Linear time perfect simulation for Markov random fields

Mark Huber
Fletcher Jones Foundation Associate Professor of Mathematics and Statistics and George R. Roberts Fellow
Chair of the Department of Mathematical Sciences
Claremont McKenna College

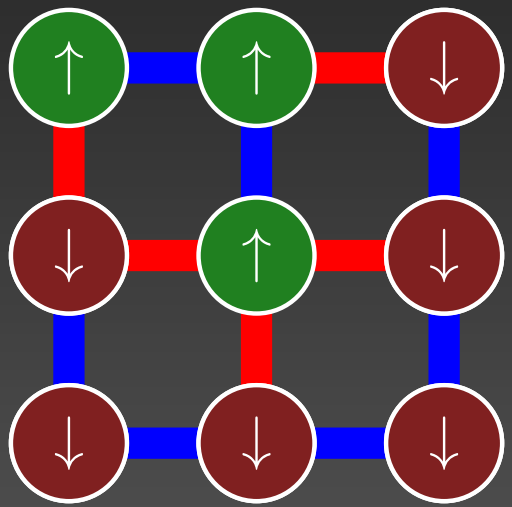
27 July, 2017

The goal

Exactly sample in linear expected time from high dimensional distributions where dimensions interact weakly and locally

Example: The Ising model

- Start with a graph $G = (V, E)$
- Each node labeled either $\uparrow$ or $\downarrow$
- Are endpoints of an edge labeled the same way?
- Give each edge with different labels a weight factor $\exp(-\beta)$
- Multiply weight factors to get the overall weight



$$w(x) = \exp(-5\beta)$$

The Big Picture

- Coupling from the past is not the only perfect simulation protocol
- One such is Partially Recursive Acceptance Rejection
- Using Cluster PRAR, get a linear time algorithm on nontrivial class of Markov random fields

Acceptance Rejection

Example: The Ising model mathematically

Define weight function

$$w(x) = \prod_{\{i,j\} \in E} \exp(-\beta \mathbb{1}(x(i) = x(j)))$$

It needs to be normalized to be a probability distribution

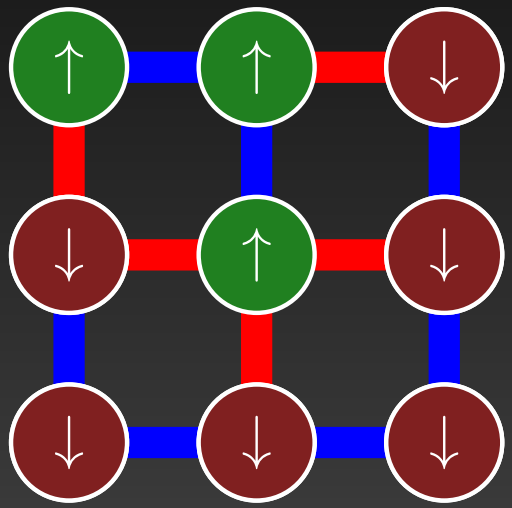
$$\mathbb{P}(X = x) = \frac{w(x)}{Z_\beta} = \frac{w(x)}{\sum_{y \in \{0,1\}^V} w(y)}$$

Note that when $\beta \geq 0$

$$w(x) \leq 1$$

Using the red edges for AR

For given X, say that edge $\{i, j\}$ is red if $X_i \neq X_j$

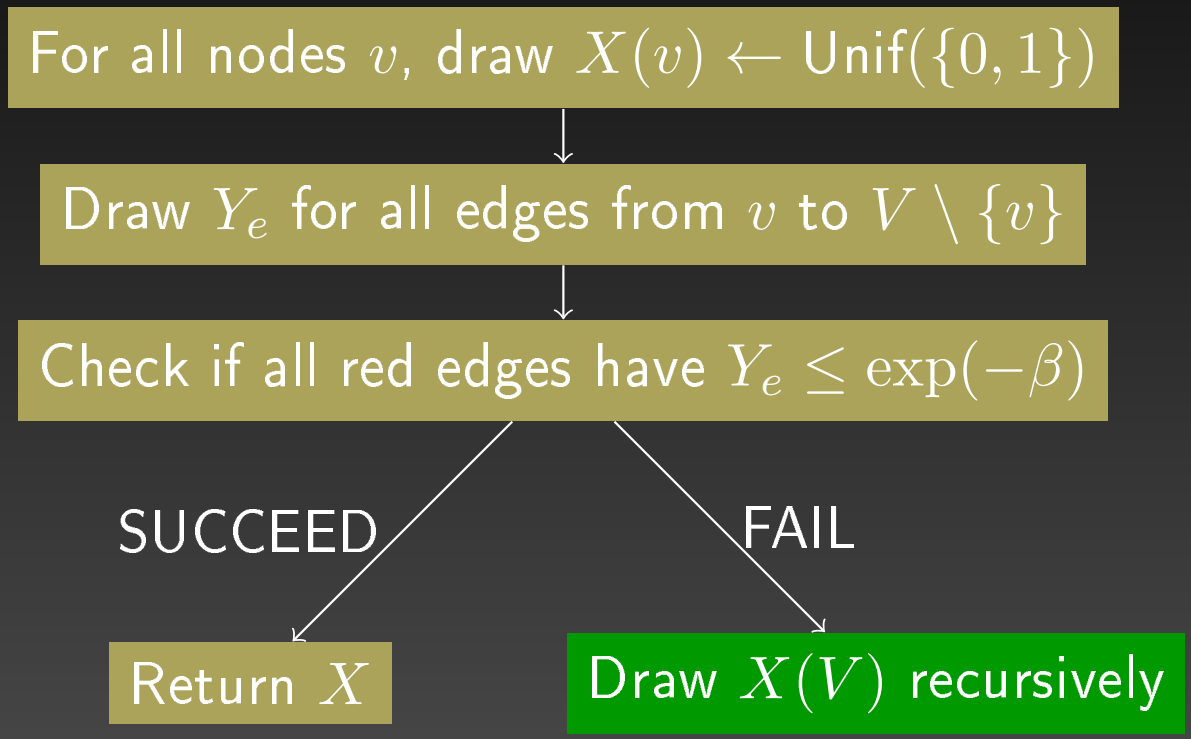


$$w(x) = \prod_{i=1}^{5} \exp(-\beta)$$

For each red edge in a randomly drawn sample, have a $\exp(-\beta)$ chance of accepting

AR in block form

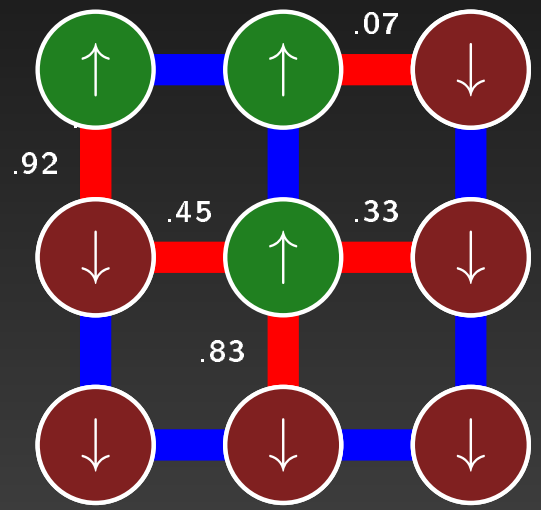
Green blocks are recursive calls



Acceptance/Rejection

`Edge_based_AR`

1. Draw each node X_i iid Unif($\{0, 1\}$)
   $\{i, j\}$ red if $X_i \neq X_j$
2. For each edge $\{i, j\}$
   draw $Y_i \sim$ Unif($[0, 1]$)
3. If for all red edges $Y_e \leq \exp(-\beta)$,
   then return X and quit
4. Else,
   draw $X \leftarrow$ `Edge_based_AR`
   return X and quit



