Perfect Sampling Algorithms: Connections

Duncan Murdoch
University of Western Ontario
November 15, 2002

Outline

1. Background: Markov chain Monte Carlo (MCMC)
2. Coupling from the past (CFTP)
3. Read-once CFTP
4. Fill's Rejection Sampler
5. Variations

This talk is based on joint work with Jeff Rosenthal, Jim Fill, Motoya Machida, and Xiaoqiang Li.

Background

- Want to study the distribution of $X \sim \pi(\cdot)$, $X \in \mathcal{X} \subset \mathcal{R}^d$.
- Example: $\pi(\cdot)$ is the posterior distribution in a Bayesian model, want to know $E(f(X)) = \int f(x)\pi(x)dx$.
- Problem: often $\pi(\cdot)$ is intractable.
- Partial solution: use Monte Carlo simulation to generate $X_1, \ldots, X_n$ from $\pi(\cdot)$, estimate $E(f(X))$ by $(1/n)\sum f(X_i)$.

MCMC continued...

- Problem: often we don't know how to sample from $\pi(\cdot)$.
- Partial solution: sample from a Markov Chain, whose steady-state distribution is $\pi(\cdot)$. Run "until convergence", treat sampled values as samples from $\pi(\cdot)$.

Common MCMC Algorithms

1. The Gibbs Sampler: Split X into components $X = (X^{(1)}, \ldots, X^{(p)})$. Draw in sequence each $X^{(j)}$ from the conditional distribution given the other components.

2. General Metropolis-Hastings: Draw Y from a "proposal distribution" $q(\cdot|X_t)$.

$$X_{t+1} = \begin{cases} Y & \text{if Unif}(0,1) < \alpha(Y|X_t) \\ X_t & \text{otherwise} \end{cases}$$

where

$$\alpha(Y|X) = \frac{\pi(Y)}{q(Y|X)} \Big/ \frac{\pi(X)}{q(X|Y)}$$

Common MCMC Algorithms cont'd

3. The Independence Sampler:
   Metropolis-Hastings with

$$\begin{aligned} q(y|x) &= q(y) \\ \alpha(Y|X_t) &= \frac{\pi(Y)}{q(Y)} \Big/ \frac{\pi(X_t)}{q(X_t)} \end{aligned}$$

4. Random-Walk Metropolis:
   Metropolis-Hastings with

$$\begin{aligned} q(y|x) &= q(|y-x|) \\ \alpha(Y|X_t) &= \pi(Y)/\,\pi(X_t) \end{aligned}$$

MCMC continued...

- Problem: The distribution of the X_t values converges to $\pi(\cdot)$, but how quickly?
- Solution: Use perfect sampling!

History

- Coupling from the past (CFTP):
  Propp, J.G. and Wilson, D.B. (1996).
  Random Structures and Algorithms
  9, 223-252.

- Fill's algorithm:
  Fill, J.A. (1998). *Ann. App. Prob.* **8**,
  131-162.

  Fill, J.A., Machida, M., Murdoch,
  D.J. and Rosenthal, J.S. (2000).
  Random Structures and Algorithms
  17, 290-316.

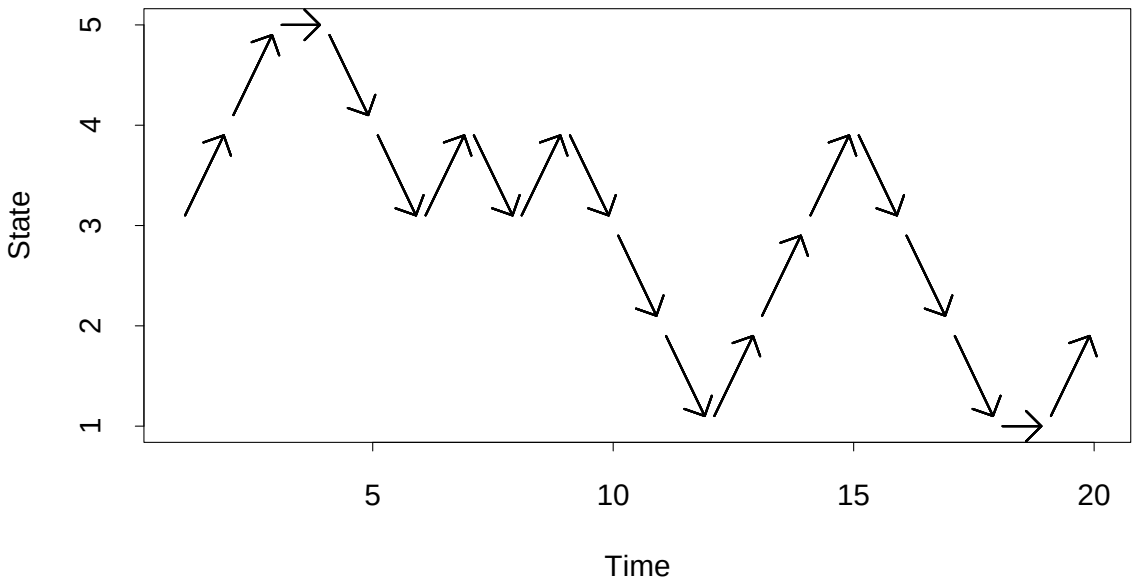
- Numerous variations on both since.

