) \leftrightarrow (B, w, w)$ for any fixed $B \in \mathcal{F}(G)$ and $w \in V$

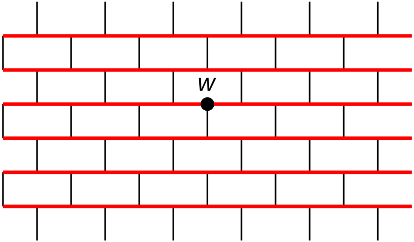


- If $u \neq v$ then P_n always lets us to do the following:
  - If $d_u(A) = 1$ add one of the missing B-edges at u
  - If $d_{u'}(A) = d_v(A) = 3$ delete the occupied non-B-edge at u'
- New state has either one more B-edge or one less non-B-edge
- We can prove that $(A, v, v) \leftrightarrow (A, v', v')$

P_n is irreducible

We show $(A, u, v) \leftrightarrow (B, w, w)$ for any fixed $B \in \mathcal{F}(G)$ and $w \in V$

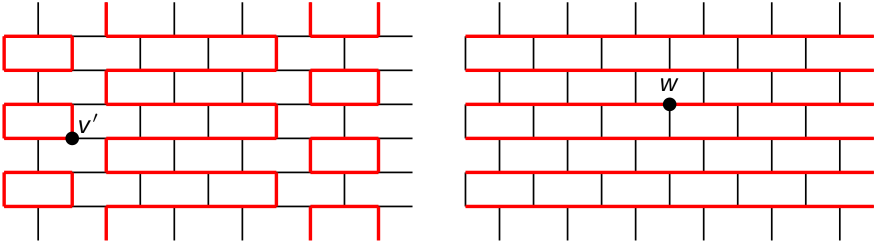




- If $u \neq v$ then P_n always lets us to do the following:
  - If $d_u(A) = 1$ add one of the missing B-edges at u
  - If $d_{u'}(A) = d_v(A) = 3$ delete the occupied non-B-edge at u'
- New state has either one more B-edge or one less non-B-edge
- We can prove that $(A, v, v) \leftrightarrow (A, v', v')$
- We can move defect to a v' adjacent to a vacant B-edge

P_n is irreducible

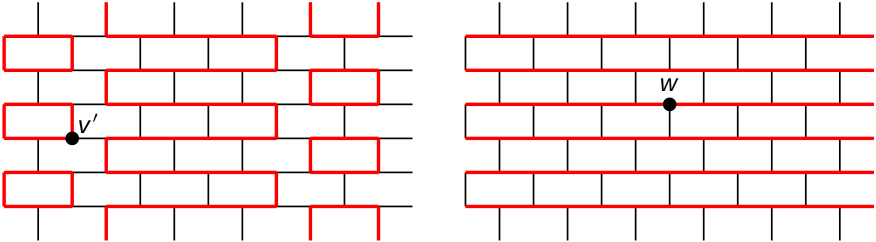
We show $(A, u, v) \leftrightarrow (B, w, w)$ for any fixed $B \in \mathcal{F}(G)$ and $w \in V$



- If $u \neq v$ then P_n always lets us to do the following:
  - If $d_u(A) = 1$ add one of the missing B-edges at u
  - If $d_{u'}(A) = d_v(A) = 3$ delete the occupied non-B-edge at u'
- New state has either one more B-edge or one less non-B-edge
- We can prove that $(A, v, v) \leftrightarrow (A, v', v')$
- We can move defect to a v' adjacent to a vacant B-edge

P_n is irreducible

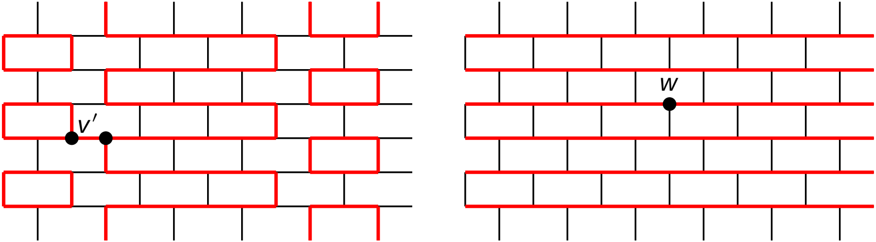
We show $(A, u, v) \leftrightarrow (B, w, w)$ for any fixed $B \in \mathcal{F}(G)$ and $w \in V$



- If $u \neq v$ then P_n always lets us to do the following:
  - If $d_u(A) = 1$ add one of the missing B-edges at u
  - If $d_{u'}(A) = d_v(A) = 3$ delete the occupied non-B-edge at u'
- New state has either one more B-edge or one less non-B-edge
- We can prove that $(A, v, v) \leftrightarrow (A, v', v')$
- We can move defect to a v' adjacent to a vacant B-edge
- Add vacant B-edge and start again

P_n is irreducible

We show $(A, u, v) \leftrightarrow (B, w, w)$ for any fixed $B \in \mathcal{F}(G)$ and $w \in V$

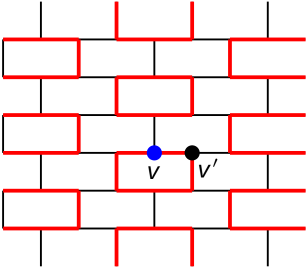


- If $u \neq v$ then P_n always lets us to do the following:
  - If $d_u(A) = 1$ add one of the missing B-edges at u
  - If $d_{u'}(A) = d_v(A) = 3$ delete the occupied non-B-edge at u'
- New state has either one more B-edge or one less non-B-edge
- We can prove that $(A, v, v) \leftrightarrow (A, v', v')$
- We can move defect to a v' adjacent to a vacant B-edge
- Add vacant B-edge and start again

Lemma: $(A, u, u) \leftrightarrow (A, v, v)$ for all $A \in \mathcal{F}(G)$ and $u, v \in V$

Lemma: $(A,u,u) \leftrightarrow (A,v,v)$ for all $A \in \mathcal{F}(G)$ and $u,v \in V$

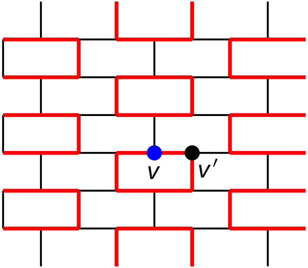
We show that $(A,v,v) \leftrightarrow (A,v',v')$ for each $v' \sim v$



Lemma: $(A, u, u) \leftrightarrow (A, v, v)$ for all $A \in \mathcal{F}(G)$ and $u, v \in V$

We show that $(A, v, v) \leftrightarrow (A, v', v')$ for each $v' \sim v$

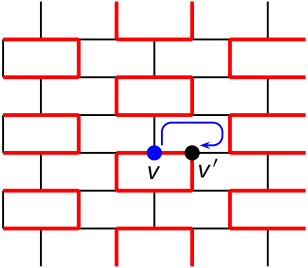
- Lemma: There always exists an odd path $P_{v\,v'}$ from v to $v' \sim v$ which alternates vacant, occupied,..., vacant



Lemma: $(A, u, u) \leftrightarrow (A, v, v)$ for all $A \in \mathcal{F}(G)$ and $u, v \in V$

We show that $(A, v, v) \leftrightarrow (A, v', v')$ for each $v' \sim v$

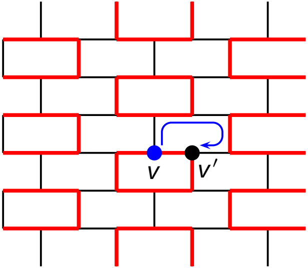
- Lemma: There always exists an odd path $P_{v\,v'}$ from v to $v' \sim v$ which alternates vacant, occupied, ..., vacant



Lemma: $(A, u, u) \leftrightarrow (A, v, v)$ for all $A \in \mathcal{F}(G)$ and $u, v \in V$

We show that $(A, v, v) \leftrightarrow (A, v', v')$ for each $v' \sim v$

- Lemma: There always exists an odd path $P_{v\,v'}$ from v to $v' \sim v$ which alternates vacant, occupied, . . . , vacant

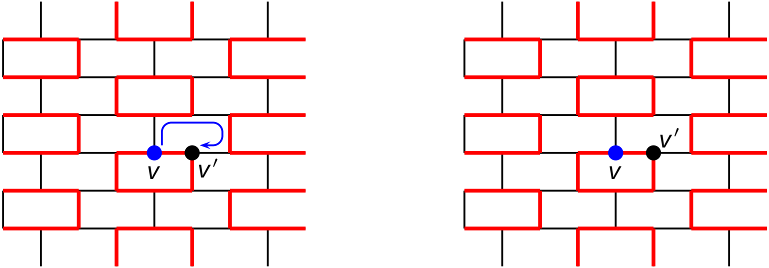


- Can move first defect along $P_{v\,v'}$ so $(A, v, v) \leftrightarrow (A', v', v)$

Lemma: $(A, u, u) \leftrightarrow (A, v, v)$ for all $A \in \mathcal{F}(G)$ and $u, v \in V$

We show that $(A, v, v) \leftrightarrow (A, v', v')$ for each $v' \sim v$

- Lemma: There always exists an odd path $P_{v\,v'}$ from v to $v' \sim v$ which alternates vacant, occupied, . . . , vacant

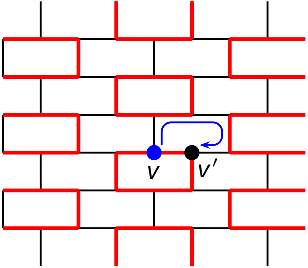


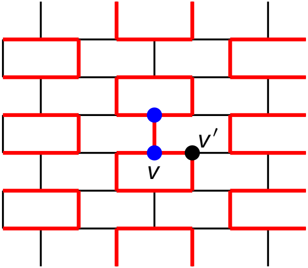
- Can move first defect along $P_{v\,v'}$ so $(A, v, v) \leftrightarrow (A', v', v)$

Lemma: $(A, u, u) \leftrightarrow (A, v, v)$ for all $A \in \mathcal{F}(G)$ and $u, v \in V$

We show that $(A, v, v) \leftrightarrow (A, v', v')$ for each $v' \sim v$

- Lemma: There always exists an odd path $P_{v\,v'}$ from v to $v' \sim v$ which alternates vacant, occupied, . . . , vacant



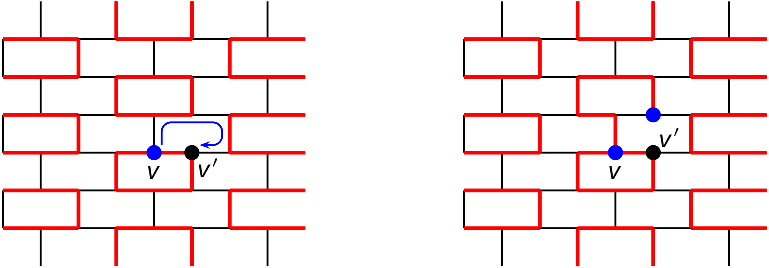


- Can move first defect along $P_{v\,v'}$ so $(A, v, v) \leftrightarrow (A', v', v)$

Lemma: $(A,u,u) \leftrightarrow (A,v,v)$ for all $A \in \mathcal{F}(G)$ and $u,v \in V$

