Stochastic Petri Nets and Discrete Event Simulation Notes

Sean L. Wu

2/11/2021

Table of Contents

Discrete Time Petri Nets (Molloy 1985 DOI:10.1109/TSE.1985.232230)

For a discrete time stochastic Petri net (DTSPN), on each time step, if no events are in conflict, then the probability of any combination of firings and non-firings (a binary vector) will simply be a product of ρ_i and $1 - \rho_i$. The probability ρ_i are assigned by the designer *a priori*, and are "the probability that the enabled transition t_i fires at the next time step, given (conditioned on) the fact that no other transition fires." This means the firing times are Geometrically distributed, if $\rho_i < 1$.

Molloy's equation 4 is:

$$\rho_i = \frac{P[E_i]}{P[E_i \cup E_0]}$$

The conditioning gives the denominator. So it's made up of more fundamental building blocks that the conflict resolution rule in the Petri Net simulator has to deal with. That's what he calls "deconditioning".

In a set of mutually exclusive transitions $\{t_i\}$, that equation holds for all, so using the constraint all $0 < \rho_i < 1$, one can solve for $P[E_0]$ (he does not show this). Let's consider the simple case with 2 mutually exclusive transitions. We get 3 equations, and 3 unknowns (the elementary unconditional probabilities). The denominators are sums here, he writes unions but because the events are mutually exclusive the intersection, coming from the inclusion-exclusion principle is the zero set.

$$P[E_0] + P[E_1] + P[E_2] = 1 \rho_1 = \frac{P[E_1]}{P[E_1] + P[E_0]} \rho_2 = \frac{P[E_2]}{P[E_2] + P[E_0]}$$

This is a nonlinear system of equations so we have to solve it the hard way. Let's solve for $P[E_1]$ first. To do so, rewrite the third equation as $P[E_2] = \rho_2(1 - P[E_1])$, where we used the fact that $P[E_2] + P[E_0] = 1 - P[E_1]$. Then, rewrite the second equation using $P[E_1] + P[E_0] = 1 - P[E_2]$, and then substitute in the definition of $P[E_2]$ in terms of only ρ_2 (known) and $P[E_1]$, so we're back to one equation and one unknown, and solve for $P[E_1] = \frac{\rho_1 - \rho_1\rho_2}{1 - \rho_1\rho_2}$. $P[E_2]$ is solved analogously to get $P[E_2] = \frac{\rho_2 - \rho_1\rho_2}{1 - \rho_1\rho_2}$. And finally $P[E_0] = 1 - P[E_1] - P[E_0]$.

The case for $\{t_i\}$ sets larger than 2 follows generally. The key is just realizing that due to mutual exclusivity, the denominators of the conditional probabilities ρ_i can be changed so $P[E_0]$ does not appear in them, solve for all the $P[E_i]$ values, then use the sum to 1 constraint to get $P[E_0]$. Molloy gives a formula for the probability that a particular transition in a set fires to be:

$$P[E_i] = \frac{\frac{\rho_i}{1-\rho_i}}{1 + \sum \frac{\rho_j}{1-\rho_j}}$$

```
molloy <- function(rho,i){
  (rho[i] / (1 - rho[i]))  / ( 1 + sum( rho / (1 - rho) ))
}

rho <- c(0.5,0.25)
```

We can check our solution works against his:

```
fractions((rho[1] - prod(rho)) / (1 - prod(rho)))
```

```
## [1] 3/7
```

```
fractions(molloy(rho,1))
```

```
## [1] 3/7
```

In Julia it looks basically the same.

```
function molloy(rho,i)
  (rho[i] / (1 - rho[i])) / (1 + sum(rho ./ (1 .- rho)))
end;

rho = [0.5,0.25];

broadcast(x -> molloy(rho,x), [1,2])
```

```
## 2-element Array{Float64,1}:
##  0.4285714285714286
##  0.14285714285714288
```

If $\rho_i = 1$ is allowed for some subset $T_D \subset T_E$ in the set of enabled, mutually exclusive (conflicting) transitions, the previous equations do not make sense. Molloy gives the only reasonable conflict resolution as saying that for $t_i \in T_E$ where $\rho_i < 1$, their probability of firing $P[E_i]$ is zero. For $t_i \in T_D$:

$$P[E_i] = \frac{1}{|T_D|}$$

By making "chains" of $\rho_i = 1$ transitions and places, one can implement deterministic firing time distributions into DTSPN.