Coupling from the Past

- CFTP samples from $\pi(\cdot)$.
- Idea: Compute result of infinitely long run from the past by coupling all possible tails of shorter runs.
- If they all give the same answer, it must be in steady-state!

Example: Random Walk

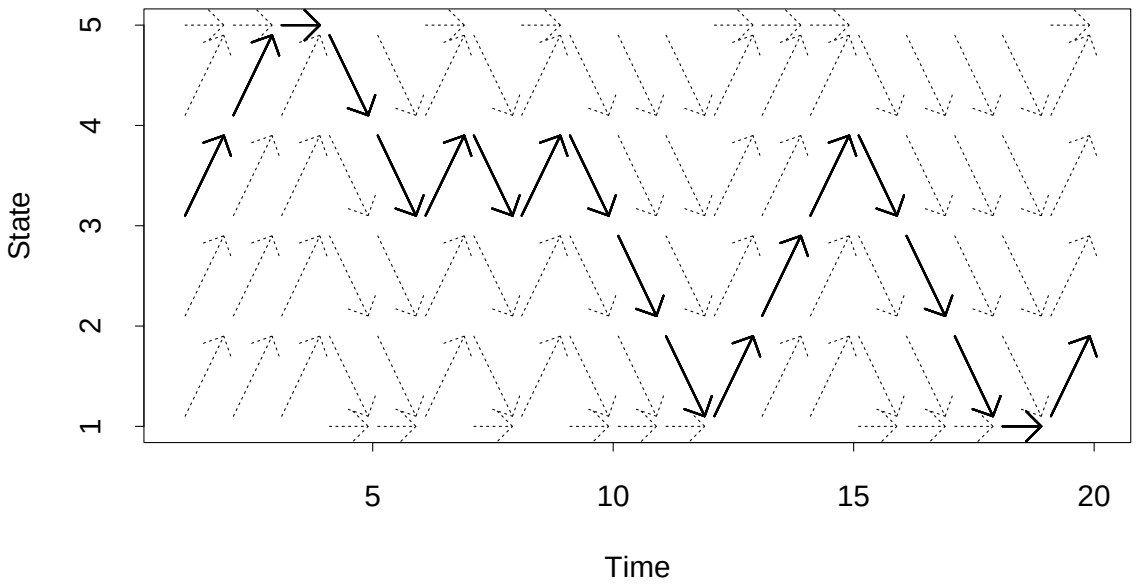
$$X_{t+1} = \begin{cases} \min(X_t + 1, 5) & p = 1/2 \\ \max(X_t - 1, 1) & p = 1/2 \end{cases}$$