We show that $(A,v,v) \leftrightarrow (A,v',v')$ for each $v' \sim v$

- Lemma: There always exists an odd path $P_{v\,v'}$ from v to $v' \sim v$ which alternates vacant, occupied,..., vacant

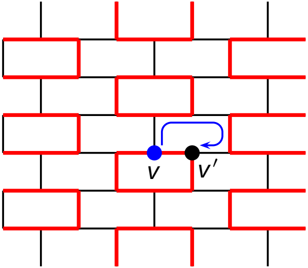


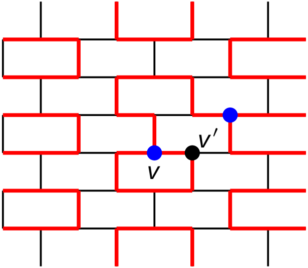
- Can move first defect along $P_{v\,v'}$ so $(A,v,v) \leftrightarrow (A',v',v)$

Lemma: $(A,u,u) \leftrightarrow (A,v,v)$ for all $A \in \mathcal{F}(G)$ and $u,v \in V$

We show that $(A,v,v) \leftrightarrow (A,v',v')$ for each $v' \sim v$

- Lemma: There always exists an odd path $P_{v\,v'}$ from v to $v' \sim v$ which alternates vacant, occupied, . . . , vacant



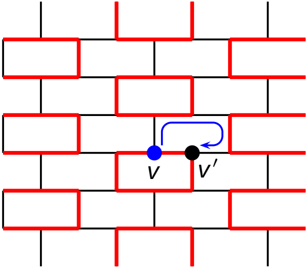


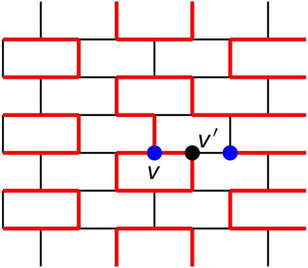
- Can move first defect along $P_{v\,v'}$ so $(A,v,v) \leftrightarrow (A',v',v)$

Lemma: $(A, u, u) \leftrightarrow (A, v, v)$ for all $A \in \mathcal{F}(G)$ and $u, v \in V$

We show that $(A, v, v) \leftrightarrow (A, v', v')$ for each $v' \sim v$

- Lemma: There always exists an odd path $P_{v\,v'}$ from v to $v' \sim v$ which alternates vacant, occupied, . . . , vacant



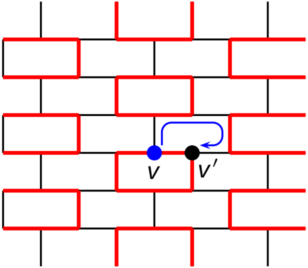


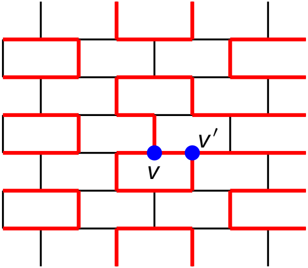
- Can move first defect along $P_{v\,v'}$ so $(A, v, v) \leftrightarrow (A', v', v)$

Lemma: $(A,u,u) \leftrightarrow (A,v,v)$ for all $A \in \mathcal{F}(G)$ and $u,v \in V$

We show that $(A,v,v) \leftrightarrow (A,v',v')$ for each $v' \sim v$

- Lemma: There always exists an odd path $P_{v\,v'}$ from v to $v' \sim v$ which alternates vacant, occupied, . . . , vacant



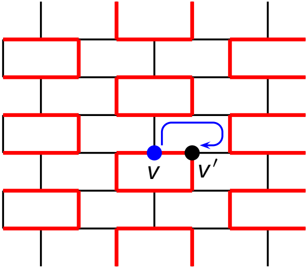


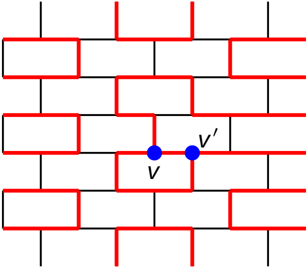
- Can move first defect along $P_{v\,v'}$ so $(A,v,v) \leftrightarrow (A',v',v)$

Lemma: $(A,u,u) \leftrightarrow (A,v,v)$ for all $A \in \mathcal{F}(G)$ and $u,v \in V$

We show that $(A,v,v) \leftrightarrow (A,v',v')$ for each $v' \sim v$

- Lemma: There always exists an odd path $P_{v\,v'}$ from v to $v' \sim v$ which alternates vacant, occupied,..., vacant



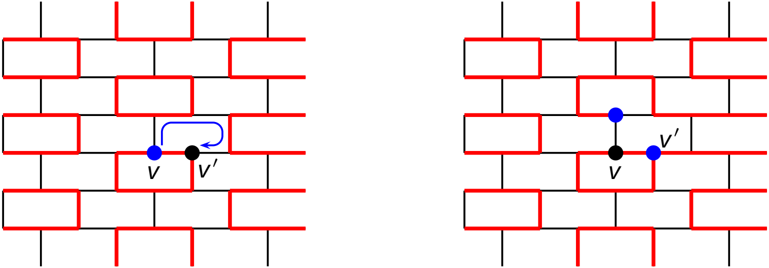


- Can move first defect along $P_{v\,v'}$ so $(A,v,v) \leftrightarrow (A',v',v)$
- Then move second defect along $P_{v\,v'}$ so $(A,v,v) \leftrightarrow (A'',v',v')$

Lemma: $(A, u, u) \leftrightarrow (A, v, v)$ for all $A \in \mathcal{F}(G)$ and $u, v \in V$

We show that $(A, v, v) \leftrightarrow (A, v', v')$ for each $v' \sim v$

- Lemma: There always exists an odd path $P_{v\,v'}$ from v to $v' \sim v$ which alternates vacant, occupied, . . . , vacant

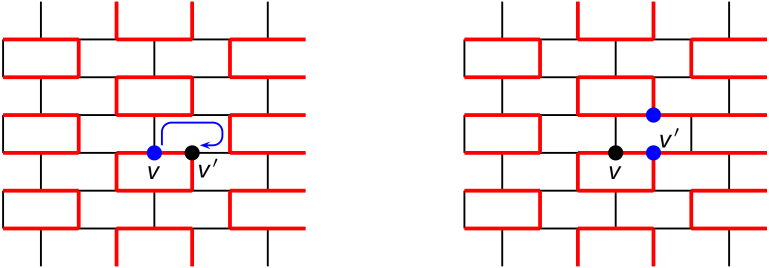


- Can move first defect along $P_{v\,v'}$ so $(A, v, v) \leftrightarrow (A', v', v)$
- Then move second defect along $P_{v\,v'}$ so $(A, v, v) \leftrightarrow (A'', v', v')$

Lemma: $(A, u, u) \leftrightarrow (A, v, v)$ for all $A \in \mathcal{F}(G)$ and $u, v \in V$

We show that $(A, v, v) \leftrightarrow (A, v', v')$ for each $v' \sim v$

- Lemma: There always exists an odd path $P_{v\,v'}$ from v to $v' \sim v$ which alternates vacant, occupied, …, vacant

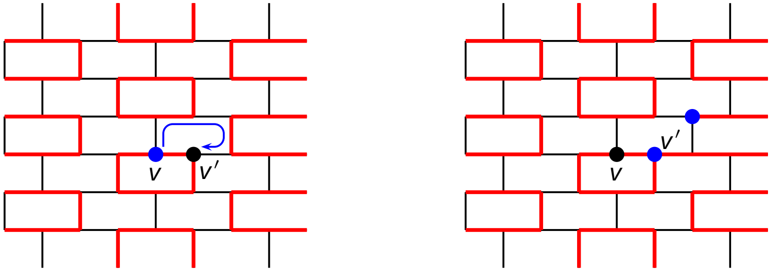


- Can move first defect along $P_{v\,v'}$ so $(A, v, v) \leftrightarrow (A', v', v)$
- Then move second defect along $P_{v\,v'}$ so $(A, v, v) \leftrightarrow (A'', v', v')$

Lemma: $(A, u, u) \leftrightarrow (A, v, v)$ for all $A \in \mathcal{F}(G)$ and $u, v \in V$

We show that $(A, v, v) \leftrightarrow (A, v', v')$ for each $v' \sim v$

- Lemma: There always exists an odd path $P_{v\,v'}$ from v to $v' \sim v$ which alternates vacant, occupied, . . . , vacant

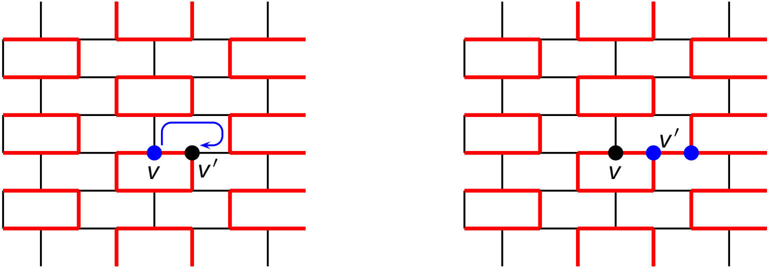


- Can move first defect along $P_{v\,v'}$ so $(A, v, v) \leftrightarrow (A', v', v)$
- Then move second defect along $P_{v\,v'}$ so $(A, v, v) \leftrightarrow (A'', v', v')$

Lemma: $(A,u,u) \leftrightarrow (A,v,v)$ for all $A \in \mathcal{F}(G)$ and $u,v \in V$

We show that $(A,v,v) \leftrightarrow (A,v',v')$ for each $v' \sim v$

- Lemma: There always exists an odd path $P_{v\,v'}$ from v to $v' \sim v$ which alternates vacant, occupied, . . . , vacant

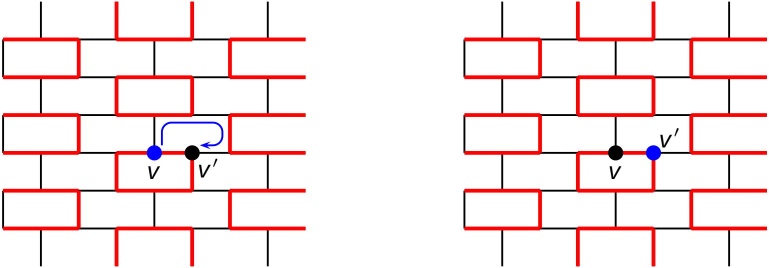


- Can move first defect along $P_{v\,v'}$ so $(A,v,v) \leftrightarrow (A',v',v)$
- Then move second defect along $P_{v\,v'}$ so $(A,v,v) \leftrightarrow (A'',v',v')$

Lemma: $(A, u, u) \leftrightarrow (A, v, v)$ for all $A \in \mathcal{F}(G)$ and $u, v \in V$

We show that $(A, v, v) \leftrightarrow (A, v', v')$ for each $v' \sim v$

- Lemma: There always exists an odd path $P_{v\,v'}$ from v to $v' \sim v$ which alternates vacant, occupied,. . . , vacant

