PERFECT Simulation

Lecture 2

Perfect simulation II

Coupling from the Past and the Fundamental Theorem of Perfect Simulation

Mark Huber
Claremont McKenna College

July, 2018

Supported by NSF grant DMS 1418495

An abbreviated Monte Carlo timeline

- 1951: Acceptance-rejection (Von Neumann)
- 1953: Metropolis-Rosenbluth-Rosenbluth-Teller-Teller (MR^{2}T^2)
- 1970: Hastings (new name Metropolis-Hastings)
- 1992: Adaptive Rejection Sampling (Gilks and Wild)
- 1996: Coupling from the Past (Propp and Wilson)
- 1998: Dominated coupling from the past (Kendall)
- 1998: Fill's method (Fill)
- 1999: Bounding chains for CFTP (H.)

An abbreviated Monte Carlo timeline

- 2000: Read-once CFTP (Wilson)
- 2000: FMMR (Fill, Machida, Murdoch, and Rosenthal)
- 2000: Randomness Recycler (Fill & H.)
- 2001: Catalytic Perfect Simulation (Breyer & Roberts)
- 2006: SDE's (Beskos, Papaspiliopoulos, & Roberts)
- 2006: Non-Markovian bounding chains (H.)
- 2009: Reverse time martingales (Łatuszyński, Kosmidos, Papaspiliopoulos, Roberts)
- 2014: Atomic regeneration for SMC (Lee, Doucet, Łatuszyński)
- 2016: Partially Recursive Acceptance Rejection (H. 2016)
- 2016: Bernoulli Factories (H. 2016)

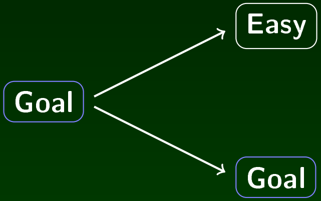
Choices made during AR

Acceptance rejection uses a branching approach using $A \subseteq B$

$$\begin{tikzcd}
 & \mathsf{Unif}(A) \\
\mathsf{Unif}(B) \arrow[ur] \arrow[dr] & \\
 & \mathsf{Unif}(B \setminus A)
\end{tikzcd}$$

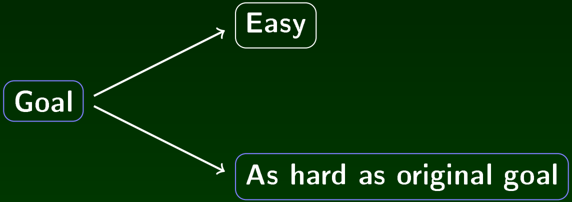
$$\mathsf{Unif}(B) = \mathsf{Unif}(A)(m(A)/m(B)) + \mathsf{Unif}(B \setminus A)(1 - m(A)/m(B))$$

Another view of AR

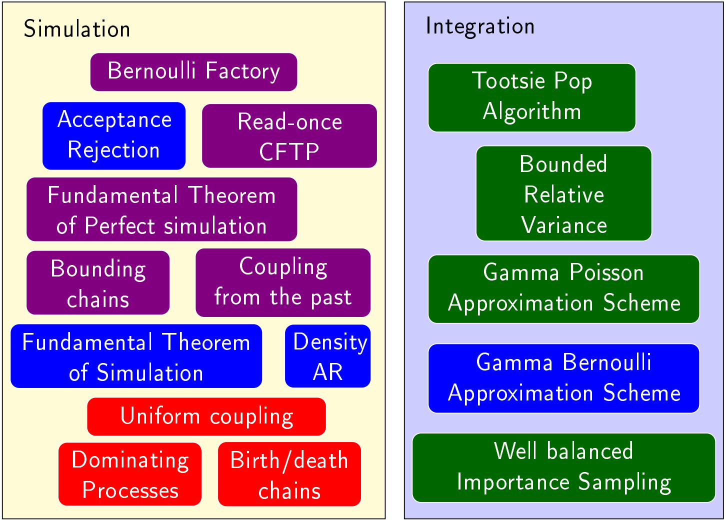


Either quit or start over with same goal

This lecture, coupling from the past



Goal dist is mixture of easy dist and another dist which computationally is as difficult as the original goal



Simulation

Bernoulli Factory

Acceptance Rejection

Read-once CFTP

Fundamental Theorem of Perfect simulation

Bounding chains

Coupling from the past

Fundamental Theorem of Simulation

Density AR

Uniform coupling

Dominating Processes

Birth/death chains

Integration

Tootsie Pop Algorithm

Bounded Relative Variance

Gamma Poisson Approximation Scheme

Gamma Bernoulli Approximation Scheme

Well balanced Importance Sampling

How to build a Perfect Simulator

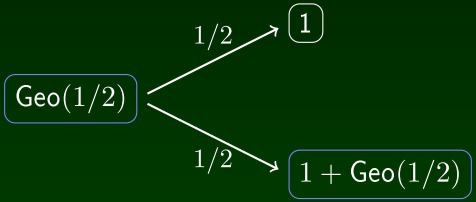
Markov chain approach

Move from state to state–under certain conditions distribution of state at least as close to stationary as we did before.

Perfect simulation approach

Randomly change the distribution that we seek to sample from until the distribution is easy.