A GSMP Formalism for Discrete Event Systems (Glynn 1989) DOI:10.1109/5.21067

2 definitions:

- CVDS: continuous variable dynamical system
- DEDS: discrete event dynamical system

The CVDS/DEDS analogy

An output process $s(t)$ for CVDS is defined as $s(t) = h(x(t))$ where $x(t)$ is some transformation of the internal state of the system. Nothing here yet corresponds to a map or differential giving system evolution.

For a DEDS he lets $S(t)$ be the output process for a system with piecewise constant trajectories, such that if the DEDS is a queueing system it is:

$$S(t) = \sum_{n=0}^{\infty} S_n I(\Lambda_n \leq t < \Lambda(n+1))$$

Then, $\Delta_{n+1} = \Lambda(n+1) - \Lambda(n)$ is the time that the system dwells in the state attained by the n-th transition, and S_n is the associated output value. The output process needs the existence of a stochastic process $X = (X_n : n \geq 0)$ describing the evolution of the internal state of the system. The "observables" are related by: $(S_n, \Delta_n) = (h_1(X_n), h_2(X_n))$.

The next thing is how to describe the evolution of the internal state for each system. For CVDS we can use a mapping, $x = f(x)$, which takes you from one spot in state space to another. Interestingly enough he notes that this representation is "noncausal", creating difficulties both mathematically and computationally. For that reasons, we usually use a (possibly time-dependent) differential relationship $x'(t) = f(t, x(s) : 0 \leq s \leq t)$. Then the model is "causally defined in terms of the infinitesimal characteristics of the system", which is a local specification. For classic differential equations, the considered dynamics are $x'(t) = f(t, x(t))$.

For DEDS the "noncausal" representation is $X = f(X, \eta)$, where η is a sequence of i.i.d. random variables. The causal representation is $X_{n+1} = f_{n+1}(\eta_{n+1}, X_k : 0 \leq k \leq n)$. For Markov processes, it becomes $X_{n+1} = f_{n+1}(X_n, \eta_{n+1})$.

GSMP (Generalized Semi-Markov Processes)

To discuss GSMP, Glynn uses the $X_{n+1} = f_{n+1}(\eta_{n+1}, X_k : 0 \leq k \leq n)$ DEDS representation. S is the countable set of states, E is the countable set of events, $s \in S$ is a physical state a system may exist in, and $E(s)$ is the finite set of events that can trigger transitions out of s. For each $e \in E(s)$, a clock c_e is associated which gives the amount of time elapsed since c_e was last activated. Each clock's time elapses at a rate/speed r_{se}. The rates can be a function of state. Let $C(s)$ be the set of clock readings possible in $s \in S$.

Algorithm for GSMP To describe a general algorithm to simulate from a GSMP we're gonna need more notation. A GSMP is a GSSMC (General State Space Markov Chain) on Σ, the state space where the internal state sequence X takes values in $\Sigma = \bigcup_{s \in S} s \times [0, \infty) \times C(s)$. Therefore, the moment the system arrives in a state at the n-th transition, occurring at time $\Lambda(n)$, the state is given by $X_n = (S_n, \Delta_n, C_n)$. The first element of the state is the physical state, the second is the dwell time, and the third is the vector of (valid) clock readings. Because the clocks accrue time deterministically within a physical state, the clock vector upon entry to the state is sufficient to describe the system. The dwell time still needs to be accounted for in the state as the joint probability of a jump and destination state change as it accrues, meaning that the probability distribution any simulation algorithm should sample from is incomplete without incorporating it.

Let $\overrightarrow{X}_n = (X_0, ..., X_n)$ be a random variable and $\overrightarrow{x}_n = (x_0, ..., x_n)$ be a realization (sampled trajectory) from it, where $x_i \in \Sigma$.

For each $e \in E(s)$:

- CDF $F(t; \overrightarrow{x}_n, e)$ over values $0 \leq t$ giving the probability e fires at time t, if it were the first to fire.
- The survival is $\overline{F}(t; \overrightarrow{x}_n, e) = 1 - F(t; \overrightarrow{x}_n, e)$.