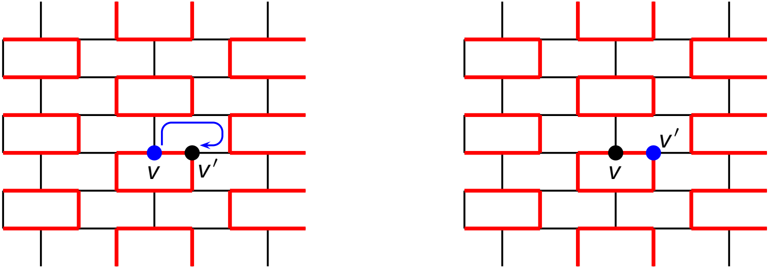


- Can move first defect along $P_{v\,v'}$ so $(A, v, v) \leftrightarrow (A', v', v)$
- Then move second defect along $P_{v\,v'}$ so $(A, v, v) \leftrightarrow (A'', v', v')$

Lemma: $(A, u, u) \leftrightarrow (A, v, v)$ for all $A \in \mathcal{F}(G)$ and $u, v \in V$

We show that $(A, v, v) \leftrightarrow (A, v', v')$ for each $v' \sim v$

- Lemma: There always exists an odd path $P_{v\,v'}$ from v to $v' \sim v$ which alternates vacant, occupied,. . . , vacant



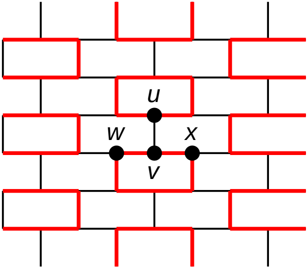
- Can move first defect along $P_{v\,v'}$ so $(A, v, v) \leftrightarrow (A', v', v)$
- Then move second defect along $P_{v\,v'}$ so $(A, v, v) \leftrightarrow (A'', v', v')$
- But each edge is traversed (flipped) exactly twice so $A'' = A$

Colouring Lemma

Lemma

Let G be a finite bicubic graph. For every $A \in \mathcal{F}(G)$ and every $v \in V$, there is a path to each neighbor such that the edges alternate vacant, occupied, . . . , vacant

Proof.



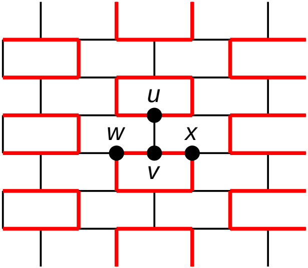
□

Colouring Lemma

Lemma

Let G be a finite bicubic graph. For every $A \in \mathcal{F}(G)$ and every $v \in V$, there is a path to each neighbor such that the edges alternate vacant, occupied, . . . , vacant

Proof.



- Independently colour each cycle, alternating red, blue

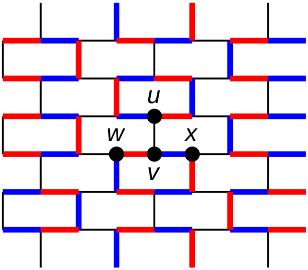
□

Colouring Lemma

Lemma

Let G be a finite bicubic graph. For every $A \in \mathcal{F}(G)$ and every $v \in V$, there is a path to each neighbor such that the edges alternate vacant, occupied, ..., vacant

Proof.



- Independently colour each cycle, alternating red, blue

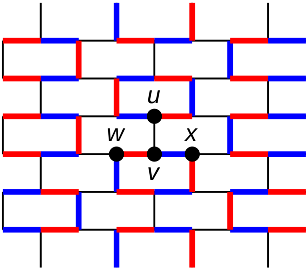
□

Colouring Lemma

Lemma

Let G be a finite bicubic graph. For every $A \in \mathcal{F}(G)$ and every $v \in V$, there is a path to each neighbor such that the edges alternate vacant, occupied, . . . , vacant

Proof.



- Independently colour each cycle, alternating red, blue
- Colour vacant edges green

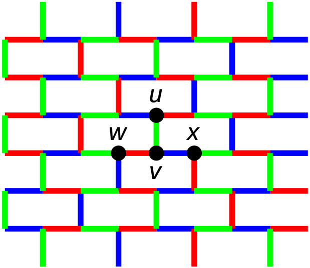
□

Colouring Lemma

Lemma

Let G be a finite bicubic graph. For every $A \in \mathcal{F}(G)$ and every $v \in V$, there is a path to each neighbor such that the edges alternate vacant, occupied, . . . , vacant

Proof.



- Independently colour each cycle, alternating red, blue
- Colour vacant edges green

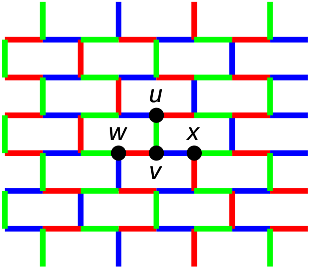
□

Colouring Lemma

Lemma

Let G be a finite bicubic graph. For every $A \in \mathcal{F}(G)$ and every $v \in V$, there is a path to each neighbor such that the edges alternate vacant, occupied, . . . , vacant

Proof.



- Independently colour each cycle, alternating red, blue
- Colour vacant edges green
- $A_{\text{red}} \cup A_{\text{green}} \in \mathcal{F}(G)$

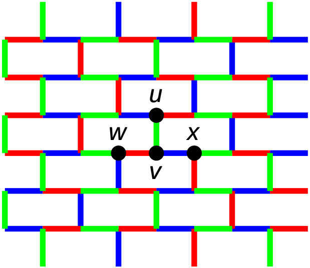
□

Colouring Lemma

Lemma

Let G be a finite bicubic graph. For every $A \in \mathcal{F}(G)$ and every $v \in V$, there is a path to each neighbor such that the edges alternate vacant, occupied, . . . , vacant

Proof.



- Independently colour each cycle, alternating red, blue
- Colour vacant edges green
- $A_{\mathrm{red}} \cup A_{\mathrm{green}} \in \mathcal{F}(G)$
- u, v, w all connected in $A_{\mathrm{red}} \cup A_{\mathrm{green}}$

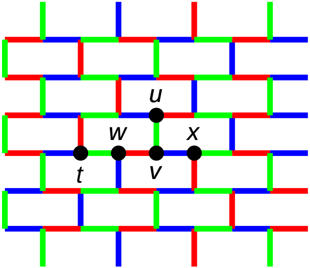
□

Colouring Lemma

Lemma

Let G be a finite bicubic graph. For every $A \in \mathcal{F}(G)$ and every $v \in V$, there is a path to each neighbor such that the edges alternate vacant, occupied, . . . , vacant

Proof.



- Independently colour each cycle, alternating red, blue
- Colour vacant edges green
- $A_{\text{red}} \cup A_{\text{green}} \in \mathcal{F}(G)$
- u, v, w all connected in $A_{\text{red}} \cup A_{\text{green}}$
- So u, v, w all in same cycle
  $v\,u\,\ldots t\,w\,v \in A_{\text{red}} \cup A_{\text{green}}$

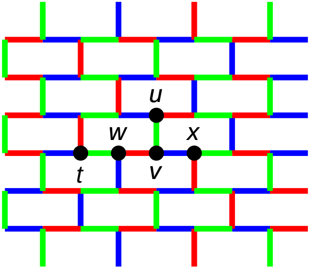
□

Colouring Lemma

Lemma

Let G be a finite bicubic graph. For every $A \in \mathcal{F}(G)$ and every $v \in V$, there is a path to each neighbor such that the edges alternate vacant, occupied, ..., vacant

Proof.



- Independently colour each cycle, alternating red, blue
- Colour vacant edges green
- $A_{\text{red}} \cup A_{\text{green}} \in \mathcal{F}(G)$
- u, v, w all connected in $A_{\text{red}} \cup A_{\text{green}}$
- So u, v, w all in same cycle $v\,u\,\ldots t\,w\,v \in A_{\text{red}} \cup A_{\text{green}}$
- Cycle must be even
  - alternates green, red, ..., green, red

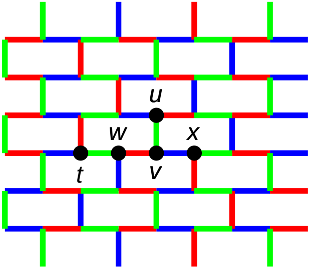
□

Colouring Lemma

Lemma

Let G be a finite bicubic graph. For every $A \in \mathcal{F}(G)$ and every $v \in V$, there is a path to each neighbor such that the edges alternate vacant, occupied, ..., vacant

Proof.



- Independently colour each cycle, alternating red, blue
- Colour vacant edges green
- $A_{\text{red}} \cup A_{\text{green}} \in \mathcal{F}(G)$
- u, v, w all connected in $A_{\text{red}} \cup A_{\text{green}}$
- So u, v, w all in same cycle $v\,u\,\ldots t\,w\,v \in A_{\text{red}} \cup A_{\text{green}}$
- Cycle must be even
  - alternates green, red, ..., green, red
- $v\,u\,\ldots t\,w$ is the desired path

□

Potts models and Wang-Swendsen-Kotecký algorithm

- q-state Potts model on graph $G = (V, E)$
  - Spin configurations $\sigma \in \{1, 2, \ldots, q\}^V$ with $q \in \{2, 3, \ldots\}$
  - $H(\sigma) = -\beta \sum_{uv \in E} \delta(\sigma_u, \sigma_v)$
  - $\mathbb{P}(\sigma) \propto e^{-H(\sigma)}$
  - $|\beta| = 1/T$

Potts models and Wang-Swendsen-Kotecký algorithm

- q-state Potts model on graph $G = (V, E)$
  - Spin configurations $\sigma \in \{1, 2, \ldots, q\}^V$ with $q \in \{2, 3, \ldots\}$
  - $H(\sigma) = -\beta \sum_{uv \in E} \delta(\sigma_u, \sigma_v)$
  - $\mathbb{P}(\sigma) \propto e^{-H(\sigma)}$
  - $|\beta| = 1/T$
- $\beta > 0$ is ferromagnetic, $\beta < 0$ is antiferromagnetic

Potts models and Wang-Swendsen-Kotecký algorithm

- q-state Potts model on graph $G = (V, E)$
  - Spin configurations $\sigma \in \{1, 2, \ldots, q\}^V$ with $q \in \{2, 3, \ldots\}$
  - $H(\sigma) = -\beta \sum_{uv \in E} \delta(\sigma_u, \sigma_v)$
  - $\mathbb{P}(\sigma) \propto e^{-H(\sigma)}$
  - $|\beta| = 1/T$
- $\beta > 0$ is ferromagnetic, $\beta < 0$ is antiferromagnetic
- When $\beta = -\infty$ we uniformly sample q-vertex colourings

Potts models and Wang-Swendsen-Kotecký algorithm

- q-state Potts model on graph $G = (V, E)$
  - Spin configurations $\sigma \in \{1, 2, \dots, q\}^V$ with $q \in \{2, 3, \dots\}$
  - $H(\sigma) = -\beta \sum_{uv \in E} \delta(\sigma_u, \sigma_v)$
  - $\mathbb{P}(\sigma) \propto e^{-H(\sigma)}$
  - $|\beta| = 1/T$
- $\beta > 0$ is ferromagnetic, $\beta < 0$ is antiferromagnetic
- When $\beta = -\infty$ we uniformly sample q-vertex colourings
- The Swendsen-Wang (SW) cluster algorithm (ferromagnetic):
  - Introduce auxiliary edge variables $\omega \in \{0, 1\}^E$
  - Consider joint (Edwards-Sokal) model $\mathbb{P}(\sigma, \omega)$
  - Update spins using $\mathbb{P}(\sigma|\omega)$ then update bonds using $\mathbb{P}(\omega|\sigma)$