Geometric with parameter $1/2$



$\mathsf{Geo}(1/2)$

$1/2$

1

$1/2$

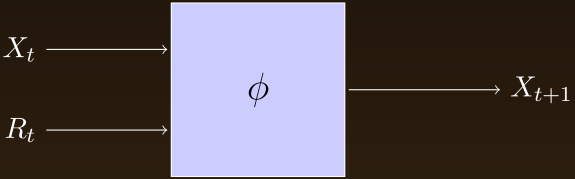
$1 + \mathsf{Geo}(1/2)$

Computers are at the end of the day machines

They take certain inputs, perform deterministic actions, which results in certain outputs

Update function

- All Markov chains can be simulated on a computer
- Computers only perform deterministic actions
- So there is a deterministic function ϕ



where X_t is the current state of the Markov chain and R_t is some randomness for the step

- Call this function an **update function**

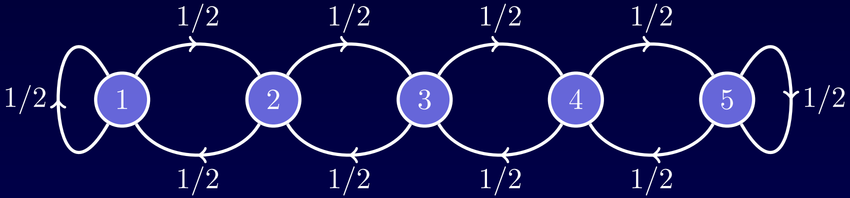
Example: update function for simple symmetric random walk with partially reflecting boundaries

- Start with a source of randomness

$$R_1, R_2, \ldots \overset{\text{iid}}{\sim} \mathsf{Unif}(\{-1, 1\})$$

- Add R_i to X_i to get X_{i+1} unless that would leave $\{1, \ldots, 5\}$, otherwise stay where you are
- Formally

$$\phi(X_t, R_t) = X_t + R_t \mathbb{1}(X_t + R_t \in \{1, \ldots, 5\})$$



Stationary update function

Definition

An update function ϕ is stationary with respect to distribution π if for $X \sim \pi$, $\phi(X, R) \sim \pi$.

Remarks

- Much of MCMC is devoted to finding stationary update functions!
- Example from last slide is stationary for $\pi \sim \text{Unif}(\{1, \ldots, 5\})$

A useful fact

Let

$$\begin{aligned}\phi_1(x_1, r_1) &= \phi(x_1, r_1)\\ \phi_2(x_1, r_1, r_2) &= \phi(\phi(x_1, r_1), r_2)\\ \phi_3(x_1, r_1, r_2, r_3) &= \phi(\phi(\phi(x_1, r_1), r_2), r_3)\\ \vdots &= \vdots\end{aligned}$$

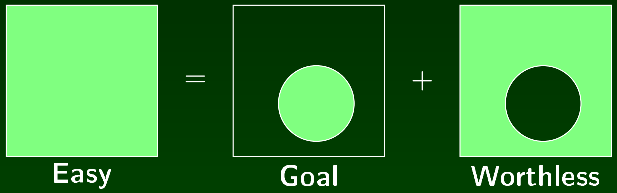
Fact

If ϕ is stationary for π, so is ϕ_n.

AR as a mixture process

Recall that to sample uniformly from A, if $R \sim \text{Unif}(B)$,

$$\text{Unif}(B) = \text{Unif}(A)\mathbb{1}(R \in A) + \text{Unif}(B \setminus A)\mathbb{1}(R \notin A)$$



For update functions

- Consider random choices over n steps

$$R = (R_1, \ldots, R_n)$$

- Then either R falls in some special set of random choices A, or it does not
- Therefore π is a mixture of these two options

Put mathematically

For $R = (R_1, \ldots, R_n)$, then for $X \sim \pi$,

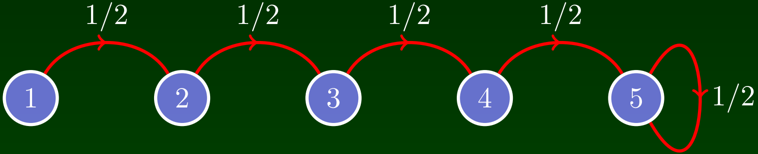
$$\begin{aligned} \pi &\sim \phi_n(X, R) \\ &\sim \phi_n(X, R)[\mathbb{1}(R \in A) + \mathbb{1}(R \notin A)] \\ &\sim \phi_n(X, R)\mathbb{1}(R \in A) + \phi_n(X, R)\mathbb{1}(R \notin A), \end{aligned}$$

and this holds for any A

What happens when $A = (1, 1, 1, 1)$

$A = \{r\}$, $r = (1, 1, 1, 1)$ means try to move to the right four times

$$\phi_4(1, r) = \phi_4(2, r) = \phi_4(3, r) = \phi_4(4, r) = \phi_4(5, r) = 5$$



No matter where we start, if $R \in A$, then $\phi_4(x, R) = \{5\}$

Using this to sample

Before we said π is a mixture:

$$\pi \sim \phi_n(X, R)\mathbb{1}(R \in A) + \phi_n(X, R)\mathbb{1}(R \notin A),$$

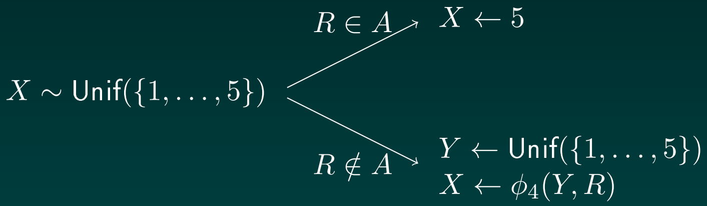
For $A = \{(1, 1, 1, 1)\}$,

$$\pi \sim \pi_{\{5\}}\mathbb{1}(R = (1, 1, 1, 1)) + \phi_4(X, R)\mathbb{1}(R \neq (1, 1, 1, 1)),$$