Acceptance/Rejection too slow

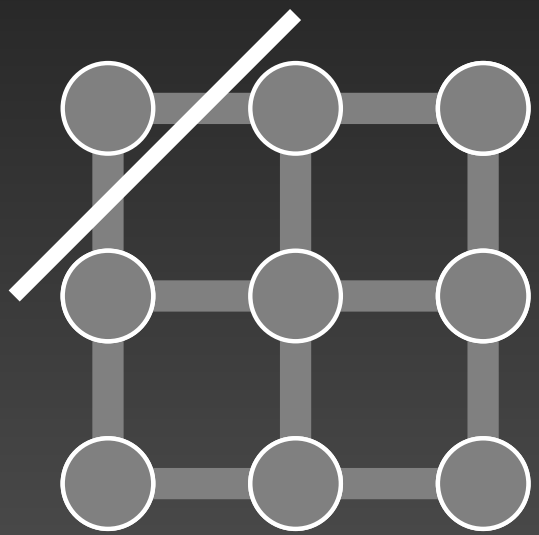
- On average, half the edges will be red
- Chance Y_i works for all red edges $\approx \exp(-\beta \#(E)/2)$
- Number of times `Edge_based_AR` calls itself on average $\approx \exp(\beta \#(E)/2)$

Recursive AR

Recursive AR

Subdivide the problem

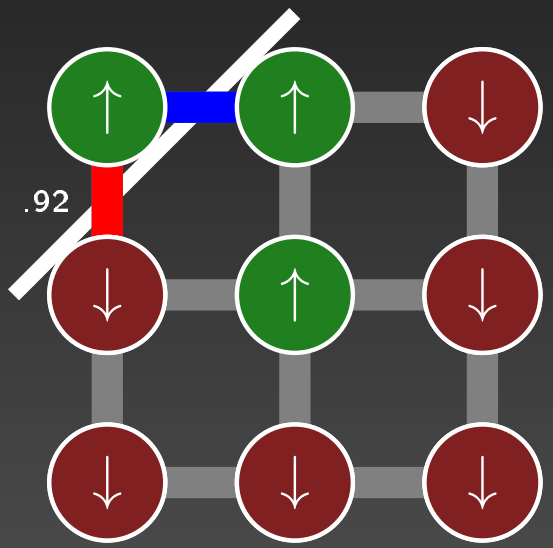
- Say $\{v\}$ and $V \setminus \{v\}$
- Generate $X(\{v\})$ and $X(V \setminus \{v\})$ separately
- For AR only have to worry about edges that cross the cut



Recursive AR

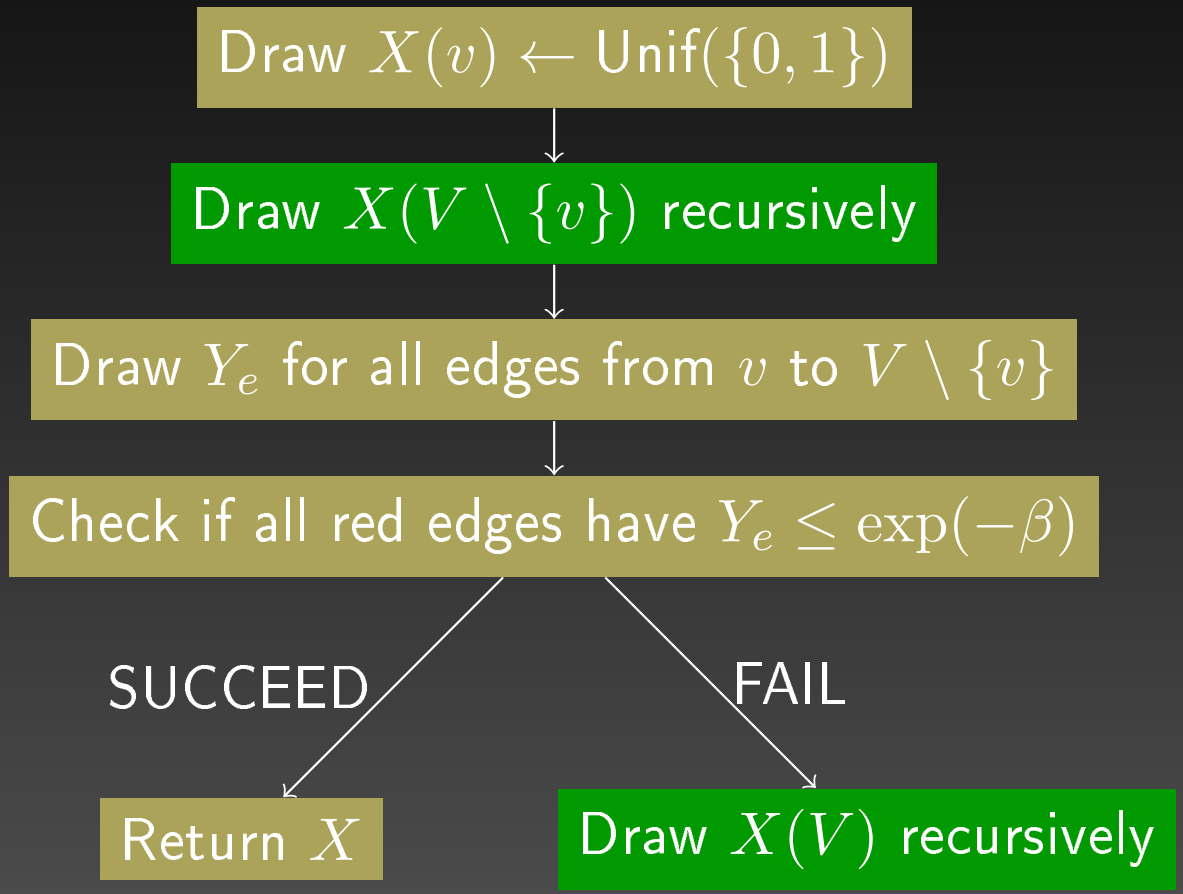
Subdivide the problem

- Say $\{v\}$ and $V \setminus \{v\}$
- Generate $X(\{v\})$ and $X(V \setminus \{v\})$ separately
- For AR only have to worry about edges that cross the cut