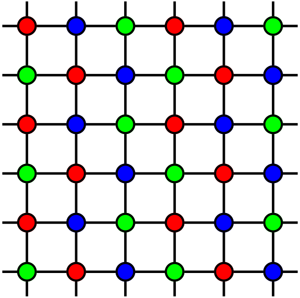
Potts models and Wang-Swendsen-Kotecký algorithm

- q-state Potts model on graph $G = (V, E)$
  - Spin configurations $\sigma \in \{1, 2, \ldots, q\}^V$ with $q \in \{2, 3, \ldots\}$
  - $H(\sigma) = -\beta \sum_{uv \in E} \delta(\sigma_u, \sigma_v)$
  - $\mathbb{P}(\sigma) \propto e^{-H(\sigma)}$
  - $|\beta| = 1/T$
- $\beta > 0$ is ferromagnetic, $\beta < 0$ is antiferromagnetic
- When $\beta = -\infty$ we uniformly sample q-vertex colourings
- The Swendsen-Wang (SW) cluster algorithm (ferromagnetic):
  - Introduce auxiliary edge variables $\omega \in \{0, 1\}^E$
  - Consider joint (Edwards-Sokal) model $\mathbb{P}(\sigma, \omega)$
  - Update spins using $\mathbb{P}(\sigma|\omega)$ then update bonds using $\mathbb{P}(\omega|\sigma)$
- Wang-Swendsen-Kotecký (WSK)
  - Extension of SW to treat antiferromagnetic Potts

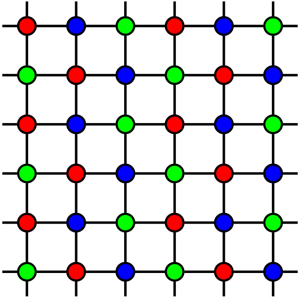
Potts models and Wang-Swendsen-Kotecký algorithm

- q-state Potts model on graph $G = (V, E)$
  - Spin configurations $\sigma \in \{1, 2, \ldots, q\}^V$ with $q \in \{2, 3, \ldots\}$
  - $H(\sigma) = -\beta \sum_{uv \in E} \delta(\sigma_u, \sigma_v)$
  - $\mathbb{P}(\sigma) \propto e^{-H(\sigma)}$
  - $|\beta| = 1/T$
- $\beta > 0$ is ferromagnetic, $\beta < 0$ is antiferromagnetic
- When $\beta = -\infty$ we uniformly sample q-vertex colourings
- The Swendsen-Wang (SW) cluster algorithm (ferromagnetic):
  - Introduce auxiliary edge variables $\omega \in \{0, 1\}^E$
  - Consider joint (Edwards-Sokal) model $\mathbb{P}(\sigma, \omega)$
  - Update spins using $\mathbb{P}(\sigma|\omega)$ then update bonds using $\mathbb{P}(\omega|\sigma)$
- Wang-Swendsen-Kotecký (WSK)
  - Extension of SW to treat antiferromagnetic Potts
- WSK at $T = 0$ equivalent to “Kempe changes”
  - Each cluster is a Kempe chain

WSK algorithm

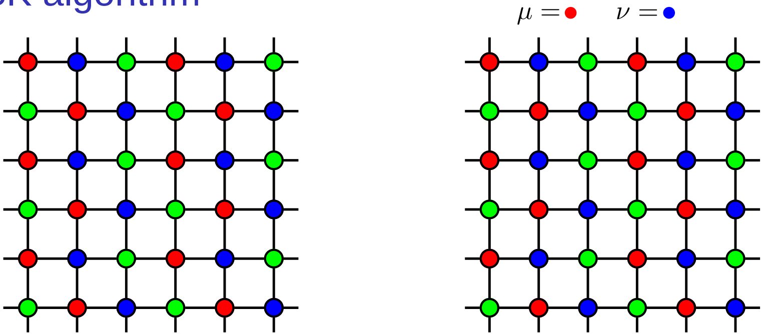


WSK algorithm



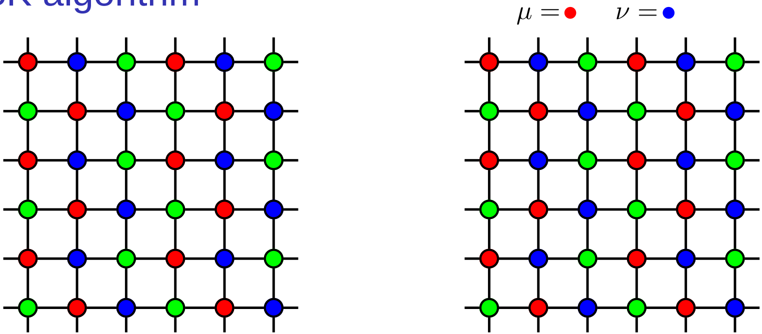
- Uniformly at random, choose two of the q possible colors $\mu, \nu \in \{1, 2 \ldots q\}$

WSK algorithm



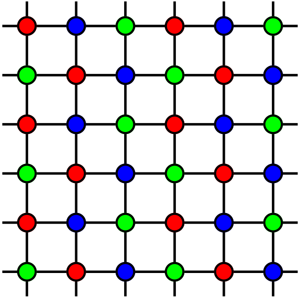
- Uniformly at random, choose two of the q possible colors $\mu, \nu \in \{1, 2 \ldots q\}$

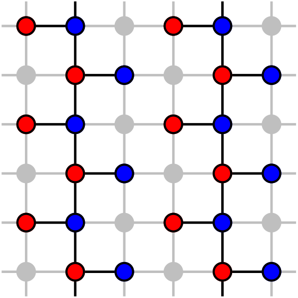
WSK algorithm



- Uniformly at random, choose two of the q possible colors $\mu, \nu \in \{1, 2 \ldots q\}$
- Freeze all the $\sigma_v \notin \{\mu, \nu\}$

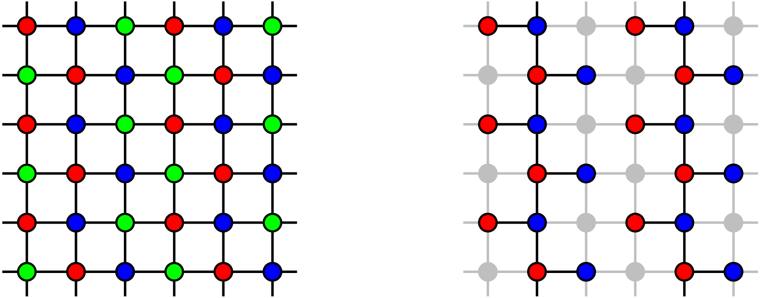
WSK algorithm





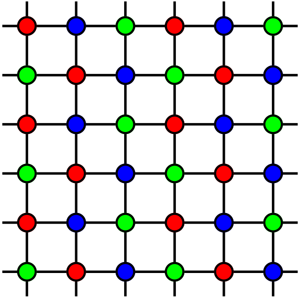
- Uniformly at random, choose two of the q possible colors $\mu, \nu \in \{1, 2 \ldots q\}$
- Freeze all the $\sigma_v \notin \{\mu, \nu\}$

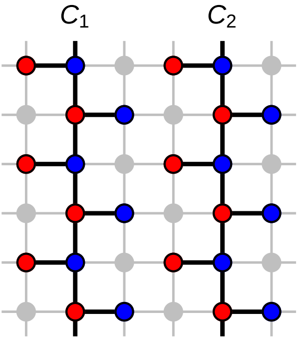
WSK algorithm



- Uniformly at random, choose two of the q possible colors $\mu, \nu \in \{1, 2 \ldots q\}$
- Freeze all the $\sigma_v \notin \{\mu, \nu\}$
- For each edge uv with with $\sigma_u \neq \sigma_v$, $\sigma_u = \mu$ and $\sigma_v = \nu$
  - Draw a bond with probability $p = 1 - e^{-1/T}$ and identify clusters

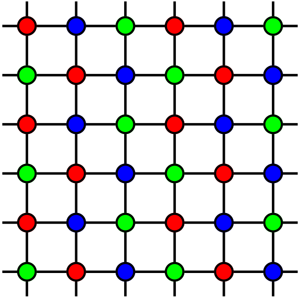
WSK algorithm

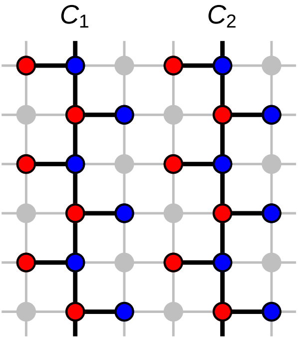




- Uniformly at random, choose two of the q possible colors $\mu, \nu \in \{1, 2 \ldots q\}$
- Freeze all the $\sigma_v \notin \{\mu, \nu\}$
- For each edge uv with with $\sigma_u \neq \sigma_v$, $\sigma_u = \mu$ and $\sigma_v = \nu$
  - Draw a bond with probability $p = 1 - e^{-1/T}$ and identify clusters

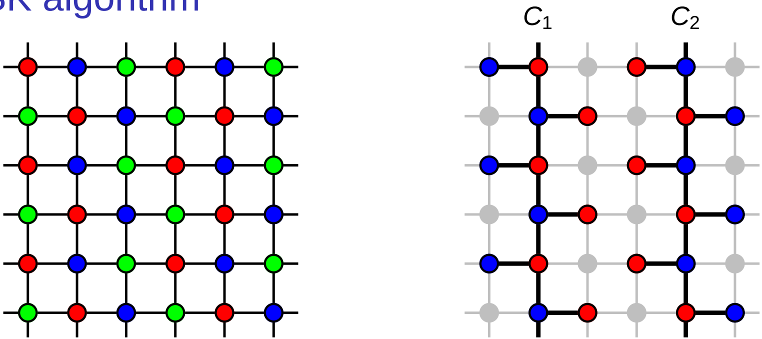
WSK algorithm





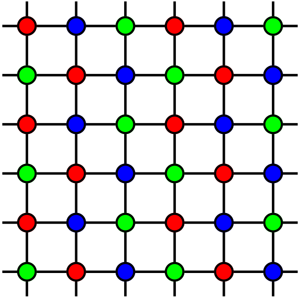
- Uniformly at random, choose two of the q possible colors $\mu, \nu \in \{1, 2 \ldots q\}$
- Freeze all the $\sigma_v \notin \{\mu, \nu\}$
- For each edge uv with with $\sigma_u \neq \sigma_v$, $\sigma_u = \mu$ and $\sigma_v = \nu$
  - Draw a bond with probability $p = 1 - e^{-1/T}$ and identify clusters
- For each cluster C_i flip the colouring with probability 1/2

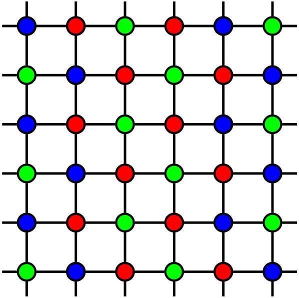
WSK algorithm



- Uniformly at random, choose two of the q possible colors $\mu, \nu \in \{1, 2 \ldots q\}$
- Freeze all the $\sigma_v \notin \{\mu, \nu\}$
- For each edge uv with with $\sigma_u \neq \sigma_v$, $\sigma_u = \mu$ and $\sigma_v = \nu$
  - Draw a bond with probability $p = 1 - e^{-1/T}$ and identify clusters
- For each cluster C_i flip the colouring with probability 1/2