Coupling

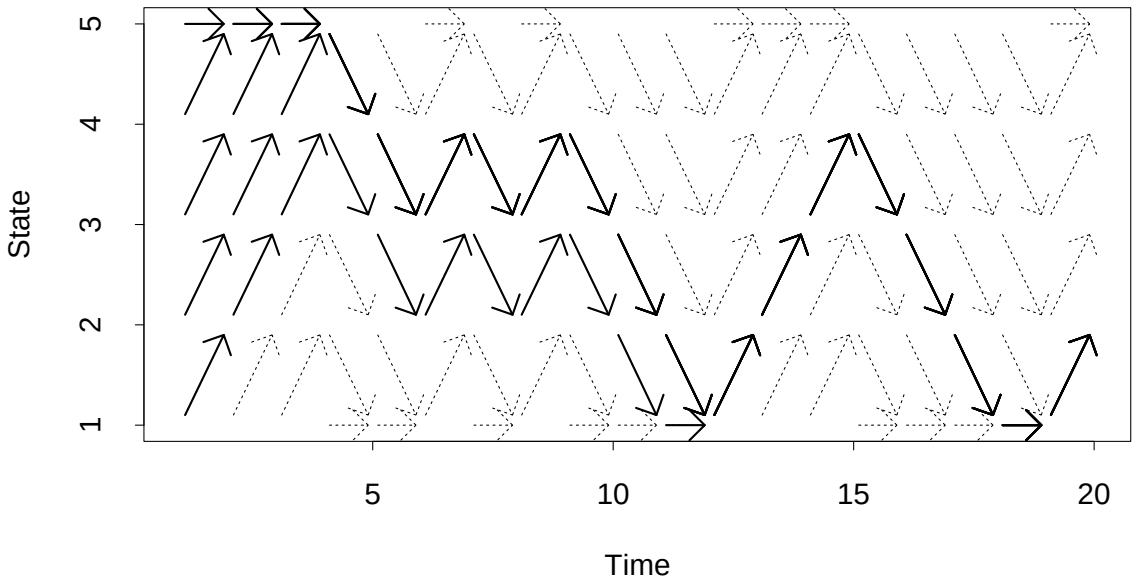
Write $X_{t+1} = \phi(X_t, U_{t+1})$ where U_t are independent with a *known* distribution and $\phi(\cdot, \cdot)$ is deterministic.

$$\begin{aligned} X_{t+1} &= \min[\max(X_t + U_{t+1}, x_{\min}), x_{\max}] \\ U_{t+1} &= \pm 1 \text{ (with equal probability)} \end{aligned}$$



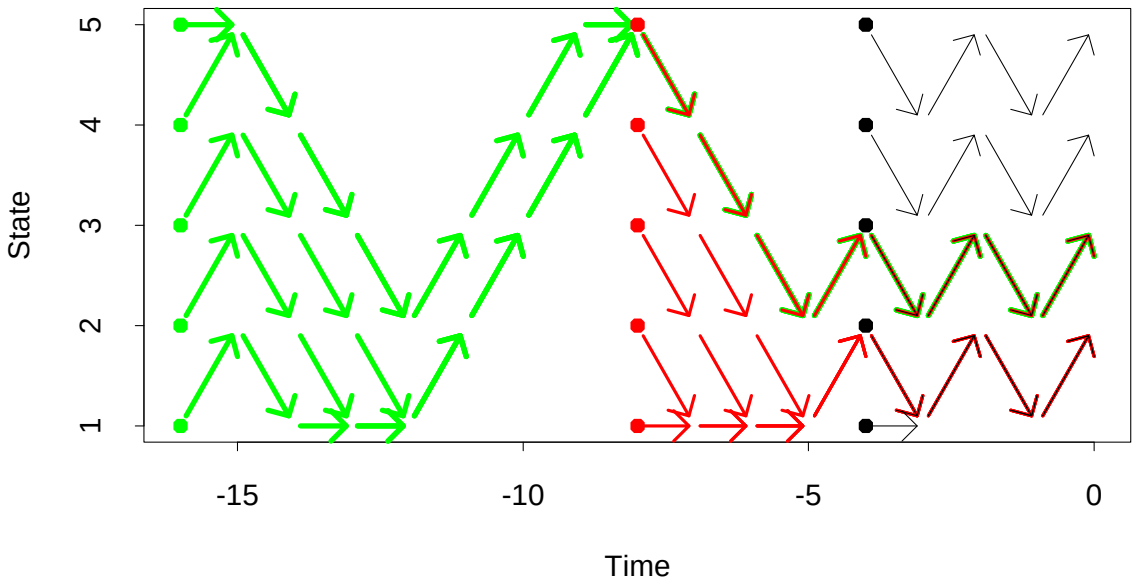
Coupling continued...

With many update functions, paths *coalesce*: regardless of the initial state, the value of X_t is the same for large enough t. The past is forgotten; no initialization bias remains.