In block form

Green blocks are recursive calls

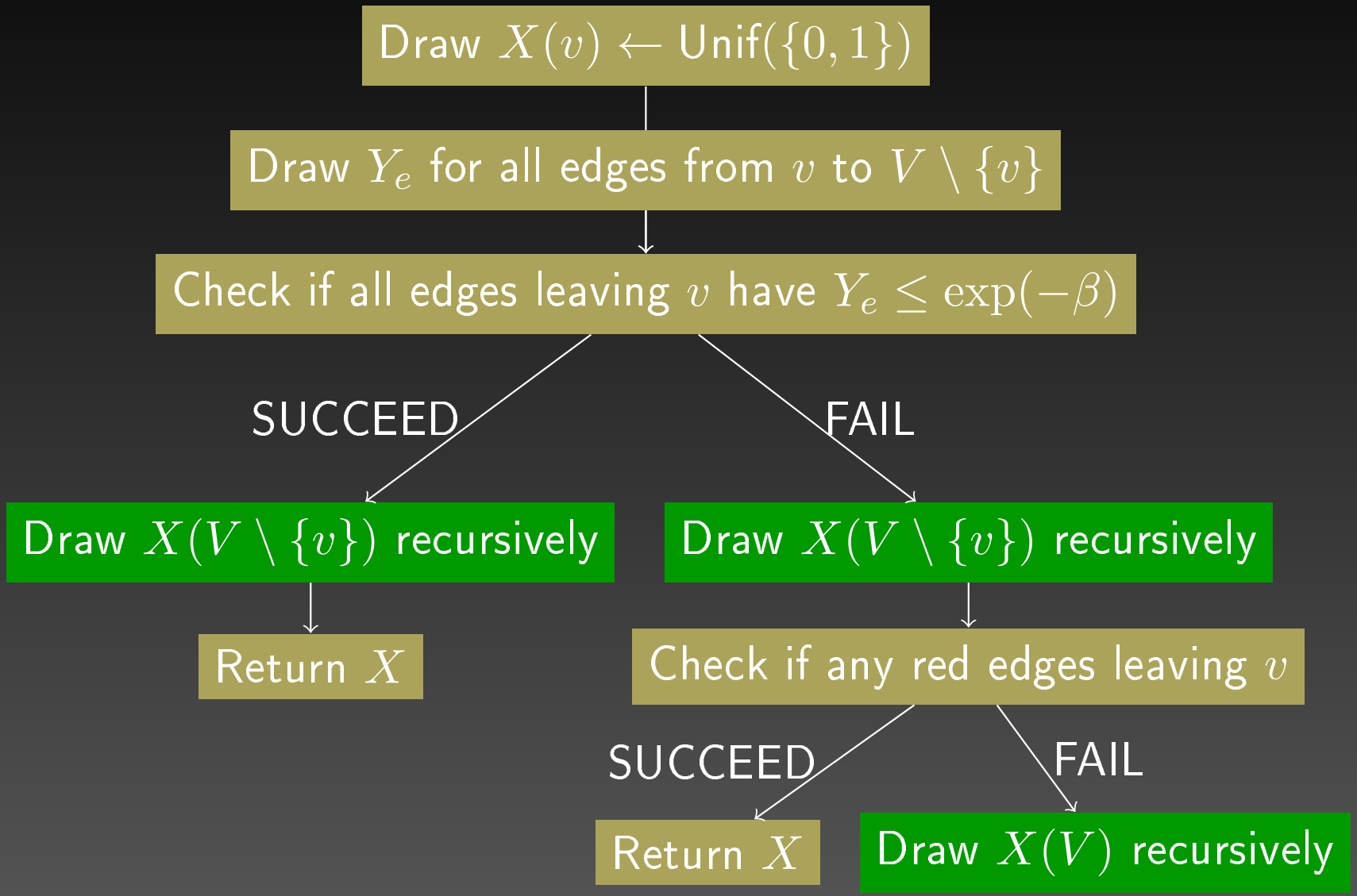


First key insight

Remark

- If $Y_{\{v,w\}} \leq \exp(-\beta)$ for all w neighboring v, then we don't need to know what $X(V \setminus \{v\})$ is!
- We can fix $X(v)$, and then find $X(V \setminus \{v\})$ recursively.

In block form



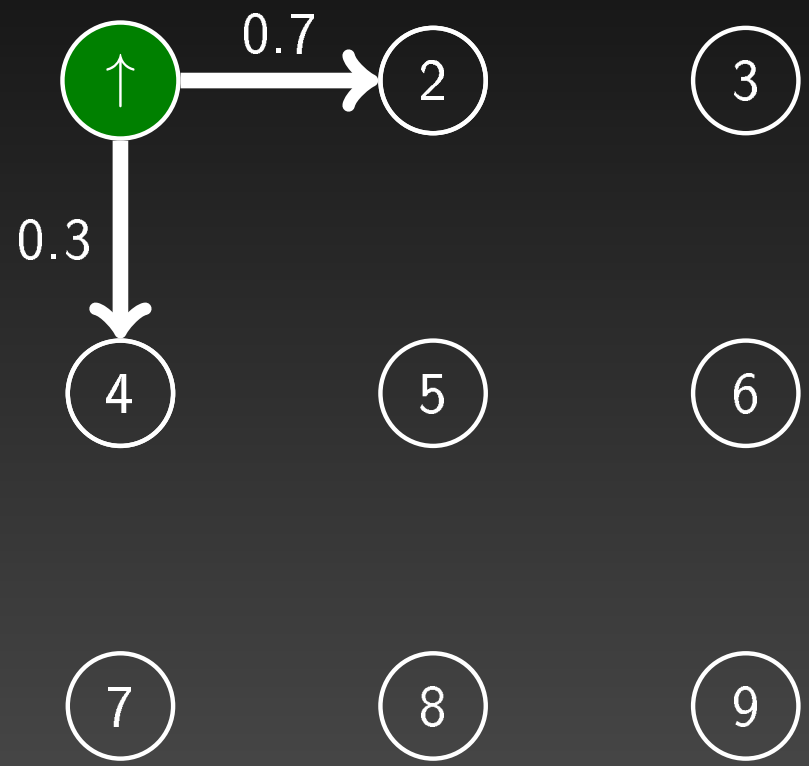
How to make it even better

This is better than before

- If for all edges leaving v, $Y_e \leq \exp(-\beta)$, then node v is fixed and never examined again
- Doing too much work on the FAIL side though
- Do not need entirely of $X(V \setminus \{v\})$ to determine next SUCCEED/FAIL
- Only need values of $X(w)$ where $Y_{\{v,w\}} > \exp(-\beta)$
- So only find those values recursively
- Ends up giving a cluster of values

Example: Ising model with $\exp(-\beta) = 2/3$

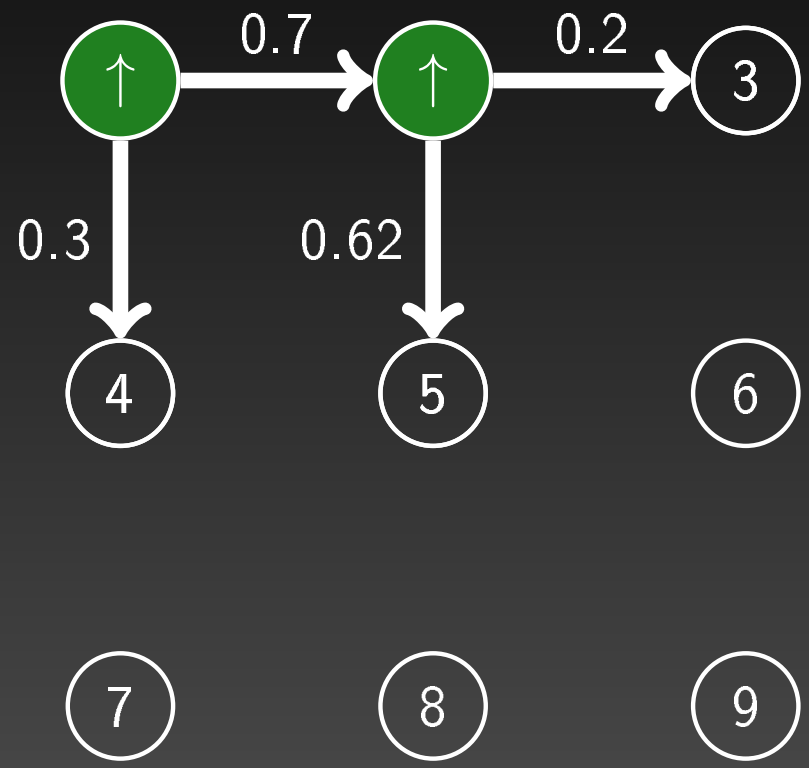
Start by assigning X_1, $Y_{\{1,2\}}$ and $Y_{\{1,4\}}$