WSK algorithm





- Uniformly at random, choose two of the q possible colors $\mu, \nu \in \{1, 2 \ldots q\}$
- Freeze all the $\sigma_v \notin \{\mu, \nu\}$
- For each edge uv with with $\sigma_u \neq \sigma_v$, $\sigma_u = \mu$ and $\sigma_v = \nu$
  - Draw a bond with probability $p = 1 - e^{-1/T}$ and identify clusters
- For each cluster C_i flip the colouring with probability 1/2

Rigorous results for WSK algorithm

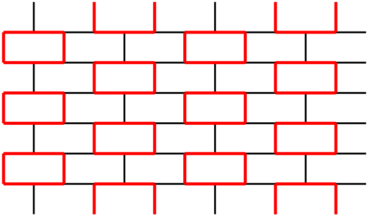
- Irreducible on all graphs when $T > 0$
- Irreducible on all graphs G when $T = 0$ and $q \geq \Delta(G) + 1$
- Irreducible on all bipartite graphs when $T = 0$
- Non-irreducible at $T = 0$:
  - $q = 4$ on triangular lattice (on torus) (Mohar & Salas 2009)
  - $q = 3$ on kagome lattice (on torus) (Mohar & Salas 2010)

Rigorous results for WSK algorithm

- Irreducible on all graphs when $T > 0$
- Irreducible on all graphs G when $T = 0$ and $q \geq \Delta(G) + 1$
- Irreducible on all bipartite graphs when $T = 0$
- Non-irreducible at $T = 0$:
  - $q = 4$ on triangular lattice (on torus) (Mohar & Salas 2009)
  - $q = 3$ on kagome lattice (on torus) (Mohar & Salas 2010)
- Worm algorithm for honeycomb-lattice fully-packed loop model can be used to simulate both these models

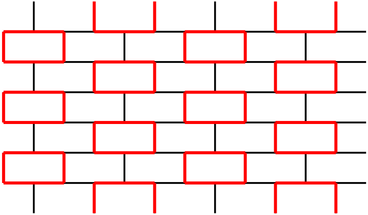
Zero-temperature Potts antiferromagnets

Use $n = 2$ worm algorithm to simulate coloured loop configurations



Zero-temperature Potts antiferromagnets

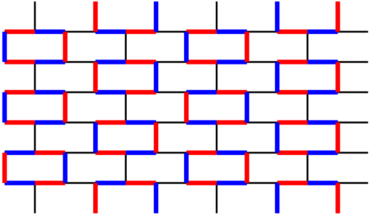
Use $n = 2$ worm algorithm to simulate coloured loop configurations



- Independently color each cycle, alternating red, blue, red, . . .

Zero-temperature Potts antiferromagnets

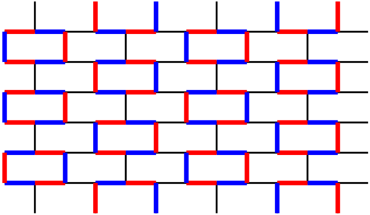
Use $n = 2$ worm algorithm to simulate coloured loop configurations



- Independently color each cycle, alternating red, blue, red, ...

Zero-temperature Potts antiferromagnets

Use $n = 2$ worm algorithm to simulate coloured loop configurations

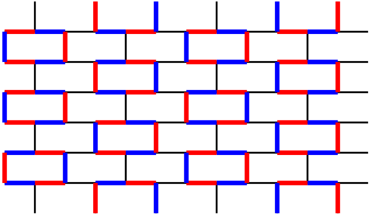


- Independently color each cycle, alternating red, blue, red, ...
- Gives dynamics on coloured loop states

$$P_{color}[a \to a'] = \frac{1}{2^{c(A')}} P_2^{worm}[A \to A']$$

Zero-temperature Potts antiferromagnets

Use $n = 2$ worm algorithm to simulate coloured loop configurations



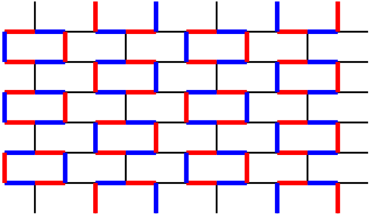
- Independently color each cycle, alternating red, blue, red, ...
- Gives dynamics on coloured loop states

$$P_{color}[a \to a'] = \frac{1}{2^{c(A')}} P_2^{worm}[A \to A']$$

- Uniformly sample 3-edge-colorings of honeycomb lattice

Zero-temperature Potts antiferromagnets

Use $n = 2$ worm algorithm to simulate coloured loop configurations



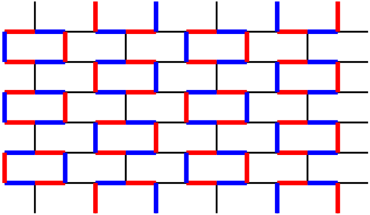
- Independently color each cycle, alternating red, blue, red, ...
- Gives dynamics on coloured loop states

$$P_{color}[a \to a'] = \frac{1}{2^{c(A')}} P_2^{worm}[A \to A']$$

- Uniformly sample 3-edge-colorings of honeycomb lattice
- Uniformly sample 3-vertex-colorings of kagome lattice

Zero-temperature Potts antiferromagnets

Use $n = 2$ worm algorithm to simulate coloured loop configurations



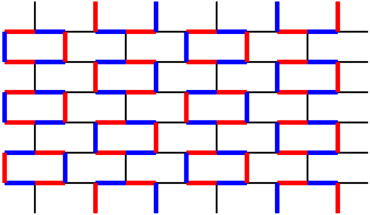
- Independently color each cycle, alternating red, blue, red, ...
- Gives dynamics on coloured loop states

$$P_{color}[a \to a'] = \frac{1}{2^{c(A')}} P_2^{worm}[A \to A']$$

- Uniformly sample 3-edge-colorings of honeycomb lattice
- Uniformly sample 3-vertex-colorings of kagome lattice
- Uniformly sample 4-vertex-colorings on triangular lattice (Baxter)

Zero-temperature Potts antiferromagnets

Use $n = 2$ worm algorithm to simulate coloured loop configurations



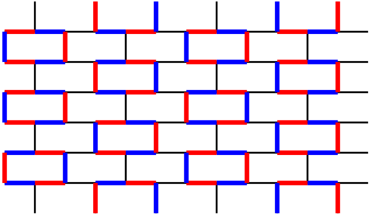
- Independently color each cycle, alternating red, blue, red, . . .
- Gives dynamics on coloured loop states

$$P_{color}[a \to a'] = \frac{1}{2^{c(A')}} P_2^{worm}[A \to A']$$

- Uniformly sample 3-edge-colorings of honeycomb lattice
- Uniformly sample 3-vertex-colorings of kagome lattice
- Uniformly sample 4-vertex-colorings on triangular lattice (Baxter)

Zero-temperature Potts antiferromagnets

Use $n = 2$ worm algorithm to simulate coloured loop configurations



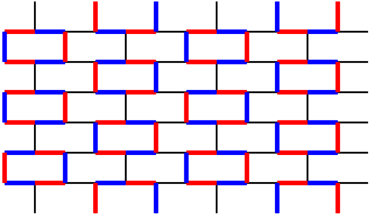
- Independently color each cycle, alternating red, blue, red, ...
- Gives dynamics on coloured loop states

$$P_{color}[a \to a'] = \frac{1}{2^{c(A')}} P_2^{worm}[A \to A']$$

- Uniformly sample 3-edge-colorings of honeycomb lattice
- Uniformly sample 3-vertex-colorings of kagome lattice
- Uniformly sample 4-vertex-colorings on triangular lattice (Baxter)
- Both models display critical phenomena

Zero-temperature Potts antiferromagnets

Use $n = 2$ worm algorithm to simulate coloured loop configurations



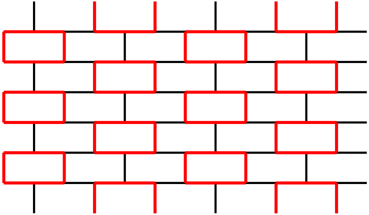
- Independently color each cycle, alternating red, blue, red, ...
- Gives dynamics on coloured loop states

$$P_{color}[a \to a'] = \frac{1}{2^{c(A')}} P_2^{worm}[A \to A']$$

- Uniformly sample 3-edge-colorings of honeycomb lattice
- Uniformly sample 3-vertex-colorings of kagome lattice
- Uniformly sample 4-vertex-colorings on triangular lattice (Baxter)
- Both models display critical phenomena
- Wang-Swendsen-Kotecký algorithm proved non-irreducible

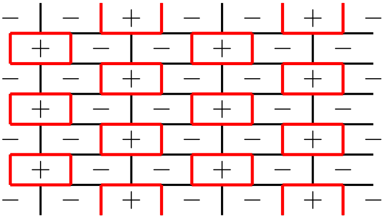
Triangular-lattice Ising antiferromagnet

Loops form boundaries of Ising spin domains on dual lattice



Triangular-lattice Ising antiferromagnet

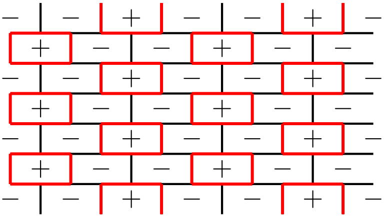
Loops form boundaries of Ising spin domains on dual lattice



Triangular-lattice Ising antiferromagnet

Loops form boundaries of Ising spin domains on dual lattice

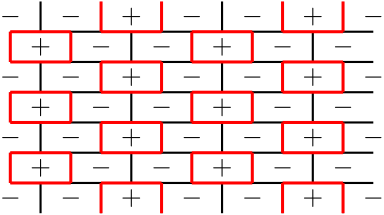
- Exact correspondence when $n = 1$ and $x = e^{-2\beta}$



Triangular-lattice Ising antiferromagnet

Loops form boundaries of Ising spin domains on dual lattice

- Exact correspondence when $n = 1$ and $x = e^{-2\beta}$

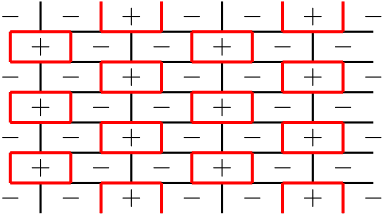


- $\phi_{H,n,x}(A_\sigma) = 2\,\mu_{H^*,\beta}(\sigma)$
- $A_\sigma := \{ij \in E : \sigma_{i^*} \neq \sigma_{j^*}\}$

Triangular-lattice Ising antiferromagnet

Loops form boundaries of Ising spin domains on dual lattice

- Exact correspondence when $n = 1$ and $x = e^{-2\beta}$

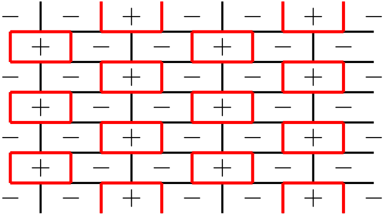


- $\phi_{H,n,x}(A_\sigma) = 2\,\mu_{H^*,\beta}(\sigma)$
- $A_\sigma := \{ij \in E : \sigma_{i^*} \neq \sigma_{j^*}\}$
- $x > 1$ implies antiferromagnetic β