Just like in AR, when $R = (1, 1, 1, 1)$, the function ϕ_4 collapses the distribution down to an atomic measure

Picture of example

$$R \leftarrow \mathsf{Unif}(\{-1,1\}^4),\ A = \{(1,1,1,1)\}$$



$X \sim \mathsf{Unif}(\{1,\ldots,5\})$

$R \in A$

$X \leftarrow 5$

$R \notin A$

$Y \leftarrow \mathsf{Unif}(\{1,\ldots,5\})$

$X \leftarrow \phi_4(Y,R)$

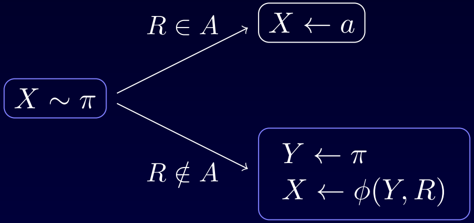
Picture of general CFTP

Ingredients

- ϕ such that with source of randomness R, ϕ is stationary for π
- So for $X_0 \sim \pi$, $\phi(X_0, R) \sim \pi$
- A such that it is easy to check if $r \in A$, and for all $x \in \Omega$ and $r \in A$, $\phi(x, r) = \{a\}$

$$\begin{tikzcd}
 & \boxed{X \sim [\phi(X_0, R) | R \in A]} \\
\boxed{X \sim \phi(X_0, R)} \arrow[ur, "R \in A"] \arrow[dr, "R \notin A"'] & \\
 & \boxed{X \sim [\phi(X_0, R | R \notin A])}
\end{tikzcd}$$

Implementing CFTP



$X \sim \pi$

$R \in A$

$X \leftarrow a$

$R \notin A$

$Y \leftarrow \pi$

$X \leftarrow \phi(Y, R)$

Pseudocode for AR and CFTP

```
AR
  1. Draw $X \leftarrow \text{Unif}(B)$
  2. If $X \notin A$, $X \leftarrow \texttt{AR}$
  3. Output $X$
```

```
CFTP
  1. Draw $R$ randomly
  2. If $R \in A$, set $X$ to be unique element of $\phi(\Omega, R)$
  3. Else draw $Y \leftarrow \texttt{CFTP}$, $X \leftarrow \phi(Y, R)$
  4. Output $X$
```

These algorithms are both recursive: they call themselves a random number of times that is unbounded

Both AR and CFTP work

Theorem (Propp & Wilson 1996)

As long as $\mathbb{P}(R \in A) > 0$, CFTP outputs X exactly from π in finite time.

Theorem

As long as $\mathbb{P}(R \in A) > 0$, AR outputs X exactly from π in finite time.

Making CTFP efficient

To run CFTP

Requirement

- Need set A such that $\#(\phi(\Omega, R)) = 1$ for all $R \in A$
- Want $\mathbb{P}(R \in A)$ large

Example: $\Omega = \{1, \dots, 5\}$

For simple symmetric random walk, A is four moves to the right

$$A = \{(1,1,1,1)\}, \; \mathbb{P}(R \in A) = (1/2)^4 = 1/16$$

When $\Omega = \{1, \dots, n\}$,

$$A = \{(1,1,1,\dots,1)\}, \; \mathbb{P}(R \in A) = (1/2)^{n-1}$$

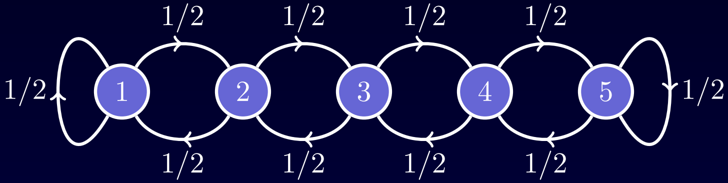
Need to do better!

Monotonicity

Coupling preserves order

1. Start with pair of states $x \leq y$
2. Either try to move both to the right, or both to the left
3. After move to x' and y', still have $x' \leq y'$

Monotonicity



Examples

- $x = 1$, $y = 3$, move equals $+1$, $x' = 2$, $y' = 4$
- $x = 4$, $y = 5$, move equals $+1$, $x' = 5$, $y' = 5$
- $x = 2$, $y = 2$, move equals -1, $x' = 1$, $y' = 1$

Mathematical formulation

Update function, $R_t \overset{\text{iid}}{\sim} \text{Unif}(\{-1,1\})$

$$X_{t+1} = \begin{cases} X_t + R_t & X_t + R_t \in \Omega \\ X_t & X_t + R_t \notin \Omega \end{cases}$$

Note that for all $r \in \{-1,1\}$

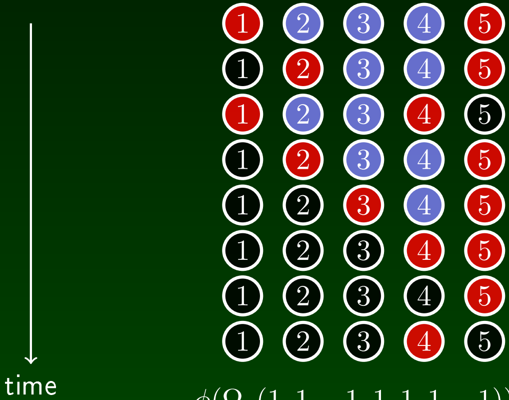
$$x \leq y \Rightarrow \phi(x,r) \leq \phi(y,r)$$