Only need to learn X_2 to continue

Example: Ising model with $\exp(-\beta) = 2/3$

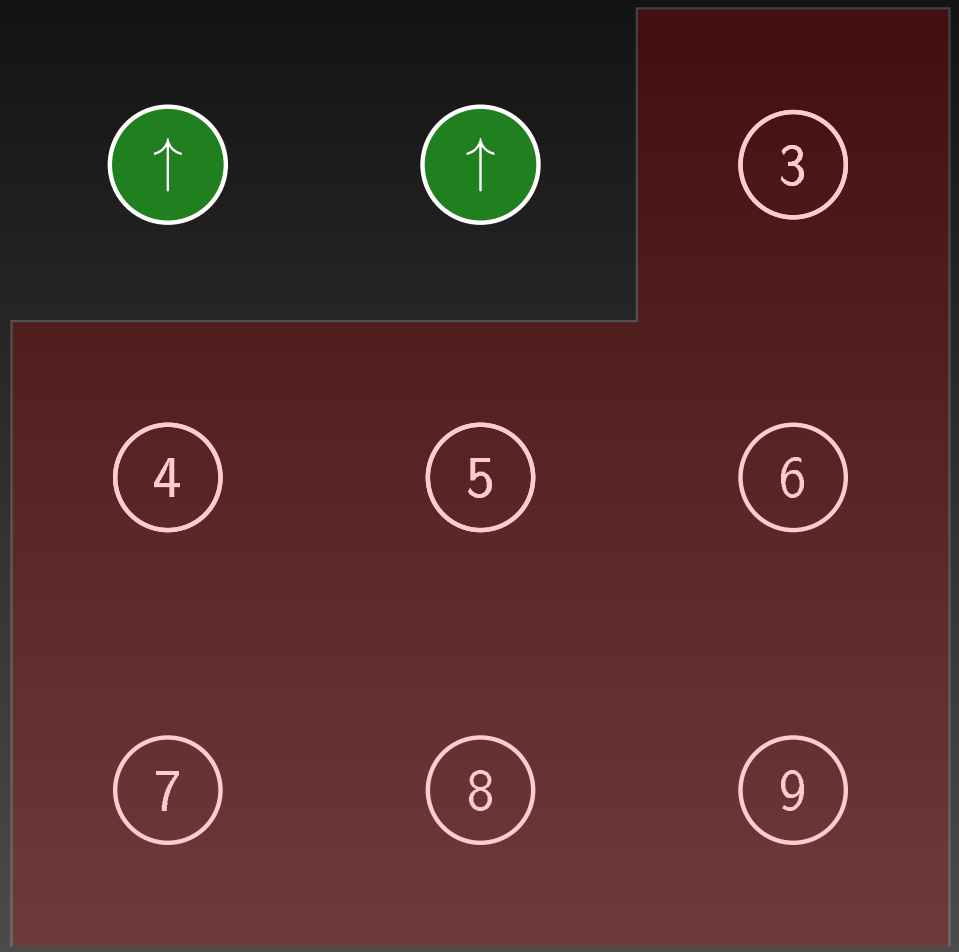
Assign X_2, $Y_{\{2,3\}}$, $Y_{\{2,5\}}$



Accept $X_2 = \uparrow$, $X_1 = \uparrow$

Example: Ising model with $\exp(-\beta) = 2/3$

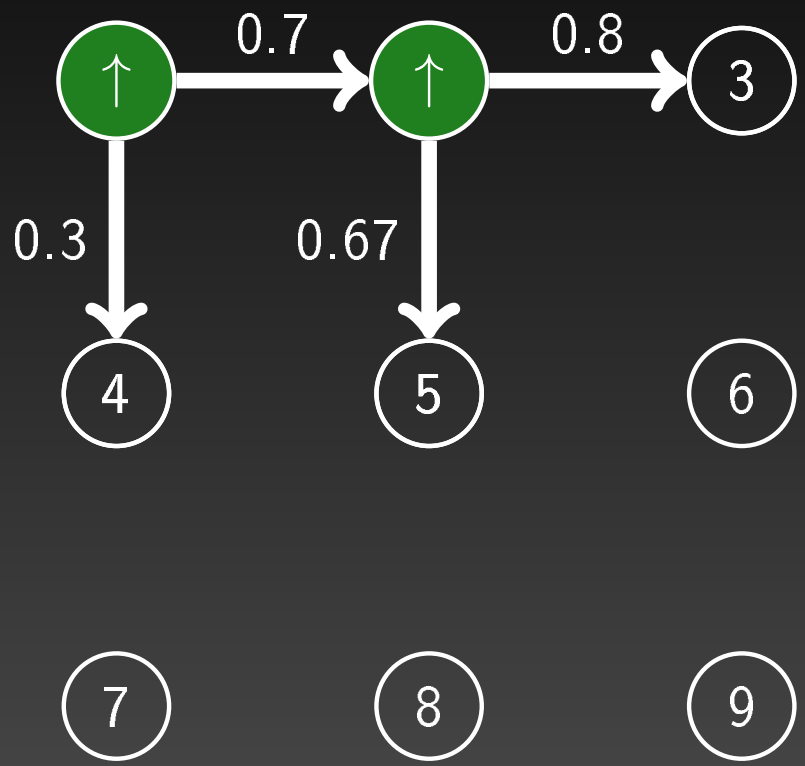
Fix X_1, X_2, draw sample for rest of nodes recursively



Accept $X_2 = \uparrow$, $X_1 = \uparrow$

Depth first search to resolve nodes

Assign X_2, $Y_{\{2,3\}}$, $Y_{\{2,5\}}$



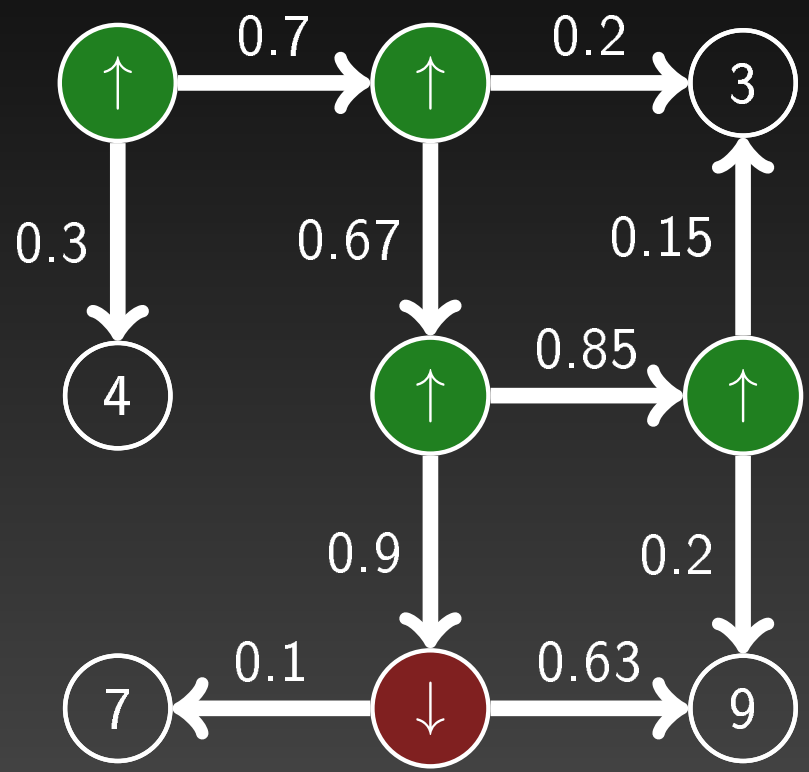
Resolve X_3 completely before resolving X_5

If $X_3 = \downarrow$, then reject $X_2 = \uparrow$, do not need to resolve X_5!