Triangular-lattice Ising antiferromagnet

Loops form boundaries of Ising spin domains on dual lattice

- Exact correspondence when $n = 1$ and $x = e^{-2\beta}$

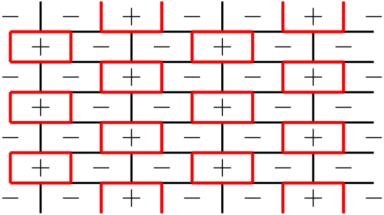


- $\phi_{H,n,x}(A_\sigma) = 2\,\mu_{H^*,\beta}(\sigma)$
- $A_\sigma := \{ij \in E : \sigma_{i^*} \neq \sigma_{j^*}\}$
- $x > 1$ implies antiferromagnetic β
- $x = +\infty$ corresponds to $\beta = -\infty$

Triangular-lattice Ising antiferromagnet

Loops form boundaries of Ising spin domains on dual lattice

- Exact correspondence when $n = 1$ and $x = e^{-2\beta}$

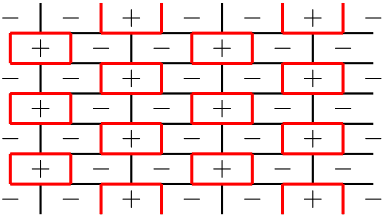


- $\phi_{H,n,x}(A_\sigma) = 2\,\mu_{H^*,\beta}(\sigma)$
- $A_\sigma := \{ij \in E : \sigma_{i^*} \neq \sigma_{j^*}\}$
- $x > 1$ implies antiferromagnetic β
- $x = +\infty$ corresponds to $\beta = -\infty$
- Both models are critical in this limit

Triangular-lattice Ising antiferromagnet

Loops form boundaries of Ising spin domains on dual lattice

- Exact correspondence when $n = 1$ and $x = e^{-2\beta}$



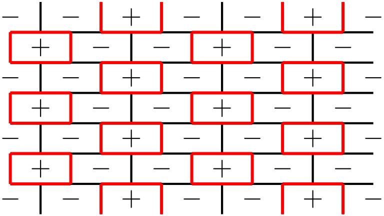
- $\phi_{H,n,x}(A_\sigma) = 2\,\mu_{H^*,\beta}(\sigma)$
- $A_\sigma := \{ij \in E : \sigma_{i^*} \neq \sigma_{j^*}\}$
- $x > 1$ implies antiferromagnetic β
- $x = +\infty$ corresponds to $\beta = -\infty$
- Both models are critical in this limit

- Can use worm to simulate AF Ising on $\triangle$-lattice at $T = 0$

Triangular-lattice Ising antiferromagnet

Loops form boundaries of Ising spin domains on dual lattice

- Exact correspondence when $n = 1$ and $x = e^{-2\beta}$



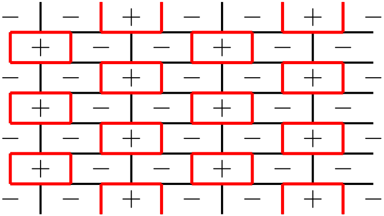
- $\phi_{H,n,x}(A_\sigma) = 2\,\mu_{H^*,\beta}(\sigma)$
- $A_\sigma := \{ij \in E : \sigma_{i^*} \neq \sigma_{j^*}\}$
- $x > 1$ implies antiferromagnetic β
- $x = +\infty$ corresponds to $\beta = -\infty$
- Both models are critical in this limit

- Can use worm to simulate AF Ising on $\triangle$-lattice at $T = 0$
- Single-spin-flip algorithms non-irreducible at $T = 0$

Triangular-lattice Ising antiferromagnet

Loops form boundaries of Ising spin domains on dual lattice

- Exact correspondence when $n = 1$ and $x = e^{-2\beta}$



- $\phi_{H,n,x}(A_\sigma) = 2\,\mu_{H^*,\beta}(\sigma)$
- $A_\sigma := \{ij \in E : \sigma_{i^*} \neq \sigma_{j^*}\}$
- $x > 1$ implies antiferromagnetic β
- $x = +\infty$ corresponds to $\beta = -\infty$
- Both models are critical in this limit

- Can use worm to simulate AF Ising on $\triangle$-lattice at $T = 0$
- Single-spin-flip algorithms non-irreducible at $T = 0$
- Tailor-made cluster algorithms non-irreducible at $T = 0$

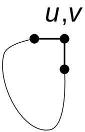
Summary

- Worm algorithms provide a simple way to simulate honeycomb-lattice loop models
  - Proven valid for all $n, x > 0$, including $x = +\infty$
- For $n = 1, 2$ also provide provably irreducible algorithms for certain critical antiferromagnetic models
  - Cluster algorithms fail for these models

Structure of the defect cluster

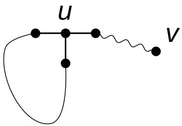
Partition $\mathcal{R}(G)$ according to structure of defect cluster

$$\mathcal{R} = \mathcal{E} \cup \mathcal{T} \cup \mathcal{D} \cup \Theta$$



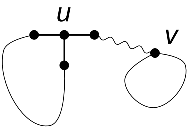
$(A, u, v) \in \mathcal{E}$

Cycle



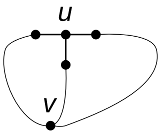
$(A, u, v) \in \mathcal{T}$

Tadpole



$(A, u, v) \in \mathcal{D}$

Dumbbell



$(A, u, v) \in \Theta$

Theta graph

Worm algorithm for fully-packed loop model

- Transition matrix:

$$P_n[(A,u,v) \to (A \triangle uu', u', v)] = P_n[(A,v,u) \to (A \triangle uu', v, u')]$$
$$= \begin{cases} 1/2 & (A,u,v) \in \mathcal{E}, \\ 1/2 & (A,u,v) \in \mathcal{T}, \\ 1/6 & (A,u,v) \in \Theta, \\ n/2(n+2) & (A,u,v) \in \mathcal{D} \text{ and } uu' \text{ is a bridge}, \\ 1/2(n+2) & (A,u,v) \in \mathcal{D} \text{ and } uu' \text{ is not a bridge}. \end{cases}$$

- Stationary distribution:

$$\pi_n(A,u,v) = \frac{n^{c(A)}}{Z_n} \begin{cases} 1/3 & (A,u,v) \in \mathcal{E}, \\ 1/3 & (A,u,v) \in \mathcal{T}, \\ (n+2)/3n & (A,u,v) \in \mathcal{D}, \\ 1/n & (A,u,v) \in \Theta. \end{cases}$$

Connectivity checking

$$P_{n,x}[(A,u,v) \to (A \triangle uu', u', v)] = P_{n,x}[(A,v,u) \to (A \triangle uu', v, u')]$$

$$= \frac{1}{2\,d(u)} \begin{cases} F(x\,n) & uu' \notin A \text{ and } u \leftrightarrow u' \text{ in } (V,A) \\ F(x) & uu' \notin A \text{ and } u \not\leftrightarrow u' \text{ in } (V,A) \\ F(1/n\,x) & uu' \in A \text{ and } u \leftrightarrow u' \text{ in } (V, A \setminus uu') \\ F(1/x) & uu' \in A \text{ and } u \not\leftrightarrow u' \text{ in } (V, A \setminus uu') \end{cases}$$

- So far we have glossed over an important caveat:

Connectivity checking

$$P_{n,x}[(A,u,v) \to (A \triangle uu', u', v)] = P_{n,x}[(A,v,u) \to (A \triangle uu', v, u')]$$

$$= \frac{1}{2\,d(u)} \begin{cases} F(x\,n) & uu' \notin A \text{ and } u \leftrightarrow u' \text{ in } (V,A) \\ F(x) & uu' \notin A \text{ and } u \not\leftrightarrow u' \text{ in } (V,A) \\ F(1/n\,x) & uu' \in A \text{ and } u \leftrightarrow u' \text{ in } (V, A \setminus uu') \\ F(1/x) & uu' \in A \text{ and } u \not\leftrightarrow u' \text{ in } (V, A \setminus uu') \end{cases}$$

- So far we have glossed over an important caveat:
  - If $n \neq 1$ we need to compute $c(A \triangle uu') - c(A)$ at each iteration

Connectivity checking

$$P_{n,x}[(A,u,v) \to (A\triangle uu', u', v)] = P_{n,x}[(A,v,u) \to (A\triangle uu', v, u')]$$

$$= \frac{1}{2\,d(u)} \begin{cases} F(x\,n) & uu' \notin A \text{ and } u \leftrightarrow u' \text{ in } (V,A) \\ F(x) & uu' \notin A \text{ and } u \not\leftrightarrow u' \text{ in } (V,A) \\ F(1/n\,x) & uu' \in A \text{ and } u \leftrightarrow u' \text{ in } (V,A \setminus uu') \\ F(1/x) & uu' \in A \text{ and } u \not\leftrightarrow u' \text{ in } (V,A \setminus uu') \end{cases}$$

- So far we have glossed over an important caveat:
  - If $n \neq 1$ we need to compute $c(A\triangle uu') - c(A)$ at each iteration
- But $c(A) = |A| - |V| + k(A)$
  - $k(A)$ is the number of components in (V,A)

Connectivity checking

$$P_{n,x}[(A,u,v) \to (A \triangle uu', u', v)] = P_{n,x}[(A,v,u) \to (A \triangle uu', v, u')]$$

$$= \frac{1}{2\,d(u)} \begin{cases} F(x\,n) & uu' \notin A \text{ and } u \leftrightarrow u' \text{ in } (V,A) \\ F(x) & uu' \notin A \text{ and } u \not\leftrightarrow u' \text{ in } (V,A) \\ F(1/n\,x) & uu' \in A \text{ and } u \leftrightarrow u' \text{ in } (V, A \setminus uu') \\ F(1/x) & uu' \in A \text{ and } u \not\leftrightarrow u' \text{ in } (V, A \setminus uu') \end{cases}$$

- So far we have glossed over an important caveat:
  - If $n \neq 1$ we need to compute $c(A \triangle uu') - c(A)$ at each iteration
- But $c(A) = |A| - |V| + k(A)$
  - $k(A)$ is the number of components in (V, A)
- Dynamic connectivity-checking algorithms are known
  - Log-time for queries and updates (Holm, de Lichtenberg & Thorup)

Connectivity checking

$$P_{n,x}[(A,u,v) \to (A\triangle uu',u',v)] = P_{n,x}[(A,v,u) \to (A\triangle uu',v,u')]$$

$$= \frac{1}{2\,d(u)} \begin{cases} F(x\,n) & uu' \notin A \text{ and } u \leftrightarrow u' \text{ in } (V,A) \\ F(x) & uu' \notin A \text{ and } u \not\leftrightarrow u' \text{ in } (V,A) \\ F(1/n\,x) & uu' \in A \text{ and } u \leftrightarrow u' \text{ in } (V,A\setminus uu') \\ F(1/x) & uu' \in A \text{ and } u \not\leftrightarrow u' \text{ in } (V,A\setminus uu') \end{cases}$$