Coupling from the Past (CFTP)

Removing the initialization bias is not enough: there may be a coupling time bias. Compute the result of an infinitely long run from the past by coupling all possible tails of shorter runs.



The CFTP Algorithm

CFTP(M):

$t \leftarrow -M$	*Try from* $-M$
$B_t \leftarrow \mathcal{X}$	*All states possible*
while $t < 0$	*Run to* $t = 0$
$\quad t \leftarrow t + 1$	*Update* t
$\quad B_t \leftarrow \phi(B_{t-1}, U_t)$	*Update* B_t
if $\#B_0 = 1$ then	*Coalesced?*
$\quad$ return(B_0)	*Yes, we're done!*
else	
$\quad$ CFTP($2M$)	*No, try from* $-2M$

Proof of Validity of CFTP

The CFTP algorithm generates X_0 as a function of the infinite sequence $U_0, U_{-1}, U_{-2}, \ldots$:

$$X_0 = C(U_0, U_{-1}, U_{-2}, \ldots) \tag{1}$$

We can generate X_1 in two ways:

$$\begin{aligned} X_1 &= \phi(X_0, U_1) && (2) \\ &= C(U_1, U_0, U_{-1}, \ldots) && (3) \end{aligned}$$

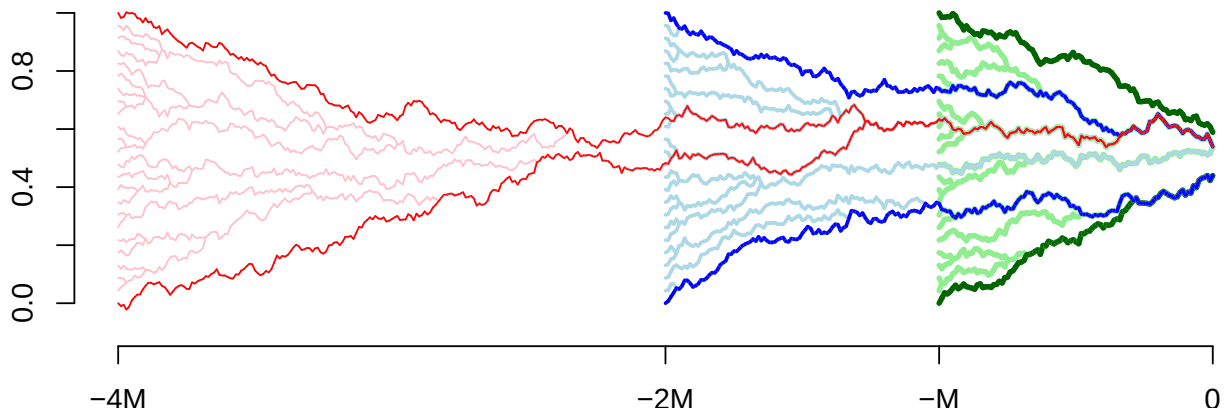
By (1) and (3) and the invariance of the distribution of an i.i.d. sequence under a shift, X_0 and X_1 have the same distribution. By (2), the distribution of X_1 is the update of the distribution of X_0 under the Markov chain transition kernel.

The only distribution left invariant is the stationary distribution.

Implementation problem

When M is increased to $2M$, CFTP needs
$U_{-2M+1}, \ldots, U_0$, but $U_{-M+1}, \ldots, U_0$ must not change.

This requires either lots of storage, or tricky programming.