Principles of Cluster PRAR

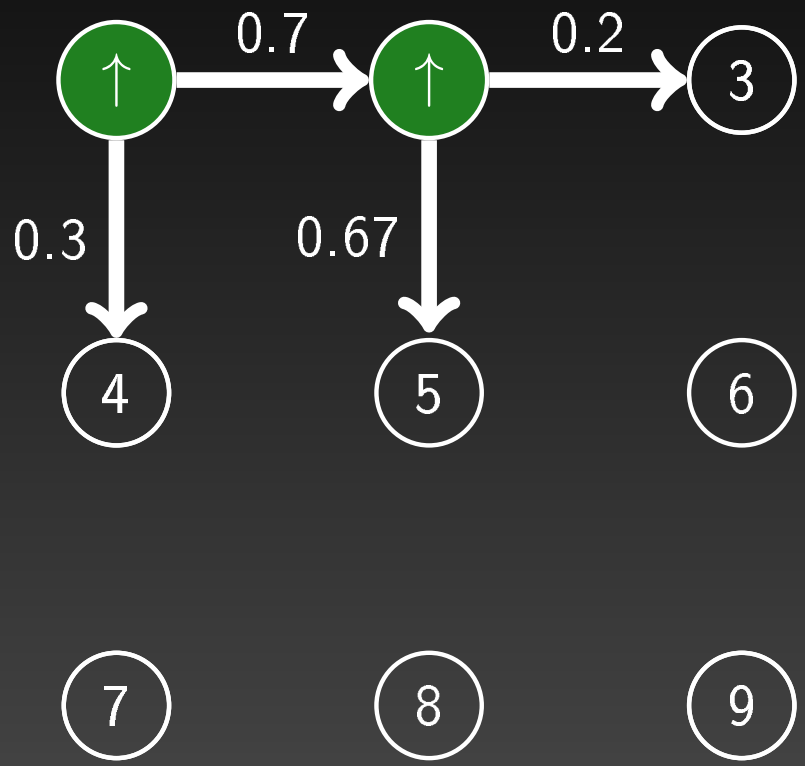
1. Depth first search in resolving nodes
2. Always have directed cluster of nodes
3. If node rejects, remove entire subcluster

Example:



X_8 causes X_5 to reject, so X_6 reset as well

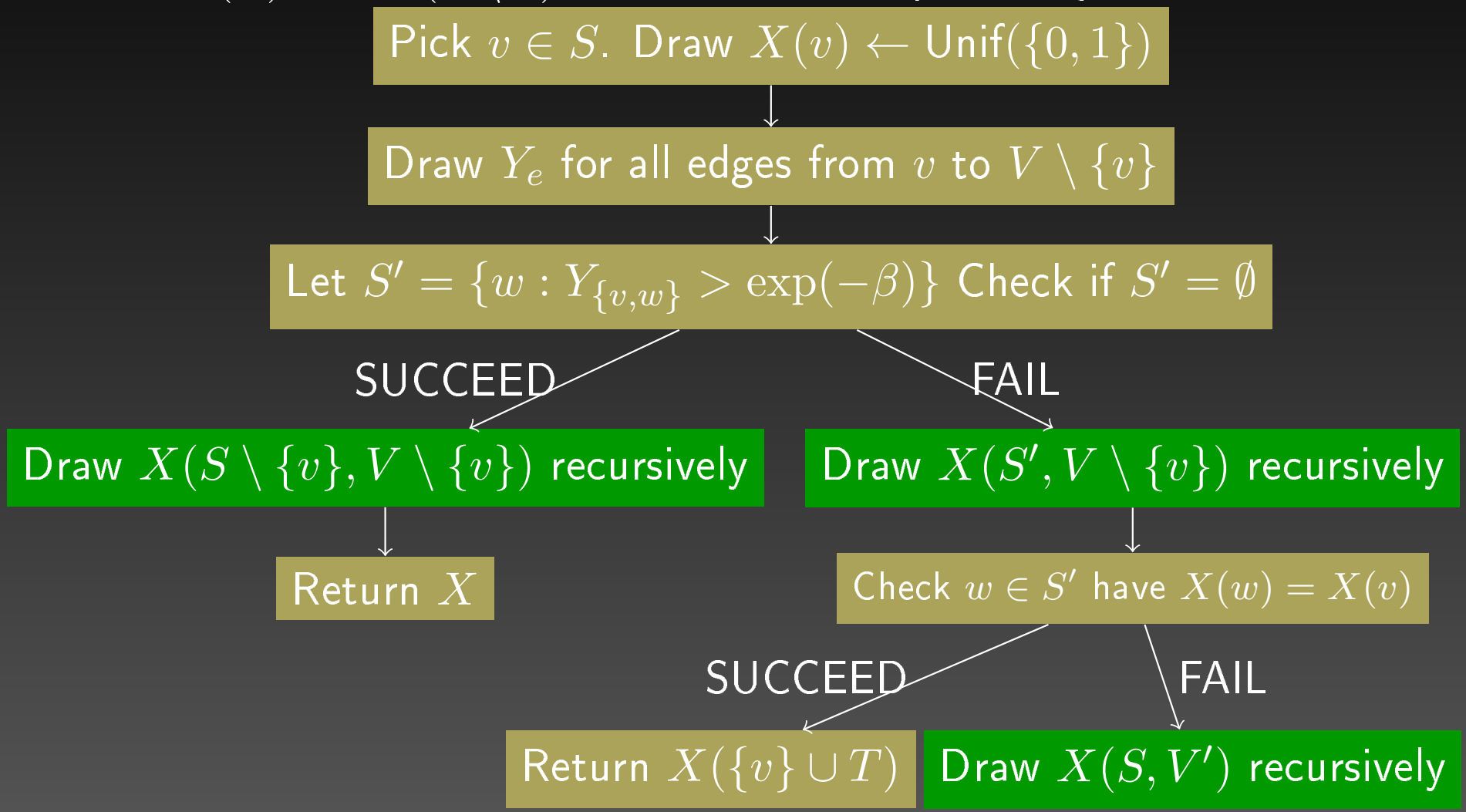
Example after rejection



X_8 causes X_5 to reject, so X_6 reset as well

In block form

Input: (S, V') Output: $X(T)$ where $S \subset T$
and $X(T)$ and $X(V \setminus T)$ can be drawn independently to form $X \sim \pi$