Minimum and Maximum state

If there is a minimum and maximum state

- Run chain forward from min and max state using the same random choices for each one
- If they meet, all of the state space has been squeezed between them, and $\phi(\Omega, R) = \{a\}$

Example: random walk on $\{1, \ldots, 5\}$



$R_1 = 1$

$R_2 = 1$

$R_3 = -1$

$R_4 = 1$

$R_5 = 1$

$R_6 = 1$

$R_7 = 1$

$R_8 = -1$

$$\phi(\Omega, (1, 1, -1, 1, 1, 1, -1)) = \{4\}$$

Monotonicity not true of all, but some chains

Monotonic update functions can be used for

- Ferromagnetic Ising model (Propp Wilson 1996)
- Autonormal model (Wilson 2000, Gibbs 2004)
- Slice samplers (Mira, Møller, Roberts 2001)
- Permutations for linear extensions (Caracciolo et al 2009, H. 2014)

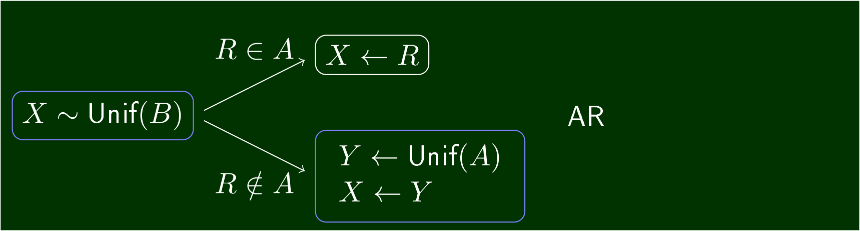
Pros and cons

- Finding a partial order on state space is easy
- Finding a monotonic update can be difficult

The Fundamental Theorem of Perfect Simulation

Generalizing AR and CFTP

So far we have two methods:



$X \sim \mathsf{Unif}(B)$

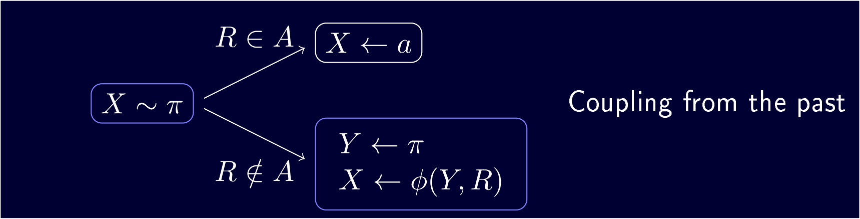
$R \in A$

$X \leftarrow R$

$R \notin A$

$Y \leftarrow \mathsf{Unif}(A)$
$X \leftarrow Y$

AR



$X \sim \pi$

$R \in A$

$X \leftarrow a$

$R \notin A$

$Y \leftarrow \pi$
$X \leftarrow \phi(Y, R)$

Coupling from the past

Generalizing AR and CFTP

These algorithms have two good properties

1. They are locally correct: if you assume that subsequent recursive calls return the correct distribution, then you could quickly prove the algorithm correct.
2. They terminate with probability 1.

Here's the good part: These two properites are sufficient for a perfect simulation algorithm to work!

Inuitive form, Fundamental Theorem of Perfect Simulation

Suppose I have an algorithm that terminates with probability 1, and if I assume the recursive calls have the corrrect distribution, then I can prove the algorithm is correct. Then the overall algorithm is correct.

Proof CFTP works

```
CFTP
  1. Draw $R$ randomly
  2. If $R \in A$, set $X$ to be unique element of $\phi(\Omega, R)$
  3. Else draw $Y \leftarrow \texttt{CFTP}$, $X \leftarrow \phi(Y, R)$
  4. Output $X$
```

Proof CFTP has output from π.

By the FTPS, in line 3, assume $Y \leftarrow \texttt{CFTP}$ gives $Y \sim \pi$. Then

$$X = \phi(Y,R)\mathbb{1}(R \in A) + \phi(Y,R)\mathbb{1}(R \notin A) = \phi(Y,R) \sim \pi.$$

□

Proof AR works

```
AR
  1. Draw $R \leftarrow \text{Unif}(B)$
  2. If $R \in A$, then $X \leftarrow R$
  3. If $X \notin A$, $Y \leftarrow \texttt{AR}$, $X \leftarrow Y$
  4. Output $X$
```

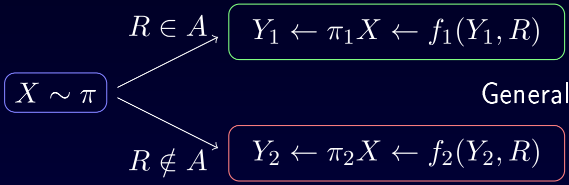
Proof AR has output from Unif(A).