- $G(t+dt;\overrightarrow{x}_n,e) = F(t+dt;\overrightarrow{x}_n,e)\prod_{e'\in E(S_n);e'\neq e}\overline{F}(t;\overrightarrow{x}_n,e')$: is the joint probability that e fires at time $t+dt$, so that $\Delta_{n+1}\in dt$, such that $e^*_{n+1}=e$ (where asterisk means it attains the minimum, is the first to fire).

Glynn calls F a residual life distribution. I suspect that means it's defined as the distribution over time remaining to fire, if "time" is a normalized, unit rate quantity, putting everything on an even footing. Otherwise the division we will see later on by the current clock speed doesn't make sense. So when it's sampled its asking the question "if this event has accured this much unit time so far, how much unit time is left before it fires?" It should be equivalent to integrated hazard.

In his notation $F_a(x;\overrightarrow{x}_n,e) = F(ax;\overrightarrow{x}_n,e), a\geq 0$ to make the probabilities relative to the entry point $\Lambda(n)$ but I think it's more clear to Just Deal With it (the extra notation).

The simulation algorithm upon arriving in a state $\overrightarrow{X}_n$ to sample X_{n+1} is:

1. For each enabled event $e\in E(S_n)$, sample a random variate $Y_{e,n}\sim F(\cdot;\overrightarrow{X}_n,e)$.
2. Set $\Delta_{n+1}=\min\{Y_{e,n}/r_{S_n,e} : e\in E(S_n)\}$ and e^*_{n+1} is the unique event which achieves the minimum for Δ_{n+1}.
3. Sample the new physical state S_{n+1} from $p(\cdot;\overrightarrow{X},e^*_{n+1})$.
4. Set:
    1. $C_{n+1,e}=-1$, for $e\notin E(S_{n+1})$ (newly disabled clocks are set to some null value)
    2. $C_{n+1,e}=0$, for $e\in N(S_{n+1};S_n,e^*_{n+1})$ (newly enabled clocks are set to zero)
    3. $C_{n+1,e}=C_{n+e}+\Delta_{n+1}r_{S_n,e}$, for $e\in 0(S_{n+1};S_n,e^*_{n+1})$ (old clocks that still run are updated with how much they ticked in the last state dwell)

Such an algorithm can sample from time-inhomogeneous generalized semi-Markov processes. It's inhomogeneous because the residual life distributions F and transition p can depend on the entire history of X. Also, it probably draws far more random numbers than it needs to. Anderson would later (18 years later, to be specific) cut down the number to just 1 per event (after initialization), and also demystified the "residual life" distributions via random time changes. Thank god.

A quick note on Exponential and Geometric processes from the pomp documentation

When approximating a continuous time process with a discrete time one, fixups have to be made in order to have a proper approximation, meaning, something that will converge to the other one as $\Delta t\to 0$. We'll look at this through the example of approximating an Exponential distribution waiting time with a Geometric distribution, on a time step Δt. The pomp docs do this. Consider a process that has expected time T such that:

$$T\sim \text{Exponential}(R)$$

Then if we set:

$$r=\frac{\log(1+R\Delta t)}{\Delta t}$$

And then the discrete process:

$$N\sim \text{Geometric}(p=1-e^{-r\Delta t})$$

Has the same expected waiting time $\mathbb{E}[N\Delta t] = \mathbb{E}[T]$. They say "in particular, $r \to R$ as $\Delta t \to 0$.

Why should r be equal to this strange equation? Intuitively it would seem setting $r = R\Delta t$ or something simple should work right? Not quite. Let's first evaluate their claim about the limit.

1. $\lim\limits_{\Delta t \to 0} \frac{\log(1+R\Delta t)}{\Delta t}$ has an indeterminate form, so we apply L'Hopital's rule.
2. $\lim\limits_{\Delta t \to 0} \frac{\frac{d}{d\Delta t}(\log(1+R\Delta t))}{\frac{d}{d\Delta t}(\Delta t)} = \frac{\left(\frac{1}{1+R\Delta t}\right)\frac{d}{d\Delta t}(1+R\Delta t)}{1} = \left(\frac{1}{1+R\Delta t}\right)\left(\frac{d}{d\Delta t}(1) + \frac{d}{d\Delta t}(R\Delta t)\right) = \frac{R}{1+R\Delta t}$
3. In L'Hopital's rule, *don't foget to actually take the limit*: $\lim\limits_{\Delta t \to 0} \frac{R}{1+R\Delta t} = R$