Cluster PRAR

Cluster PRAR returns value of X on one or more nodes that contain v:

`CPRAR_Ising`(v, V) *Output:* $T, X(T)$

Let $T \leftarrow \{v\}$, and draw $X(v)$ uniform over $\{\uparrow, \downarrow\}$
For each neighbor w of v, draw $Y_w \leftarrow \text{Unif}([0,1])$
$S \leftarrow \{w : Y_w > \exp(-\beta)\}$
While $(\exists w \in S \setminus T)$
  Let w be any element of $S \setminus T$
  $(T', X(T')) \leftarrow$ `CPRAR_Ising`$(w, V \setminus T)$
  $T \leftarrow T \cup T'$
  If a neighbor w of v has $X(w) \neq X(v)$ then $T \leftarrow \emptyset$, $S \leftarrow \emptyset$
If $(T = \emptyset)$ then $(T, X(T)) \leftarrow$ `CPRAR_Ising`(v, V)

The Running time

Want a finite expected time to decide to accept or reject $X(v)$

- What is expected change in number of nodes w where we need to know $X(w)$?
- Always remove one such node
- If Δ is maximum degree in graph, expect to add $\exp(-\beta)(\Delta - 1)$ new nodes

$$-1 + \exp(-\beta)(\Delta - 1) < 0 \Leftrightarrow \exp(-\beta) \leq \frac{1}{\Delta - 1}$$

Running time result

Lemma

Let $\gamma = \exp(-\beta)(\Delta - 1) - 1$. *When* $\gamma < 0$, `CPRAR_Ising` *makes an expected number of choices for* X *at nodes and* Y *at edges bounded above by*

$$\Delta + 1 + \Delta/(-\gamma)$$

Does it work?

The Fundamental Theorem of Perfect Simulation

Theorem (H. 2015, 2017)

(Intuitive version) As long as the algorithm terminates with probability 1, then you can assume that any recursive calls to the same algorithm are correct when proving that the original algorithm is correct.

Popping

Popping algorithms

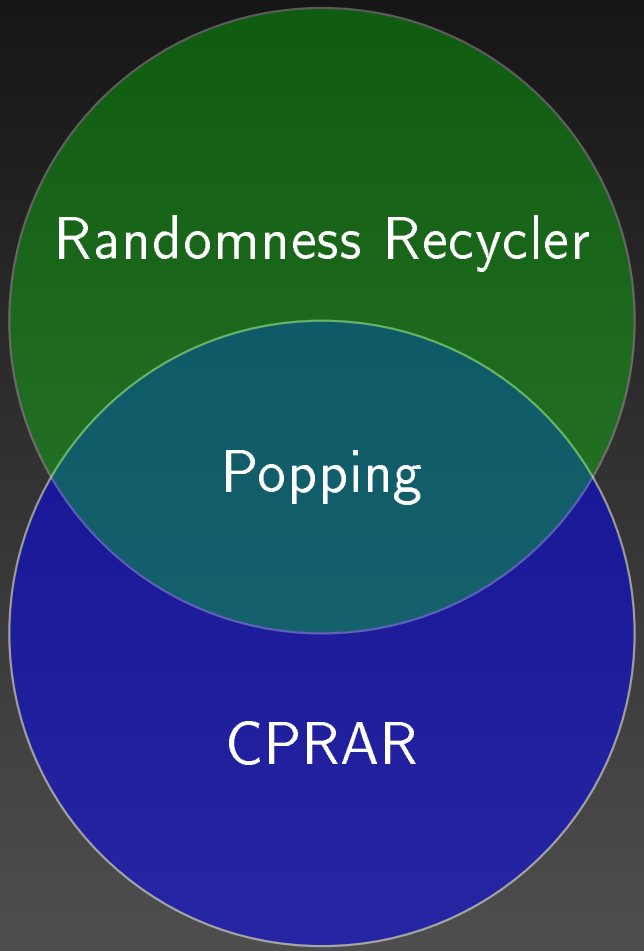
Generating random spanning trees more quickly than the cover time
D. B. Wilson
Proc. 28th ACM Sympos. on the Theory of Comp., *296–303, 1996*

Generating a random sink-free orientation in quadratic time
H. Cohn, R. Pemantle, and J. Propp
Elec. J. Of Combinatorics, *9, #R10, 2001*

Two popping algorithms

- Loop erased random walk for uniform random directed spanning trees
- Sink-popping for sink-free orientations of a graph

Generalizing popping algorithms

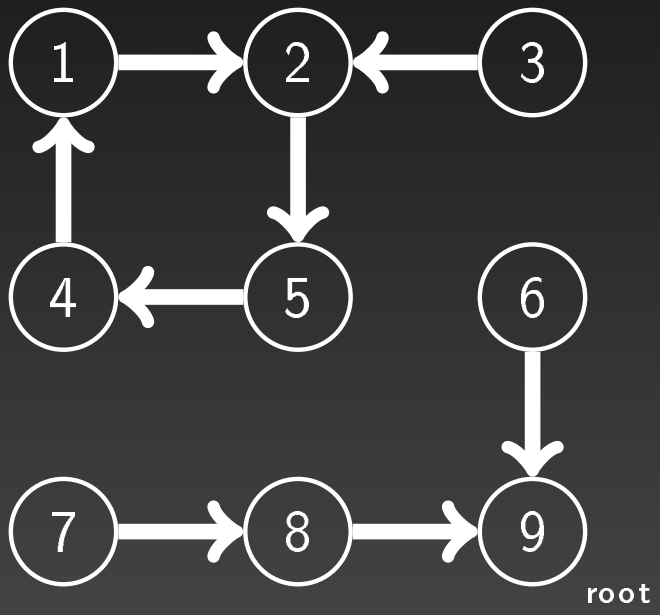


The Randomness Recycler: A New Approach to Perfect Sampling
J. A. Fill and M. L. Huber
Proc. 41st Sympos. on Foundations of Comp. Sci., *503–511, 2000*

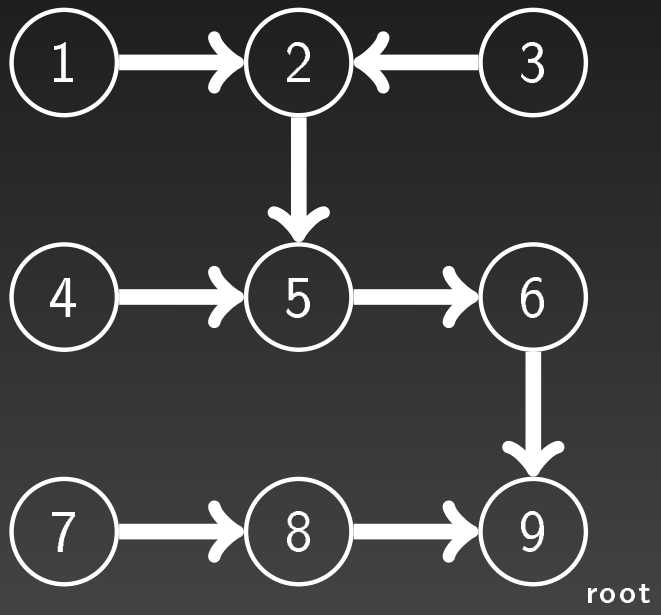
Random directed spanning tree

To generate uniformly over RDST's:

- For every node other than root, uniformly choose neighbor
- Repeat until there are no cycles