By the FTPS, in line 2 assume $Y \sim \text{Unif}(A)$. For $C \subseteq A$:

$$
\begin{aligned}
\mathbb{P}(X \in C) &= \mathbb{P}(R \in C) + \mathbb{P}(R \notin A)\mathbb{P}(Y \in C) \\
&= \frac{m(C)}{m(B)} + \left(1 - \frac{m(A)}{m(B)}\right)\frac{m(C)}{m(A)} \\
&= \frac{m(C)}{m(A)}\left(\frac{m(A)}{m(B)} + 1 - \frac{m(A)}{m(B)}\right) = \frac{m(C)}{m(A)}.
\end{aligned}
$$

□

General Perfect Simulation



$X \sim \pi$

$R \in A$

$Y_1 \leftarrow \pi_1 X \leftarrow f_1(Y_1, R)$

$R \notin A$

$Y_2 \leftarrow \pi_2 X \leftarrow f_2(Y_2, R)$

General Perfect Simulation

- At each step, take one of two paths
- Say that a path terminates if π_i easy to sample from so no further branching is necessary

Fundamental Theorem of Perfect Simulation

Theorem

Suppose that at each step of the perfect simulation, given random choices R, $Y_1 \sim \pi_1$, $Y_2 \sim \pi_2$

$$X \sim f_1(Y_1, R)\mathbb{1}(R \in A) + f_2(Y_2, R)\mathbb{1}(R \notin A).$$

If the algorithm terminates with probability 1, then the result is $X \sim \pi$.

Some notes

- Presented for 2 choices, but works for any finite number
- AR: (π_1 can be anything)

$$f_1(Y_1, R) = R,\ \ \pi_2 = \mathsf{Unif}(A),\ \ f_2(Y_2, R) = Y_2$$

- CFTP: (π_1 can be anything)

$$f_1(Y_1, R) = \phi(\Omega, R),\ \ \pi_2 = \pi,\ \ f_2(Y_2, R) = \phi(Y_2, R)$$

Outline of proof of FTPS

- Suppose I limit the number of steps I take to N, at which point I always return $\perp \notin \Omega$. Call the output of this time constrained algorithm X_N. Call the output of the time unconstrained algorithm X. Then

$$\mathbb{P}(X_N \in A) \leq \mathbb{P}(X \in A) \leq \mathbb{P}(X_N \in A) + \mathbb{P}(X_N = \perp).$$

- By local correctness of branches, for any set A can show by induction that

$$\mathbb{P}(X_N \in A) \leq \pi(A) \leq \mathbb{P}(X_N \in A) + \mathbb{P}(X_N = \perp)$$

- Assuming the algorithm terminates with probability 1, $\lim_{N \to \infty} \mathbb{P}(X_N = \perp) = 0$.

Other perfect sampling protocols that fit this framework:

- Adaptive Acceptance Rejection
- Popping
- Randomness recycler
- Partially recursive acceptance rejection
- Fill, Machida, Murdoch, and Rosenthal (FMMR)

Bounding chains

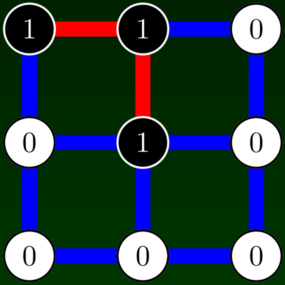
What to do when update function not monotonic

One solution is bounding chains

- Configuration $(x_1, x_2, \ldots, x_n)$
- Bounding chain for each i has bounding set A_i
- Example: For Ising model start with $A_i = \{-1, 1\}$
- MCMC often only updates one or two x_i at a time
- If x_i is updated, at same time update the bound A_i

$$A_i = \cup_{x:(\forall j)(x_j \in A_i)} \{\phi(x, R)_i\}$$

Example: Strauss process on finite graph

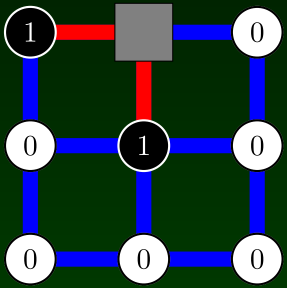


Density γ^2

- Nodes either occupied (labeled 1) or unoccupied (labeled 0)
- Parameter $\gamma \in [0, 1]$, density proportional to

$$\gamma^{\#\{\{i,j\}:x(i)x(j)=1\}}$$

Use reversible update

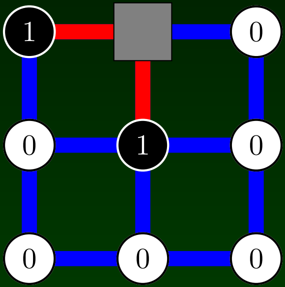


Accept 1 with prob γ^2

Accept 0 with prob 1

- Choose a node uniformly to update
- Propose new node label Bern$(\lambda/(1+\lambda))$
- Always accept a 0, accept a 1 with probability $\gamma^{\#\{\text{neighboring 1's}\}}$

Pseudocode for update



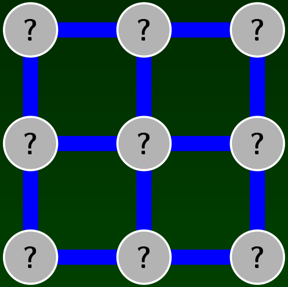
Accept 1 with prob γ^2

Accept 0 with prob 1

- Choose a node uniformly to update
- Propose new node label Bern$(\lambda/(1+\lambda))$
- Always accept a 0, accept a 1 with probability $\gamma^{\#\{\text{neighboring 1's}\}}$