- So far we have glossed over an important caveat:
  - If $n \neq 1$ we need to compute $c(A\triangle uu') - c(A)$ at each iteration
- But $c(A) = |A| - |V| + k(A)$
  - $k(A)$ is the number of components in (V,A)
- Dynamic connectivity-checking algorithms are known
  - Log-time for queries and updates (Holm, de Lichtenberg & Thorup)
- Simultaneous Breadth-First-Search (BFS) much simpler
  - Polynomial-time with small (known) exponent (k-arm)

Connectivity checking

$$P_{n,x}[(A,u,v)\to(A\triangle uu',u',v)] = P_{n,x}[(A,v,u)\to(A\triangle uu',v,u')]$$

$$= \frac{1}{2\,d(u)}\begin{cases} F(x\,n) & uu'\notin A \text{ and } u\leftrightarrow u' \text{ in } (V,A) \\ F(x) & uu'\notin A \text{ and } u\not\leftrightarrow u' \text{ in } (V,A) \\ F(1/n\,x) & uu'\in A \text{ and } u\leftrightarrow u' \text{ in } (V,A\setminus uu') \\ F(1/x) & uu'\in A \text{ and } u\not\leftrightarrow u' \text{ in } (V,A\setminus uu') \end{cases}$$

- So far we have glossed over an important caveat:
  - If $n\neq 1$ we need to compute $c(A\triangle uu') - c(A)$ at each iteration
- But $c(A) = |A| - |V| + k(A)$
  - $k(A)$ is the number of components in (V,A)
- Dynamic connectivity-checking algorithms are known
  - Log-time for queries and updates (Holm, de Lichtenberg & Thorup)
- Simultaneous Breadth-First-Search (BFS) much simpler
  - Polynomial-time with small (known) exponent (k-arm)
- If $n>1$ the “colouring method” avoids the issue entirely

The colouring method

- Consider a finite graph $G = (V, E)$
- Let $K(A)$ denote the set of connected components of (V, A)
- Define a "generalized random-cluster model"

$$\mathbb{P}_W(A) \propto \prod_{C \in K(A)} W(C) \qquad A \subseteq E$$

- $W(\cdot) \geq 0$ assigns a weight to each connected subgraph of G

The colouring method

- Consider a finite graph $G = (V, E)$
- Let $K(A)$ denote the set of connected components of (V, A)
- Define a "generalized random-cluster model"

$$\mathbb{P}_W(A) \propto \prod_{C \in K(A)} W(C) \qquad A \subseteq E$$

- $W(\cdot) \geq 0$ assigns a weight to each connected subgraph of G
- If $W(C) = q\, v^{|E(C)|}$ this is just the Fortuin-Kasteleyn (FK) model

The colouring method

- Consider a finite graph $G = (V, E)$
- Let $K(A)$ denote the set of connected components of (V, A)
- Define a "generalized random-cluster model"

$$\mathbb{P}_W(A) \propto \prod_{C \in K(A)} W(C) \qquad A \subseteq E$$

- $W(\cdot) \geq 0$ assigns a weight to each connected subgraph of G
- If $W(C) = q\, v^{|E(C)|}$ this is just the Fortuin-Kasteleyn (FK) model
- Loop model corresponds to

$$W(C) = \begin{cases} n\, x^{|E(C)|} & C \text{ is a cycle or isolated vertex} \\ 0 & \text{otherwise} \end{cases}$$

The colouring method

- Consider a finite graph $G = (V, E)$
- Let $K(A)$ denote the set of connected components of (V, A)
- Define a "generalized random-cluster model"

$$\mathbb{P}_W(A) \propto \prod_{C \in K(A)} W(C) \qquad A \subseteq E$$

- $W(\cdot) \geq 0$ assigns a weight to each connected subgraph of G
- If $W(C) = q\, v^{|E(C)|}$ this is just the Fortuin-Kasteleyn (FK) model
- Loop model corresponds to

$$W(C) = \begin{cases} n\, x^{|E(C)|} & C \text{ is a cycle or isolated vertex} \\ 0 & \text{otherwise} \end{cases}$$

- Simulate $\mathbb{P}_W$ by introducing auxiliary vertex variables (colours)
- Like SW, choose bonds conditioned on colours & vice versa

The colouring method (2)

- Introduce (vertex) m-colourings $\sigma \in \{1, 2, \ldots m\}^V$

The colouring method (2)

- Introduce (vertex) m-colourings $\sigma \in \{1, 2, \ldots m\}^V$
- For each colour α define a new weight function W_α so that

$$W(C) = \sum_{\alpha=1}^{m} W_\alpha(C) \qquad \text{and} \qquad W_\alpha(C) \geq 0$$

The colouring method (2)

- Introduce (vertex) m-colourings $\sigma \in \{1, 2, \dots m\}^V$
- For each colour α define a new weight function W_α so that

$$W(C) = \sum_{\alpha=1}^{m} W_\alpha(C) \qquad \text{and} \qquad W_\alpha(C) \geq 0$$

1. Given $A \subseteq E$ let C be coloured α with $\mathbb{P}(\sigma_C = \alpha) = W_\alpha(C)/W(C)$

The colouring method (2)

- Introduce (vertex) m-colourings $\sigma \in \{1, 2, \ldots m\}^V$
- For each colour α define a new weight function W_α so that

$$W(C) = \sum_{\alpha=1}^{m} W_\alpha(C) \qquad \text{and} \qquad W_\alpha(C) \geq 0$$

1. Given $A \subseteq E$ let C be coloured α with $\mathbb{P}(\sigma_C = \alpha) = W_\alpha(C)/W(C)$
2. Choose a new bond configuration on G_α using any Markov chain for which W_α is stationary

The colouring method (2)

- Introduce (vertex) m-colourings $\sigma \in \{1, 2, \ldots m\}^V$
- For each colour α define a new weight function W_α so that

$$W(C) = \sum_{\alpha=1}^{m} W_\alpha(C) \qquad \text{and} \qquad W_\alpha(C) \geq 0$$

1. Given $A \subseteq E$ let C be coloured α with $\mathbb{P}(\sigma_C = \alpha) = W_\alpha(C)/W(C)$
2. Choose a new bond configuration on G_α using any Markov chain for which W_α is stationary

- Trick is to choose at least one W_α which is easy to simulate

The colouring method (2)

- Introduce (vertex) m-colourings $\sigma \in \{1, 2, \ldots m\}^V$
- For each colour α define a new weight function W_α so that

$$W(C) = \sum_{\alpha=1}^{m} W_\alpha(C) \qquad \text{and} \qquad W_\alpha(C) \geq 0$$

1. Given $A \subseteq E$ let C be coloured α with $\mathbb{P}(\sigma_C = \alpha) = W_\alpha(C)/W(C)$
2. Choose a new bond configuration on G_α using any Markov chain for which W_α is stationary

- Trick is to choose at least one W_α which is easy to simulate
- Can update other $W_{\alpha'}$ by "doing nothing" (identity matrix)

The colouring method (2)

- Introduce (vertex) m-colourings $\sigma \in \{1, 2, \dots m\}^V$
- For each colour α define a new weight function W_α so that

$$W(C) = \sum_{\alpha=1}^{m} W_\alpha(C) \qquad \text{and} \qquad W_\alpha(C) \geq 0$$

1. Given $A \subseteq E$ let C be coloured α with $\mathbb{P}(\sigma_C = \alpha) = W_\alpha(C)/W(C)$
2. Choose a new bond configuration on G_α using any Markov chain for which W_α is stationary

- Trick is to choose at least one W_α which is easy to simulate
- Can update other $W_{\alpha'}$ by "doing nothing" (identity matrix)
- FK model: take $W_\alpha = 1$ (percolation is easy to simulate)

The colouring method (2)

- Introduce (vertex) m-colourings $\sigma \in \{1, 2, \ldots m\}^V$
- For each colour α define a new weight function W_α so that

$$W(C) = \sum_{\alpha=1}^{m} W_\alpha(C) \qquad \text{and} \qquad W_\alpha(C) \geq 0$$

1. Given $A \subseteq E$ let C be coloured α with $\mathbb{P}(\sigma_C = \alpha) = W_\alpha(C)/W(C)$
2. Choose a new bond configuration on G_α using any Markov chain for which W_α is stationary

- Trick is to choose at least one W_α which is easy to simulate
- Can update other $W_{\alpha'}$ by "doing nothing" (identity matrix)
- FK model: take $W_\alpha = 1$ (percolation is easy to simulate)
- Loop model: take $W_\alpha = 1$ (Ising is easy to simulate using worm)

The colouring method (2)

- Introduce (vertex) m-colourings $\sigma \in \{1, 2, \dots m\}^V$
- For each colour α define a new weight function W_α so that

$$W(C) = \sum_{\alpha=1}^{m} W_\alpha(C) \qquad \text{and} \qquad W_\alpha(C) \geq 0$$

1. Given $A \subseteq E$ let C be coloured α with $\mathbb{P}(\sigma_C = \alpha) = W_\alpha(C)/W(C)$
2. Choose a new bond configuration on G_α using any Markov chain for which W_α is stationary

- Trick is to choose at least one W_α which is easy to simulate
- Can update other $W_{\alpha'}$ by "doing nothing" (identity matrix)
- FK model: take $W_\alpha = 1$ (percolation is easy to simulate)
- Loop model: take $W_\alpha = 1$ (Ising is easy to simulate using worm)
- Get algorithms for $q > 1$ FK and $n > 1$ loop models

The colouring method (2)

- Introduce (vertex) m-colourings $\sigma \in \{1, 2, \dots m\}^V$
- For each colour α define a new weight function W_α so that

$$W(C) = \sum_{\alpha=1}^{m} W_\alpha(C) \qquad \text{and} \qquad W_\alpha(C) \geq 0$$

1. Given $A \subseteq E$ let C be coloured α with $\mathbb{P}(\sigma_C = \alpha) = W_\alpha(C)/W(C)$
2. Choose a new bond configuration on G_α using any Markov chain for which W_α is stationary

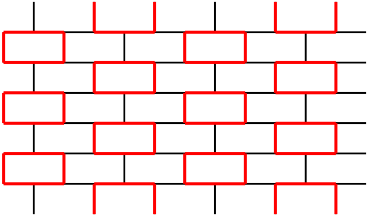
- Trick is to choose at least one W_α which is easy to simulate
- Can update other $W_{\alpha'}$ by "doing nothing" (identity matrix)
- FK model: take $W_\alpha = 1$ (percolation is easy to simulate)
- Loop model: take $W_\alpha = 1$ (Ising is easy to simulate using worm)
- Get algorithms for $q > 1$ FK and $n > 1$ loop models
  - No connectivity-checking needed

SW for antiferromagnetic Ising

SW for antiferromagnetic Ising model on finite graph $G = (V, E)$

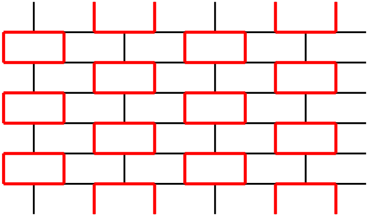
$$\mathbb{P}(\sigma, n) \propto \prod_{ij \in E} [(1-p)\,\delta_{\omega_{ij},0} + p(1-\delta_{\sigma_i,\sigma_j})\delta_{\omega_{ij},1}]$$

