CFTP with Read-Once Randomness

Emilia Oikarinen

Laboratory for Theoretical Computer Science

Helsinki University of Technology

10th November 2003

Motivation

- CFTP introduced earlier has drawbacks — old random numbers have to be re-used.
- Requires storage for the random numbers — memory costs.
- Recall the problems with using new random numbers on each restart or running the Markov chains into-the-future until coalescence.

This talk is based on:

- D. Wilson, How to Couple from the Past Using a Read-Once Source of Randomness. *Random Structures and Algorithms* 16, pp. 85-113, 2000.
- O. Häggström, *Finite Markov Chains and Algorithmic Applications*, Section 12. Cambridge University Press, 2002.

Contents

- A Variation of the CFTP Algorithm
- CFTP with Read-Once Randomness
- Computational Aspects
- Conclusions

A Variation of the CFTP Algorithm

Idea of read-once randomness: Use coupling into-the-future, but don't stop when coalescence is reached. Instead continue for an extra random amount of time.

We consider first some modifications of the from-the-past algorithm:

- Recall that instead of $(N_1, N_2, \ldots) = (1, 2, 4, 8, \ldots)$ we can use any strictly increasing sequence of positive numbers.
- Let $N_1 < N_2 < \cdots$ be a *random* strictly increasing sequence of positive integers independent of $U_0, U_{-1}, U_{-2} \ldots$ used in the CFTP algorithm.
- Thus CFTP with starting times $-N_1, -N_2, \ldots$ produces unbiased sample from the target distribution.

- Note, that there is no harm to run the chains from a few more earlier starting times $-N_i$ after coalescence at time 0.
- Let

$$\begin{aligned} N_1 &= N_1^* \\ N_2 &= N_1^* + N_2^* \\ N_3 &= N_1^* + N_2^* + N_3^* \\ \vdots & \quad \vdots \end{aligned}$$

  where $(N_1^*, N_2^*, \ldots)$ is an i.i.d. sequence of positive random integer-valued variables.
- Let distribution of N_i^*'s to be the same as needed to get coalescence in the coupling into-the-future algorithm.

Probability of Coalescence

- **Claim:** The probability the the CFTP algorithm starting from time $-N_1 = -N_1^*$ results in coalescence by time 0 is at least $\frac{1}{2}$.
- **Proof:** Let M_1 be the number of steps needed to get coalescence in CFTP algorithm starting at time $-N_1$ (running past time 0 if needed).

  M_1 and N_1^* have the same distribution and are independent. Thus (by symmetry) $\mathbf{Pr}[M_1 \leq N_1^*] = \mathbf{Pr}[M_1 \geq N_1^*]$.

Furthermore, $\mathbf{Pr}[M_1 \leq N_1^*] + \mathbf{Pr}[M_1 \geq N_1^*]$

$$= 1 - \mathbf{Pr}[M_1 > N_1^*] + 1 - \mathbf{Pr}[M_1 < N_1^*]$$
$$= 2 - (\mathbf{Pr}[M_1 > N_1^*] + \mathbf{Pr}[M_1 < N_1^*])$$
$$= 2 - \mathbf{Pr}[M_1 \neq N_1^*]$$
$$\geq 2 - 1 = 1.$$

Thus $\mathbf{Pr}[M_1 \leq N_1^*] \geq \frac{1}{2}$ and the claim holds.

- More generally, the probability the the CFTP algorithm starting from time $-N_j = -(N_1^* + N_2^* + \cdots + N_j^*)$ results in coalescence by time $-N_{j-1}$ is at least $\frac{1}{2}$.

A Successful Restart

- We say that jth restart is *successful* if it results in coalescence no later than time $-N_{j-1}$.
- Thus each restart has probability of at least $\frac{1}{2}$ of being successful.
- The probability of successful restart is not equal to $\frac{1}{2}$ only if there is a tie $M_j = N_j^*$, where M_j is the amount of time needed to get coalescence starting from time $-N_j$ in the CFTP algorithm.
- To simplify things, we prefer to work with a probability of *exactly* $\frac{1}{2}$.

A $*$-Successful Restart

- We say that jth restart is $*$-*successful* if either $M_j < N_j^*$ or $M_j = N_j^*$ and a fair coin toss comes up heads.
- Clearly, each restart has probability $\frac{1}{2}$ of being $*$-successful.
- Now, we have a *strange but correct* (unbiased) variant of the CFTP algorithm that generates starting times $-N_1, -N_2, \ldots$ and continues until a restart is $*$-successful.
- But no read-once randomness yet ...

CFTP with Read-Once Randomness

- We need to understand the distribution of the number of $*$-failing (opposite to $*$-successful) restarts before getting a $*$-successful restart in our algorithm.
- **Geometric distribution:** If we have a coin with heads-probability p which we toss repeatedly and independently until it comes up heads, then the number of tails Y is geometrically distributed with parameter p; $\mathbf{Pr}[Y = n] = p(1-p)^n$.
- The number of $*$-failing restarts Y is clearly geometrically distributed with parameter $\frac{1}{2}$. Thus the final (i.e. $*$-successful) restart takes place at time $-N_{Y+1}$.

CFTP with Read-Once Randomness Cont'd

- Thus we first run the chains from time $-N_{Y+1}$ to time $-N_Y$, then from $-N_Y$ to time $-N_{Y-1}$ and so on until time 0.
- No prior attempts with starting times that fail to give coalescence at time 0 are needed!
- How can this be achieved?

A Twin Run

- Run two independent copies of the coupling into-the-future algorithm until they *both* coalescence.
- The copy that coalescences first is the *winner* and the other is the *loser*, and a fair coin toss is used as a tie-breaker if they coalescence simultaneously.
- We call this a *twin run*.

The crucial observation is, that the evolution of the Markov chain from time $-N_{Y+1}$ to time $-N_Y$ has exactly the same distribution as the evolution of the winner of a twin run.

How to proceed from time $-N_Y$ to time 0?

- We get from time $-N_{Y+1}$ to time $-N_Y$ using the twin run.
- Then simulate a geometric random variable Y with parameter $p = \frac{1}{2}$ to determine the number of $*$-failing restarts.
- If $Y = 0$, we have coalescence at time $-N_Y = 0$ and we're done.
- Otherwise, $Y \geq 1$. The value $X(-N_Y)$ has been established using the first twin run.
- To simulate the evolution from time $-N_Y$ to time $-N_{Y-1}$ we use another twin run.

- We let the chain evolve as in the *loser* of the twin run, where the loser runs from time 0 until the time winner gets coalescence.
- This gives precisely the right distribution of the evolution $(X(-N_Y), X(-N_Y+1), \ldots, X(-N_{Y-1}))$.
- Then simulate the chain from time N_{Y-1} to N_{Y-2} in the same way using another twin run and so on until time 0.
- The value of the chain at time 0 has exactly the same distribution as the variant of CFTP algorithm described earlier.

Thus we have a **unbiased sample from the stationary distribution without storing** or **rereading** any of the **random numbers**.

Computational Aspects

- Expected running time is of the same order of magnitude as in the original CFTP algorithm.
- There are applications in which the read-once CFTP is up to logarithmically faster.

Conclusions

- Exact sampling without need to re-use/store any random numbers.
- Expected running time is of the same order of magnitude as in the original CFTP algorithm.