Bounding chain keeps track of "unknown" nodes

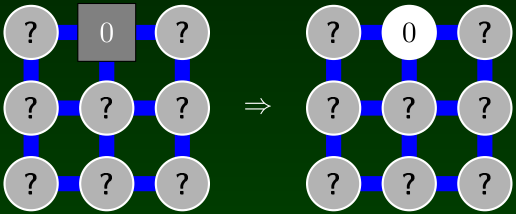
- Add new symbol $? = \{0, 1\}$ representing node value unknown
- Initially, start with all nodes labeled $?$



Then update the bounding chain as you take steps

Example of bounding chain update

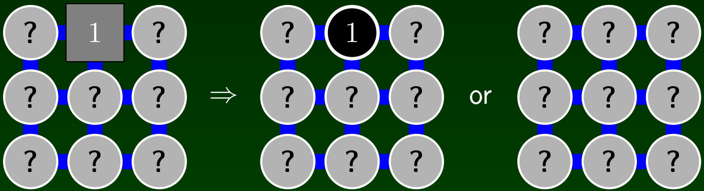
- Suppose we try to label a node 0, then always accepted



Example of bounding chain update 2

- Suppose we try to label a node 1, chance of acceptance in

$$\gamma^0, \gamma^1, \gamma^2, \gamma^3$$



$U \in [0, \gamma^3]$

$U \in [\gamma^3, 1]$

- Only accept if $U \sim \text{Unif}([0,1])$ has $U \leq \gamma^3$

Example of bounding chain update 3

- Once all the ? nodes are gone, $\phi(\Omega, R) = \{x\}$
- So bounding chains allow use of CFTP
- Let $Q_t = \#\{i : x(i) =?\}$, then can be shown that

$$\mathbb{E}[Q_{t+1}|Q_t] \leq Q_t \left(1 - \frac{\Delta\lambda(1-\gamma)/(1+\lambda)}{n}\right),$$

where n is # of nodes; Δ is the max degree of the graph, for

$$\lambda \leq \frac{1}{\Delta(1-\gamma)\lambda/(1+\lambda) - 1}$$

Running time for bounding chain procedure

- Hence

$$\mathbb{E}[Q_{t+1}] \leq n \left(1 - \frac{\Delta\lambda(1-\gamma)/(1+\lambda)}{n}\right)^t,$$

- For

$$\lambda \leq \frac{1}{\Delta(1-\gamma)\lambda/(1+\lambda) - 1}$$

  expected number of steps needed by CFTP is

$$\Theta(n \ln(n))$$

Read once CFTP

Notation

S	When $R \in A$
F	When $R \notin A$

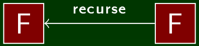
A typical run

F

Notation

S	When $R \in A$
F	When $R \notin A$

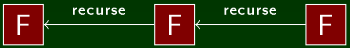
A typical run



Notation

S	When $R \in A$
F	When $R \notin A$

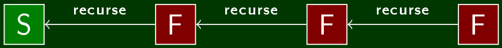
A typical run



Notation

S	When $R \in A$
F	When $R \notin A$

A typical run

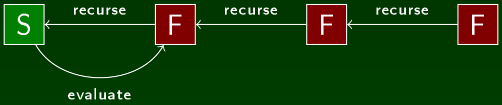


S ← recurse — F ← recurse — F ← recurse — F

Notation

S	When $R \in A$
F	When $R \notin A$

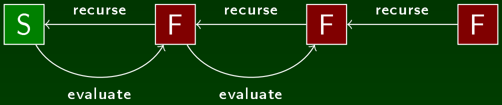
A typical run



Notation

S	When $R \in A$
F	When $R \notin A$

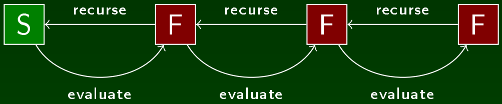
A typical run



Notation

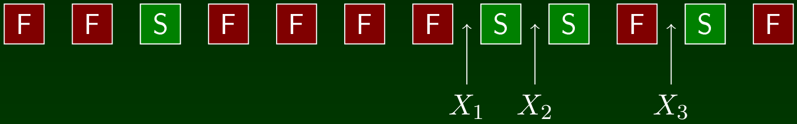
S	When $R \in A$
F	When $R \notin A$

A typical run



An observation

There are a geometric number of recursions, with mean equal to the multiplicative inverse of the probability of a success block $R \in A$



F F S F F F F S S F S F

X_1 X_2 X_3

Every sample before the second and later success blocks comes from π

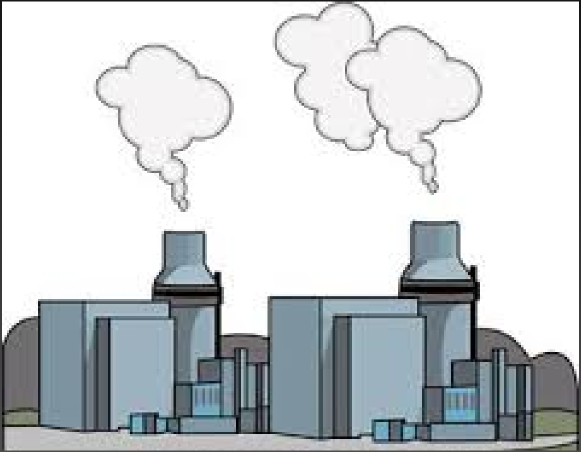
Read once CFTP[1]

Generating $X_1, \ldots, X_n \pi$ iid

```
1. Repeat
2.     Draw $R$
3. Until $R \in A$
4. Let $k \leftarrow 0$, $X_k \leftarrow$ single element of $\phi(\Omega, R)$
5. Repeat
6.     Draw $R$, $k \leftarrow k + \mathbb{1}(R \in A)$
7.     $X_k \leftarrow \phi(X_k, R)$
8. Until $k = n + 1$
9. Output $X_1, \ldots, X_n$
```

[1] D. B. Wilson, How to couple from the past using a read-once source of randomness, *Random Structures Algorithms*, 16(1):85–113, 2000

Bernoulli Factories



What is a Bernoulli factory?