Mapping 3-edge colorings to dual 4-vertex colorings



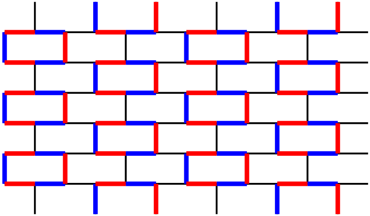
Mapping 3-edge colorings to dual 4-vertex colorings

- Randomly color the cycles $\iff$ proper edge 3-coloring



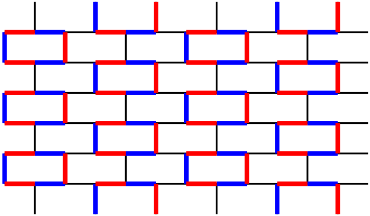
Mapping 3-edge colorings to dual 4-vertex colorings

- Randomly color the cycles $\iff$ proper edge 3-coloring



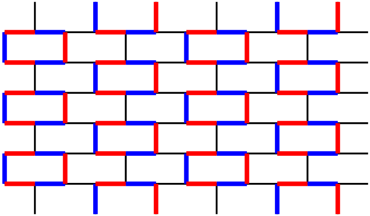
Mapping 3-edge colorings to dual 4-vertex colorings

- Randomly color the cycles $\iff$ proper edge 3-coloring
- Use two independent Ising variables on each face, σ, τ



Mapping 3-edge colorings to dual 4-vertex colorings

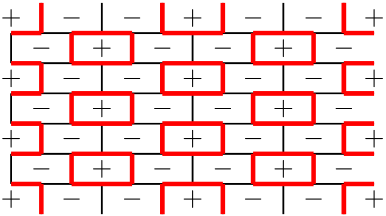
- Randomly color the cycles $\iff$ proper edge 3-coloring
- Use two independent Ising variables on each face, σ, τ



- Union of red and vacant edges = spin domain for σ

Mapping 3-edge colorings to dual 4-vertex colorings

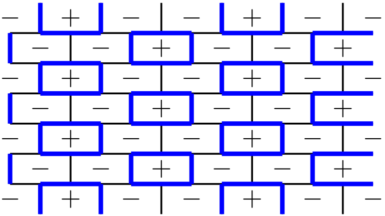
- Randomly color the cycles $\iff$ proper edge 3-coloring
- Use two independent Ising variables on each face, σ, τ



- Union of red and vacant edges
= spin domain for σ

Mapping 3-edge colorings to dual 4-vertex colorings

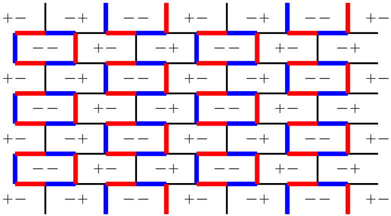
- Randomly color the cycles $\iff$ proper edge 3-coloring
- Use two independent Ising variables on each face, σ, τ



- Union of red and vacant edges = spin domain for σ
- Union of blue and vacant edges = spin domain for τ

Mapping 3-edge colorings to dual 4-vertex colorings

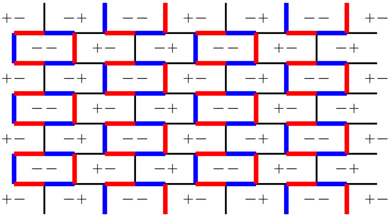
- Randomly color the cycles $\iff$ proper edge 3-coloring
- Use two independent Ising variables on each face, σ, τ



- Union of red and vacant edges = spin domain for σ
- Union of blue and vacant edges = spin domain for τ
- Combine the σ and τ configurations

Mapping 3-edge colorings to dual 4-vertex colorings

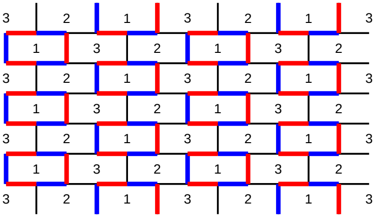
- Randomly color the cycles $\iff$ proper edge 3-coloring
- Use two independent Ising variables on each face, σ, τ



- Union of red and vacant edges = spin domain for σ
- Union of blue and vacant edges = spin domain for τ
- Combine the σ and τ configurations
- $-- = 1$, $-+ = 2$, $+- = 3$, $++ = 4$

Mapping 3-edge colorings to dual 4-vertex colorings

- Randomly color the cycles $\iff$ proper edge 3-coloring
- Use two independent Ising variables on each face, σ, τ



- Union of red and vacant edges = spin domain for σ
- Union of blue and vacant edges = spin domain for τ
- Combine the σ and τ configurations
- $-- = 1$, $-+ = 2$, $+- = 3$, $++ = 4$

Markov-chain Monte Carlo

- Markov chain
  - State space S, with $|S| < \infty$
  - Transition matrix P
  - Stationary distribution π

Markov-chain Monte Carlo

- Markov chain
  - State space S, with $|S| < \infty$
  - Transition matrix P
  - Stationary distribution π
- Observables (random variables) $X, Y, \ldots$

Markov-chain Monte Carlo

- Markov chain
  - State space S, with $|S| < \infty$
  - Transition matrix P
  - Stationary distribution π
- Observables (random variables) $X, Y, \ldots$
- Simulate Markov chain $s_0 \xrightarrow{P} s_1 \xrightarrow{P} s_2 \xrightarrow{P} \ldots$ with $s_t \in S$

Markov-chain Monte Carlo

- Markov chain
  - State space S, with $|S| < \infty$
  - Transition matrix P
  - Stationary distribution π
- Observables (random variables) $X, Y, \ldots$
- Simulate Markov chain $s_0 \xrightarrow{P} s_1 \xrightarrow{P} s_2 \xrightarrow{P} \ldots$ with $s_t \in S$
- Get time series $X_0, X_1, X_2, \ldots$ with $X_t = X(s_t)$

Markov-chain Monte Carlo

- Markov chain
  - State space S, with $|S| < \infty$
  - Transition matrix P
  - Stationary distribution π
- Observables (random variables) $X, Y, \ldots$
- Simulate Markov chain $s_0 \xrightarrow{P} s_1 \xrightarrow{P} s_2 \xrightarrow{P} \ldots$ with $s_t \in S$
- Get time series $X_0, X_1, X_2, \ldots$ with $X_t = X(s_t)$
- Define the autocorrelation function

$$\rho_X(t) := \frac{\langle X_s X_{s+t} \rangle_\pi - \langle X \rangle_\pi^2}{\text{var}_\pi(X)}$$

Markov-chain Monte Carlo

- Markov chain
  - State space S, with $|S| < \infty$
  - Transition matrix P
  - Stationary distribution π
- Observables (random variables) $X, Y, \ldots$
- Simulate Markov chain $s_0 \xrightarrow{P} s_1 \xrightarrow{P} s_2 \xrightarrow{P} \ldots$ with $s_t \in S$
- Get time series $X_0, X_1, X_2, \ldots$ with $X_t = X(s_t)$
- Define the autocorrelation function

$$\rho_X(t) := \frac{\langle X_s X_{s+t} \rangle_\pi - \langle X \rangle_\pi^2}{\mathrm{var}_\pi(X)}$$

- Stationary process – start “in equilibrium”

Integrated autocorrelation times

- The integrated autocorrelation time

$$\tau_{\mathrm{int},X} := \frac{1}{2} \sum_{t=-\infty}^{\infty} \rho_X(t)$$

Integrated autocorrelation times

- The integrated autocorrelation time

$$\tau_{\text{int},X} := \frac{1}{2} \sum_{t=-\infty}^{\infty} \rho_X(t)$$

- If $\widehat{X}$ is the sample mean of $\{X_t\}_{t=1}^{T}$ then we have

$$\text{var}(\widehat{X}) \sim 2\,\tau_{\text{int},X} \frac{\text{var}(X)}{T}, \qquad T \to \infty$$

Integrated autocorrelation times

- The integrated autocorrelation time

$$\tau_{\text{int},X} := \frac{1}{2} \sum_{t=-\infty}^{\infty} \rho_X(t)$$

- If $\widehat{X}$ is the sample mean of $\{X_t\}_{t=1}^{T}$ then we have

$$\text{var}(\widehat{X}) \sim 2\,\tau_{\text{int},X} \frac{\text{var}(X)}{T}, \qquad T \to \infty$$

- 1 “effectively independent” observation every $2\,\tau_{\text{int},X}$ steps

Exponential autocorrelation times

- $\rho_X(t)$ typically decays exponentially as $t \to \infty$
- The exponential autocorrelation time

$$\tau_{\exp,X} := \limsup_{t\to\infty} \frac{t}{-\log|\rho_X(t)|} \quad \text{and} \quad \tau_{\exp} := \sup_X \tau_{\exp,X}$$

Exponential autocorrelation times

- $\rho_X(t)$ typically decays exponentially as $t \to \infty$
- The exponential autocorrelation time

$$\tau_{\exp,X} := \limsup_{t\to\infty} \frac{t}{-\log|\rho_X(t)|} \quad \text{and} \quad \tau_{\exp} := \sup_X \tau_{\exp,X}$$

  - Typically $\tau_{\exp,X} = \tau_{\exp} < \infty$ and $\tau_{\text{int},X} \le \tau_{\exp}$ for all X

Exponential autocorrelation times

- $\rho_X(t)$ typically decays exponentially as $t \to \infty$
- The exponential autocorrelation time

$$\tau_{\exp,X} := \limsup_{t\to\infty} \frac{t}{-\log|\rho_X(t)|} \quad \text{and} \quad \tau_{\exp} := \sup_X \tau_{\exp,X}$$

  - Typically $\tau_{\exp,X} = \tau_{\exp} < \infty$ and $\tau_{\text{int},X} \leq \tau_{\exp}$ for all X
- Start the chain with arbitrary distribution α
- Distribution at time t is αP^t

Exponential autocorrelation times

- $\rho_X(t)$ typically decays exponentially as $t \to \infty$
- The exponential autocorrelation time

$$\tau_{\exp,X} := \limsup_{t\to\infty} \frac{t}{-\log|\rho_X(t)|} \quad \text{and} \quad \tau_{\exp} := \sup_X \tau_{\exp,X}$$

  - Typically $\tau_{\exp,X} = \tau_{\exp} < \infty$ and $\tau_{\text{int},X} \le \tau_{\exp}$ for all X
- Start the chain with arbitrary distribution α
- Distribution at time t is αP^t

Lemma

αP^t tends to π with rate bounded by $e^{-t/\tau_{\exp}}$

Critical slowing-down

- Near a critical point the autocorrelation times typically diverge like

$$\tau \sim \xi^{z}$$

Critical slowing-down

- Near a critical point the autocorrelation times typically diverge like

$$\tau \sim \xi^z$$

- More precisely, we have a family of exponents:
  z_{exp}, and $z_{\text{int},X}$ for each observable X.

Critical slowing-down

- Near a critical point the autocorrelation times typically diverge like

$$\tau \sim \xi^z$$

- More precisely, we have a family of exponents:
  z_{exp}, and $z_{\mathrm{int},X}$ for each observable X.
- Different algorithms for the same model can have very different z
- E.g. $d = 2$ Ising model
  - Glauber (Metropolis) algorithm $z \approx 2$
  - Swendsen-Wang algorithm $z \approx 0.2$