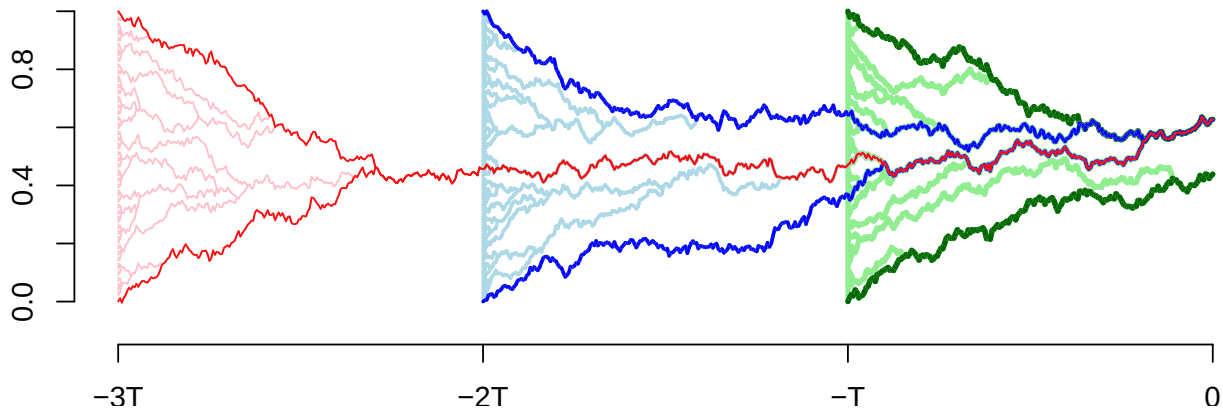


This figure and later ones use Wilson's (2000) multishift coupler.

Read-once CFTP

Do CFTP, but only use each U_t once (Wilson, 1999).

Motivation:

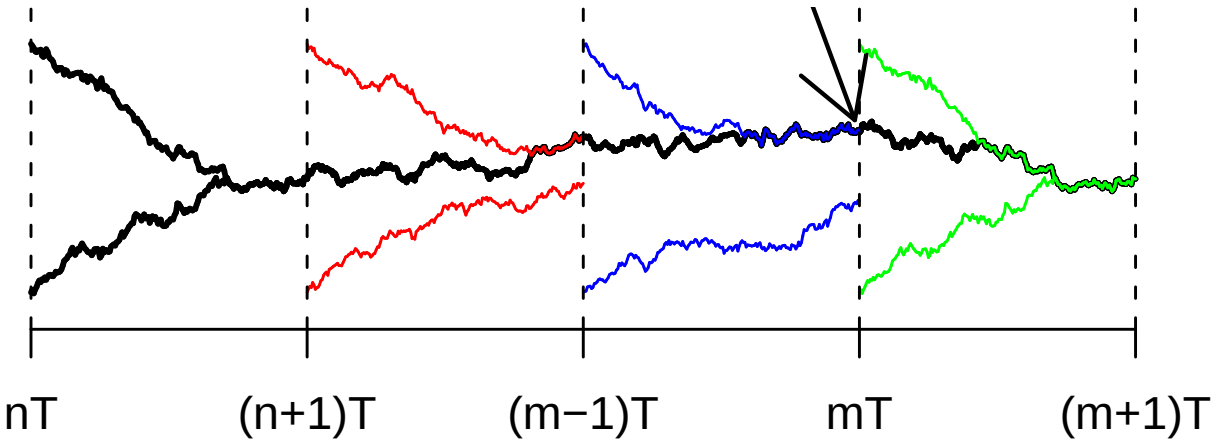


$C_{-n} \stackrel{\Delta}{=} \{\text{Paths from } -nT \text{ coalesce by } -(n-1)T\}$

The pair $(C_{-n}, X_{-(n-1)T})$ depends only on $\{U_t\}_{t=-n+1}^{t=-(n-1)T}$, so such pairs are independent across n.

ROCFTP–The Algorithm

1. Run from $t = 0$ to the first C_n event.
2. Follow the path from $X_{(n+1)T}$.
3. Run from $t = (n+1)T$ until C_m, $m > n$, occurs.
4. Output X_{mT}.



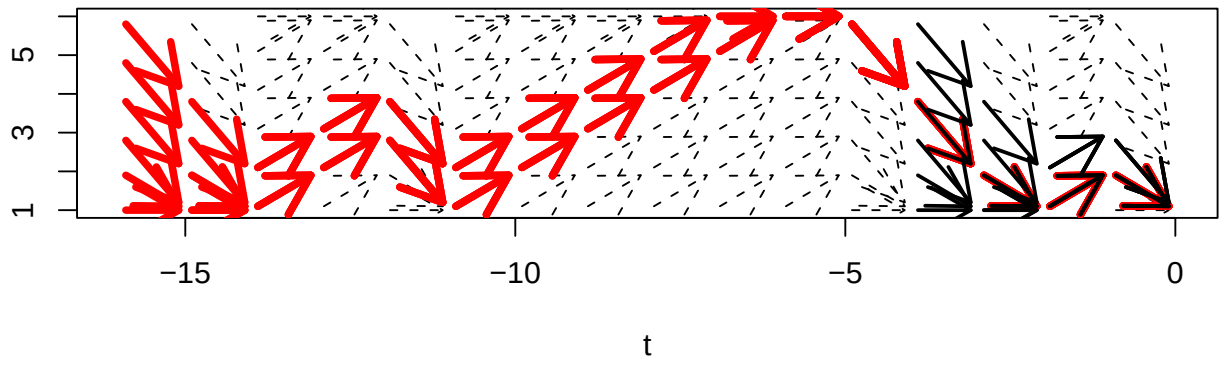
Valid since distribution of X_{mT} is the same as if a fixed stopping time was used.

