

# SMART: Simulation and Markovian Analyzer for Reliability and Timing

Gianfranco Ciardo      Andrew S. Miner

Department of Computer Science, College of William and Mary, Williamsburg, VA 23187  
{ciardo,asminer}@cs.wm.edu

## Abstract

SMART is a new tool for performance, reliability, availability, and performability modeling. Numerical solution algorithms are available for both continuous- and discrete-time Markov chains. Mixed-time non-Markovian models can be studied using simulation. Multiple interacting models and fixed-point iterative techniques for the decomposition and solution of complex models can be easily specified. To assist in the model specification and help in avoiding common mistakes, the input language is strongly typed.

## 1 Main features of SMART

SMART allows to describe explicit **state-space-based models**: semi-Markov processes (SMPs), independent SMPs (ISMPs), continuous time Markov chains (CTMCs), and discrete time Markov chains (DTMCs); or **high-level formalisms**: stochastic Petri nets (SPNs) and queueing networks (QNs), which define an underlying generalized semi-Markov process (GSMP) or, in special cases, a Markov-regenerative process (MRGP), a SMP, an ISMP, a CTMC, or a DTMC. The last two can be solved numerically for steady-state or transient analysis, while discrete-event simulation can be used for any process.

The strictly-typed declarative input language has predefined **types** (`bool`, `int`, and `real`), and **natures** (`const`, `ph`, `rand`, and `proc`). These can be organized in **arrays** and **sets**.

Functions, possibly recursive, are easily defined:

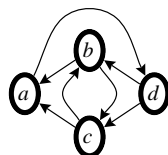
```
int fact(int n) := if(n==0,1,n*fact(n-1));
```

The `for` statement is useful for parametric studies:

```
for (int i in {0..9}, int j in {0..9}) {
  int measure[i][j] := mymodel(i,j); }
```

Fixed-point iterative solutions can be described. SMART determines whether enough initial values are provided: the dependency graph must be acyclic after removing the nodes with a “guess”.

```
converge {
  real d guess 0.5;
  real b guess 0.5;
  converge {
    real c := fc(d,b);
    real b := fb(d,c); }
  real a := fa(b,c);
  real d := fd(a); }
```

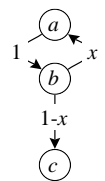


SMART understands and manipulates random variables with discrete or continuous time phase-type distributions, `ph int` or `ph real`. Random variables with general distributions, `rand int` or `rand real`, are managed through simulation.

```
ph real X := expo(0.1);
ph real Y := expo(0.2);
ph real sumXY := X+Y;
ph real prodRX := 2.5*X;
ph real chooseXY := choose(0.3,X,0.7,Y);
ph real minXY := min(X,Y);
ph real maxXY := max(X,Y);
ph real geomX := geom(0.1,X);
rand real prodXY := X*Y;
```

State-space-based or high-level models are defined by calling predefined functions that specify their components. Output measures can be defined as well:

```
dtmc DD(real x) := {
  state a,b,c;
  init(a:1.0);
  arcs(a:b:1.0, b:a:x, b:c:1-x);
  real m1:=prob(at(inState(c),7));
};
real myM := DD(0.2).m1;
```



## 2 Ongoing and future work

A prototype of a distributed version is being tested. It solves different models, or the same model with different parameters, on different processors.

A graphical interface for the specification of state-space-based and high-level models is under way.

Numerical steady-state and possibly transient solution algorithms will be implemented for some of the more complex stochastic processes.

Database capabilities will be provided to allow saving and restoring partial results during long or multiple runs. Only the results that truly need to be updated will be recomputed.

## 3 Acknowledgements

This research was partially supported by the National Aeronautics and Space Administration under NASA Contract No. NAS1-19480; by a joint STTR project with Genoa Software Systems, Inc., for the Army Research Office; and by the matching grant FED-95-011 from the Virginia Center for Innovative Technology.