Given an iid stream

$$B_1, B_2, \ldots$$

of random variables of mean p, create a single Bernoulli with mean

$$f(p)$$

using as few draws from the stream as possible

An example

Let

$$f(p) = p(1-p)$$

then

$$W = B_1(1-B_2)$$

is a Bernoulli random variable with mean $p(1-p)$

Von Neumann's Bernoulli Factory: Stopping time

We are trying to flip a fair coin, so let

$$f(p) = 1/2$$

then let

$$T = \inf\{t : t \text{ is even and } B_{2t} \neq B_{2t-1}\}$$

Set $W = B_T$

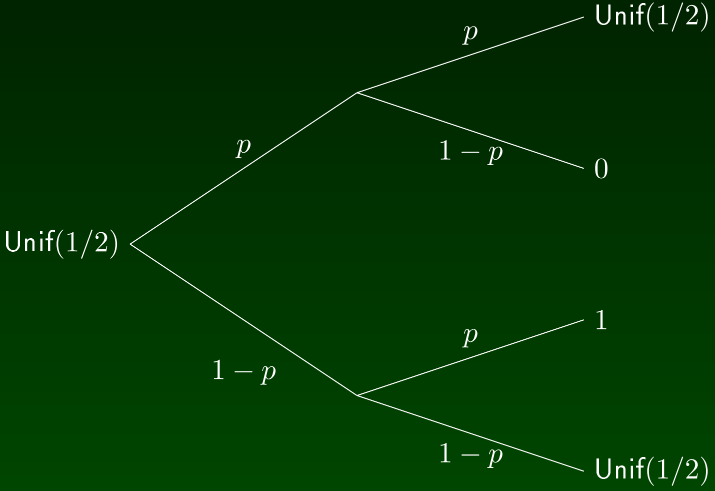
Von Neumann’s BF: Repeat form

```
Repeat
    Draw $A, B \leftarrow \text{Bern}(p)$
Until $A \neq B$
Ouptut $B$
```

Von Neumann's BF: Recursion form

```
VNBF
  1. Draw $A, B \leftarrow \mathsf{Bern}(p)$
  2. If $A = B$ then draw $B \leftarrow$ VNBF
  3. Ouptut $B$
```

Von Neumann's BF: Picture form



BF History: origins

S. Asmussen, P. W. Glynn, and H. Thorisson, Stationarity Detection in the Initial Transient Problem, *ACM Trans. Modeling and Computer Simulation*, 2(2):130–157, 1992.

- Simulation from stationary distribution of regenerative Markov processes
- Required as subroutine ability to generate from Bernoulli factory with $f(p) = Cp$ for constant C

BF History: next steps

M. S. Keane and G. L. O'Brien, A Bernoulli factory, *ACM Trans. Modeling and Computer Simulation*, 4:213–219, 1994.

- Introduced term Bernoulli factory
- Gave necessary and sufficient conditions on f for a Bernoulli factory to exist
- Mathematical construct rather than algorithm.
- Unknown if expected run time finite or tails heavy or light

BF History: Bernstein connection

S. Nacu and Y. Peres, Fast simulation of new coins from old, *Ann. Appl. Probab.*, 15(1A):93–115, 2005.

- Gave method with exponential tails (so unknown if expected run time finite)
- Used Bernstein polynomials to approximate $f(p)$:

$$\sum_{i=0}^{n} a_i p^i (1-p)^i \leq f(p) \leq \sum_{i=0}^{n} b_i p^i (1-p)^i$$

- Algorithm, but required exponential time to implement
- Showed $f(p) = 2p$ sufficient to get any real analytic f

BF History: first practical algorithm

K. Łatuszyński, I. Kosmidis, O. Papaspiliopoulos, and G. O. Roberts. Simulation events of unknown probability via reverse time Martingales, *Random Structures Algorithms*, 38:441–452, 2011.

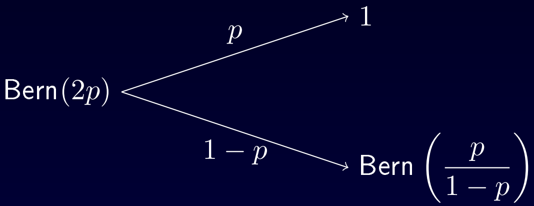
- Practical implementation of Nacu & Peres
- Introduced reverse time Martingales technique for perfect simulation
- Numerical experiments indicated run time not linear in C

BF History: FTPS view

M. Huber, Nearly optimal Bernoulli factories for linear functions, *Combinatorics, Probability and Computing*, arXiv: 1308.1562. 25(4):577–591, 2016.