This is the easiest perfect sampler to program.

Impatience Bias

CFTP may terminate, but there is no upper bound on how long it will take. If we give up when $M > M^*$, we introduce an *impatience bias*: we are more likely to see outcomes arising from fast coalescence.

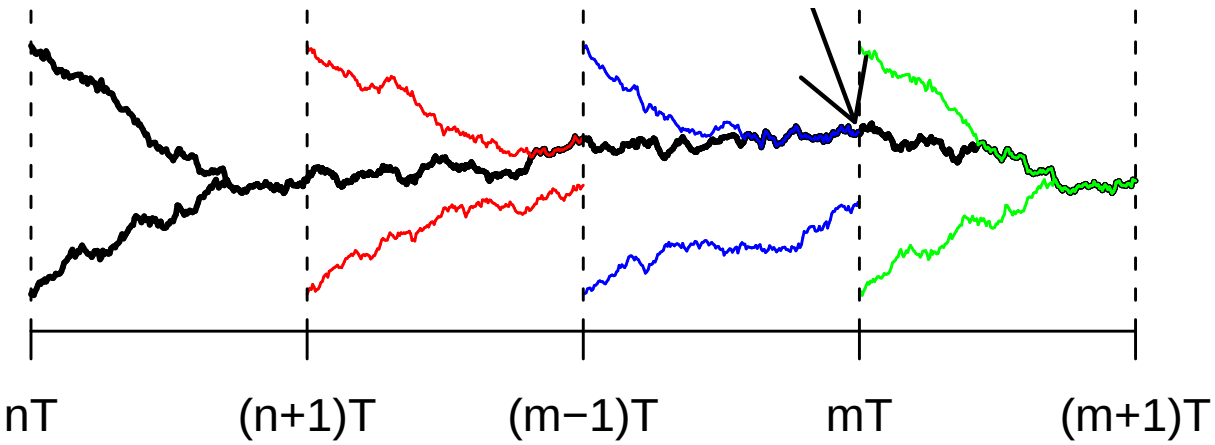
E.g. asymmetric random walk on $\{1, \ldots, N\}$ that steps up 1, down 2 can coalesce at 1 in $N/2$ steps, but needs at least $N - 1$ steps to coalesce at N.



Fill's Algorithm is interruptible

Fill's rejection sampling algorithm is a variation on ROCFTP that does *not* have impatience bias, i.e. it is interruptible.

Given coalescence, the value at the end of the block is independent of the value at the start.



This allows us to construct a rejection sampler.

The Reverse Process

Fill's algorithm makes use of the reversal of a Markov chain. The reversal Y_t of X_t has the same transition probabilities, in reverse:

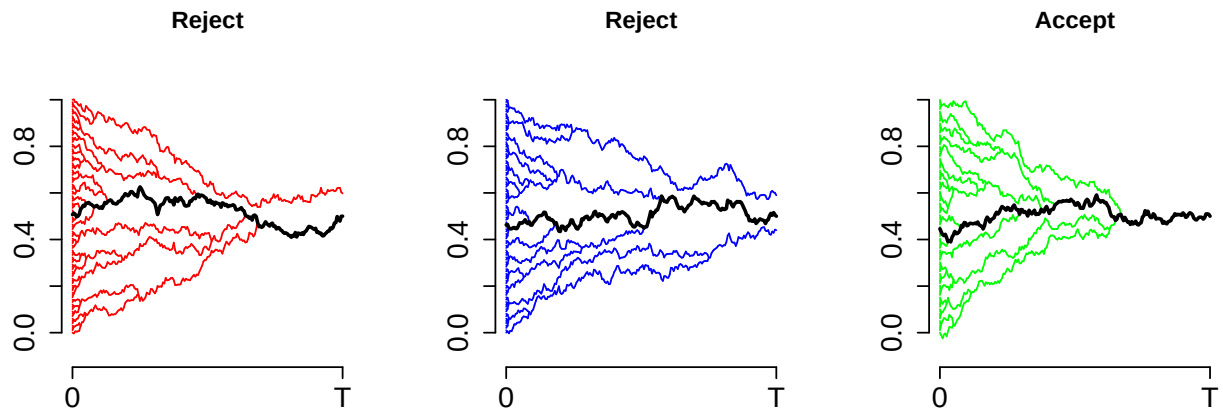
$$P(Y_{t+1} \in A | Y_t \in B) = P(X_t \in A | X_{t+1} \in B)$$

where we assume X_t is in steady-state.

For reversible chains (common in MCMC), Y_t has the same transition probabilities as X_t. If not, we may be able to use Bayes Rule to find the transition probabilities.

Fill's Algorithm

1. Arbitrarily choose X_T.
2. Using the reverse process, simulate $X_{T-1}, \ldots, X_0$.
3. Choose $U_1, \ldots, U_T$ such that $X_{t+1} = \phi(X_t, U_{t+1})$.
4. Use $U_1, \ldots, U_T$ to simulate all paths from 0 to T.
5. If all paths coalesce, output X_0.
6. Otherwise, reject and repeat independently.



Is Fill's $X_0 \sim \pi(\cdot)$?

Yes. We are constructing the value at the beginning of a coalescent block, just as ROCFTP does. The fact that X_T is chosen arbitrarily doesn't matter as long as X_T is in the support of π, since X_0 is independent of it.

Is Fill interruptible?

Yes (well, maybe). We always take T steps to generate a block, and each block is independent of all others. There is no dependence between the number of steps and the output value (though there may be dependence on the length of time to calculate each step).

Practical difficulties with Fill

Usually it is possible to work out the transition probabilities of the reversal of X_t, but it may be hard.

It is usually quite difficult to draw the U_t values. U_t are i.i.d. from a distribution that is designed to be straightforward to sample, but we require draws conditional on $X_t = \phi(X_{t-1}, U_t)$. Since $\phi(\cdot, \cdot)$ may be quite complicated in practical situations, this is not easy.

Variations

In practice, $\mathcal{X}$ is usually too big to follow every path—$\phi(B_t, U_{t+1})$ is impractical to calculate. We usually work with bounds, i.e. we define $\Phi(\cdot, U_{t+1})$ on sets such that

$$\Phi(B_t, U_{t+1}) \supseteq \phi(B_t, U_{t+1})$$

e.g. if $x \leq y$ implies $\phi(x, u) \leq \phi(y, u)$ (monotonicity), then it is sufficient to follow minimal and maximal paths to determine coalescence.

Dominated CFTP (also known as coupling into and from the past) makes use of this idea together with time reversals to handle unbounded spaces.

Dominated CFTP

When $\mathcal{X}$ is unbounded, often CFTP fails. If the step size $X_{t+1} - X_t$ stays bounded, then there is no M large enough that coalescence can happen (the process is not uniformly ergodic).

Kendall and Møller solved this by coupling X_t to a more-easily simulated dominating process D_t, i.e. defining

$$\begin{aligned} X_{t+1} &= \phi_1(X_t, U_{t+1}) \\ D_{t+1} &= \phi_2(D_t, U_{t+1}) \end{aligned}$$

in such a way that $D_t \geq X_t$ implies $D_{t+1} \geq X_{t+1}$.

We then simulate D_t backwards from its steady-state distribution at time 0, and use it to bound X_t.

Conclusions

- All of these algorithms are closely related, being based on the idea of coalescence of coupled paths.
- CFTP is easiest to understand, but ROCFTP is easiest to implement.
- Routine use of ROCFTP is possible, but not always: some kind of [partial] ordering is essential in a large state space.
- CFTP generalizes to non-Markov processes more easily than ROCFTP does.

For more on perfect sampling, see David Wilson's bibliography:

`http://dimacs.rutgers.edu/~dbwilson/exact.html`