Reject

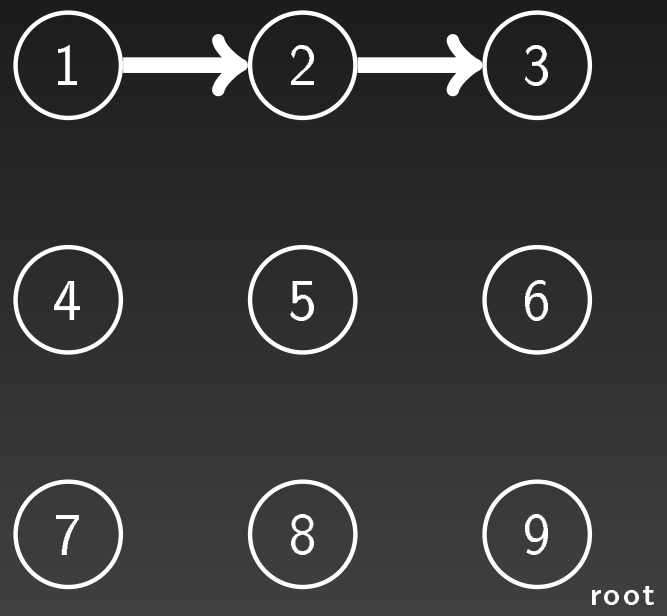


Accept

CPRAR gives loop-erased random walk

- For each node as we recurse, draw direction uniformly from neighbors

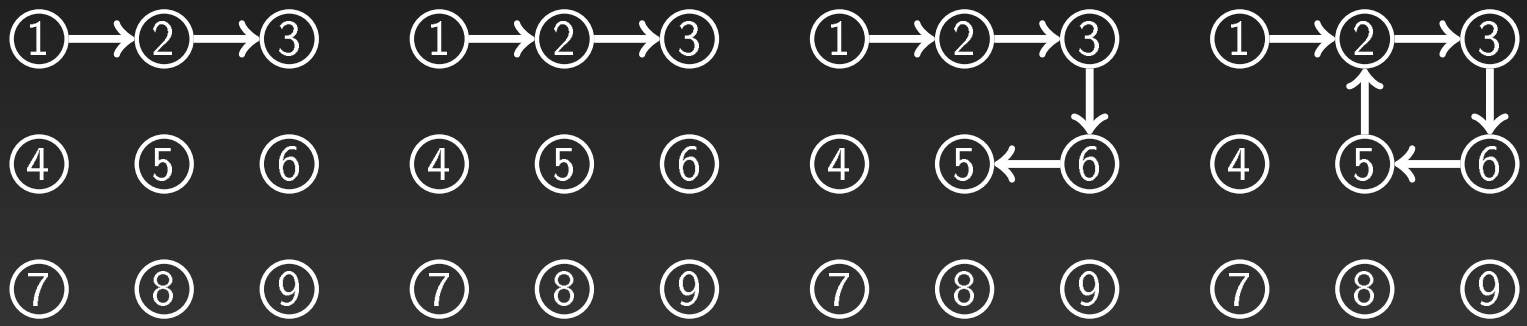
First step



Second step

CPRAR gives loop-erased random walk

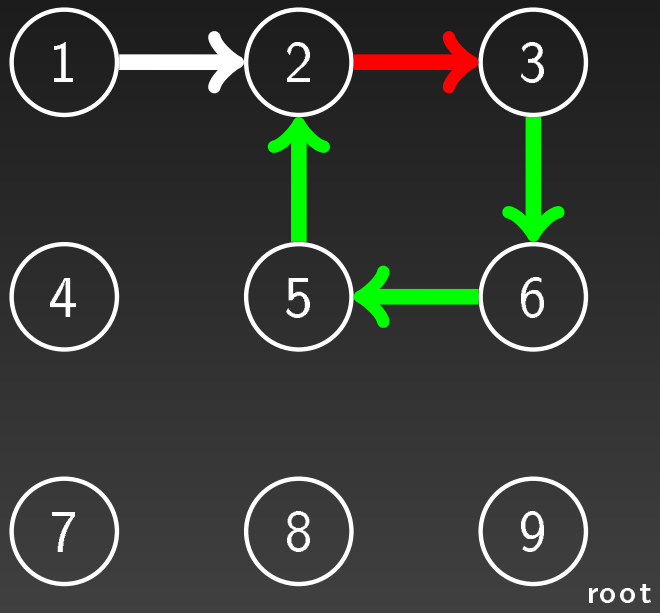
As recursion continues, might create a loop



CPRAR gives loop-erased random walk

As recursion continues, if reject, subcluster = loop

- For every node other than root, uniformly choose neighbor
- Repeat until there are no cycles



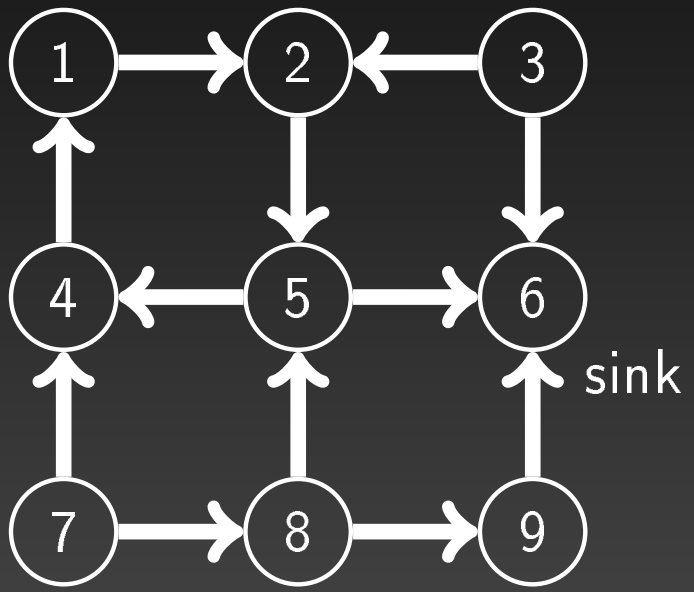
Accept $(5, 2)$, $(6, 5)$, and $(3, 6)$

Reject $(2, 3)$

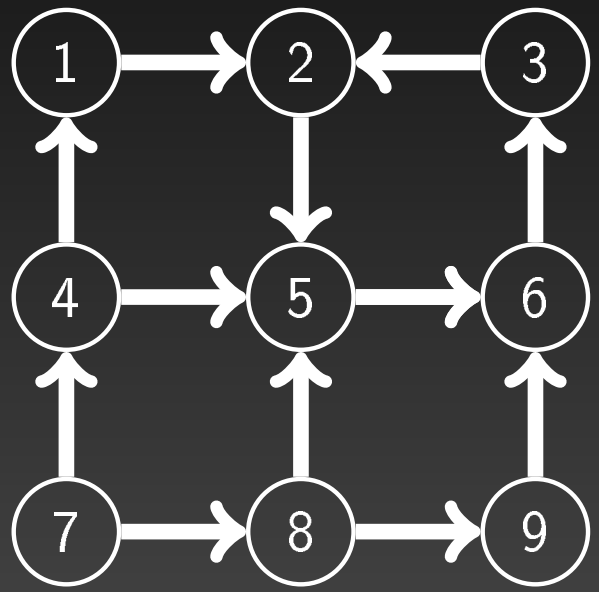
Begin again at node 2

Similar result for sink-free orientations of a graph

- Cohn, Pemantle, Propp $O(mn)$ expected running time
- CPRAR gives $\Theta(m)$ expected running time



Reject



Accept

Markov random fields

What is a Markov random field?