- Used a move from distribution to distribution approach
- FTPS used to prove correctness
- Runs in time proportional to optimal # of steps needed

Illustrate for $f(p) = 2p$



Bern$(2p)$

p

1

$1-p$

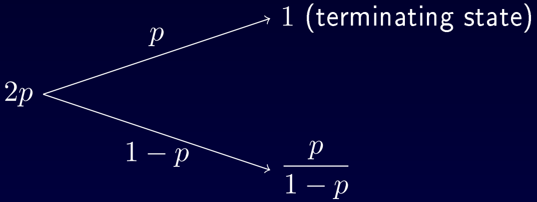
Bern $\left(\frac{p}{1-p}\right)$

Works because (for $X \sim$ Bern$(2p)$,

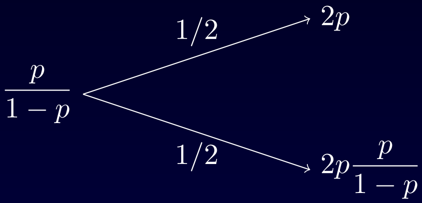
$$\mathbb{P}(X=1) = 2p = (p)(1) + (1-p)\left(\frac{p}{1-p}\right)$$

Shorthand

Since the only distributions we are interested here are Bernoulli, which are determined by their parameter, shorthand to write:



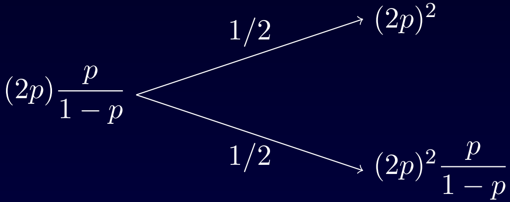
What to do with $p/(1-p)$



Here

$$\frac{p}{1-p} = \frac{1}{2} \cdot 2p + \frac{1}{2}(2p)\frac{p}{1-p}$$

What to do with $(2p)p/(1-p)$



$(2p)\dfrac{p}{1-p}$

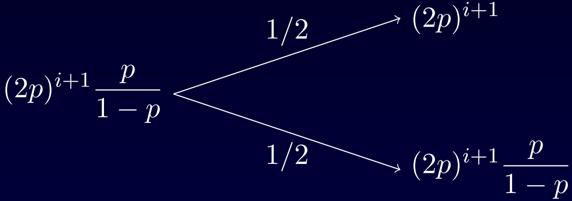
$1/2$

$(2p)^2$

$1/2$

$(2p)^2\dfrac{p}{1-p}$

What to do with $(2p)^i p/(1-p)$



$(2p)^{i+1}\dfrac{p}{1-p}$

$1/2$

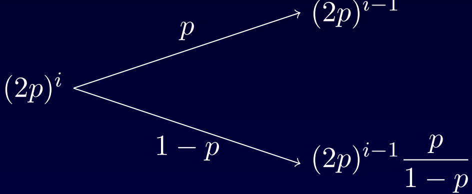
$(2p)^{i+1}$

$1/2$

$(2p)^{i+1}\dfrac{p}{1-p}$

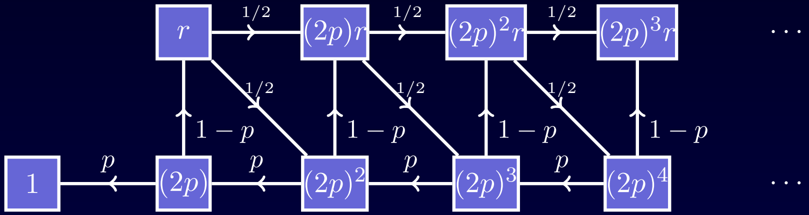
Now back to $(2p)^i$

When goal is of form $(2p)^i$, flip a p-coin once



Created Markov chain on distributions

Let $r = p/(1-p)$



- If we get to distribution Bernoulli with mean 1, terminate
- So far, we do not have any way of terminating at 0

To deal with $(2p)^i$ *where* i *is large*

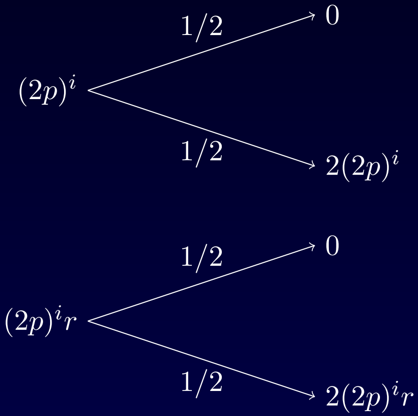
Assume $2p < 1 - \epsilon$ for known $\epsilon > 0$

$$(2p)^i \leq (1-\epsilon)^i < \exp(-i\epsilon)$$

For $i > 1/\epsilon$, $\exp(-1) < 1/2$, which means

$$2(2p)^i < 1$$

What to do with $(2p)^i$



$(2p)^i$

$1/2$

0

$1/2$

$2(2p)^i$

$(2p)^i r$

$1/2$

0

$1/2$

$2(2p)^i r$

How to deal with $2(2p)^i$

Bring 2 inside the exponent,

$$2(2p)^i = (2 \cdot 2^{1/i})^i,$$

For i large

$$2^{1/i} \approx 1 + \ln(2)/i,$$

so does not increase constant much

Running time

Theorem

The average number of steps (with appropriate tuning of constants) for a Bernoulli Factory with $f(p) = Cp \leq 1 - \epsilon$ is $9.5C\epsilon^{-1}$.

Summary

- Coupling from the past is second most widely used perfect simulation technique after AR
- Takes advantage of existing update functions for Markov chains
- FTPS gives condition for perfect simulations to work similar to stationary updates for MCMC to work
- Idea applicable to situations such as Bernoulli Factories
- Goal: randomly transform your problem/distribution to an easier one