Definition

A *clique* is a subset of nodes such that each pair of nodes is connected by an edge. A *k-clique* is a clique of size k.

[Note: a node is a 1-clique, an edge is a 2-clique]

Fact

A distribution over a finite graph is a Markov random field (MRF) if the probability of a state is the product of factors over the cliques of the graph.

$$\phi(X) = \prod_{C \text{ a clique}} \phi_C(X(C))$$

[Note: the Ising model distribution factors into nodes and edges, and so is a MRF]

When can we ignore a clique?

Let v be a node, C be a clique with $v \in C$, $X(v)$ be a label for v

Then we always accept $X(v)$ as a label for v wrt clique C if:

$$U_C \leq \frac{\min_{y(V \setminus \{v\})} \phi_C((X(v), y(V \setminus \{v\})))}{\max_{y(V \setminus \{v\})} \phi_C((X(v), y(V \setminus \{v\})))} = p_C$$

Here $U_C \sim \text{Unif}([0, 1])$

Otherwise, resolve clique C recursively, and check if

$$U_C \leq \frac{\phi_C(X(v), x(V \setminus \{x\})}{\max_{y(V \setminus \{v\})} \phi_C((X(v), y(V \setminus \{v\})))} = p_C(x)$$

Cluster PRAR over an MRF

Cluster PRAR returns value of X on one or more nodes that contain v:

$\texttt{CPRAR}(v, V)$ *Output:* $T, X(T)$

Let $T \leftarrow \{v\}$, and draw $X(v)$ using $f_{\{v\}}$
For each clique C containing v, draw $U_C \leftarrow \text{Unif}([0,1])$
$\mathcal{C} \leftarrow \{C : U_C > p_C\}$, $S \leftarrow \cup_{C \in \mathcal{C}} C$
While $(\exists w \in S \setminus T)$
  Let w be any element of $S \setminus T$
  $(T', X(T')) \leftarrow \texttt{CPRAR}(w, V \setminus T)$
  $T \leftarrow T \cup T'$
  If there is a clique C with $U_C > p_C(x)$ then $T \leftarrow \emptyset$, $S \leftarrow \emptyset$
If $(T = \emptyset)$ then $(T, X(T)) \leftarrow \texttt{CPRAR}(v, V)$

Running time bound

Notation

For a node v, let $\mathcal{C}(v)$ be the set of cliques of size larger than 1 that contain v.

Theorem

For a node v, let U_C be iid Unif$([0,1])$ for all $C \in \mathcal{C}(v)$ and $X(v)$ be chosen according to the MRF restricted to v. Let

$$n_v = \mathbb{E}\left[\sum_{C \in \mathcal{C}(v): U_C > p_C} \#(C)\right]$$

If $n_v < 1$ for all v, then the expected number of steps taken by the algorithm is at most

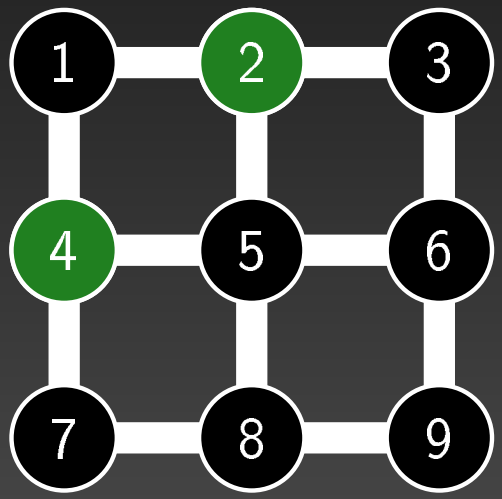
$$[1 + 1/(1 - \max_w n_w)] \max_w \#(\mathcal{C}(v))$$

Example: hard core gas model

Hard core gas model

Each node either one (green) or off (black)

- Each on node gives factor of λ to weight
- No two adjacent nodes can both be on



$$w(x) = \lambda^2$$

Hard core gas model in CPRAR

Suppose that we choose "off" for node 1

- Regardless of value for nodes 2 and 4, we always accept!

Now suppose that we choose "on" for node 1

- Then have to know value for both nodes 2 and 4 in order to advance

So

$$n_v \leq \frac{1}{1+\lambda} \cdot 0 + \frac{\lambda}{1+\lambda}(\Delta - 1)$$

So linear time when

$$\lambda < \frac{1}{\Delta - 2}$$

Hard core gas model better analysis

Add a 1 to tree of possibilities: add 1 to ϕ

Add a 0 to tree of possibilities: subtract $1/(\Delta - 2)$ from ϕ After one step moving from ϕ to ϕ', if no chance of retreating one level of recursion:

$$\mathbb{E}[\phi'|\phi] \leq \phi + \frac{\lambda}{1+\lambda} - \frac{1}{\Delta - 2}\frac{1}{1+\lambda}$$

After one step moving from ϕ to ϕ', if chance of retreating one level of recursion:

$$\mathbb{E}[\phi'|\phi] \leq \phi + \frac{\lambda}{1+\lambda} - \left[2 - 2(\Delta - 2)\frac{1}{\Delta - 2}\right]\frac{1}{1+\lambda}$$

Example: autonormal model

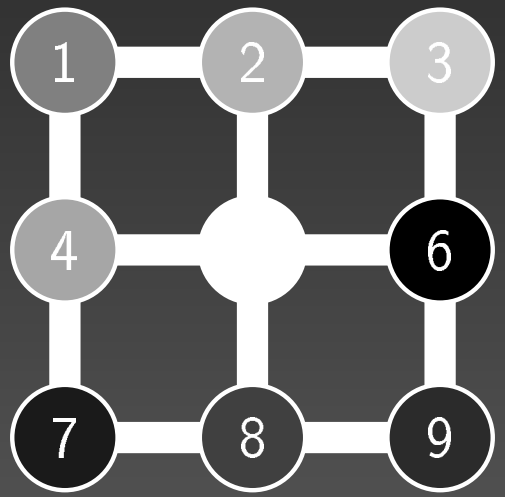
Autonormal model

On the statistical analysis of dirty pictures
J. Besag
JRSS B, *48(3):259–302, 1986*

Generalizes Ising model

Each node labeled with a number in $[0, 1]$

Weight given by edge $\{i, j\}$ is $\exp(-\beta(x(i) - x(j))^2)$



Autonormal model

Good news!

- Despite being a continuous model, CPRAR works the same way as for Ising model
- Minimum chance of accepting edge still $\exp(-\beta)$
- Still runs in linear time for

$$\exp(-\beta) \leq \frac{1}{\Delta - 1}$$

The infinite lattice

What about finite window on infinite lattice?

Theory relies on

- Existance of densities for single node and rest of sample
- As long as a unique density exists for rest of sample, can use CPRAR to obtain part of a sample

Conclusion

- When interactions between dimensions are weak and local (Markov random field) then can sample a single dimension in constant expected time
- Can be used to generate entire sample in linear time
- Gives popping algorithms automatically
- Can also use to get piece of an infinite dimensional problem
- Works equally well on continuous and discrete spaces
- Comparable to CFTP, but without drawbacks such as memory storage, extra work for continuous spaces