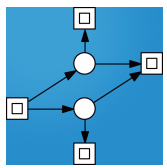
MARCIE – Model checking And Reachability analysis done effiCIEntly

Monika Heiner *Christian Rohr* Martin Schwarick



Department of Computer Science

Brandenburg University of Technology Cottbus

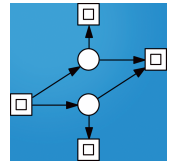
http://www-dssz.informatik.tu-cottbus.de/marcie.html

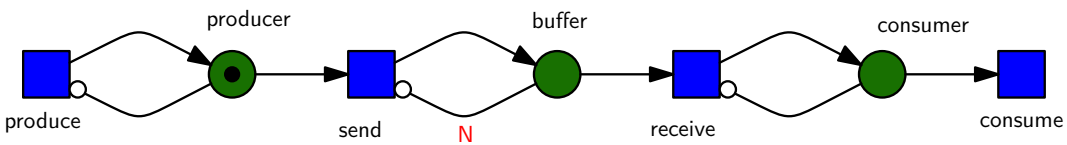
Petri Nets 2013

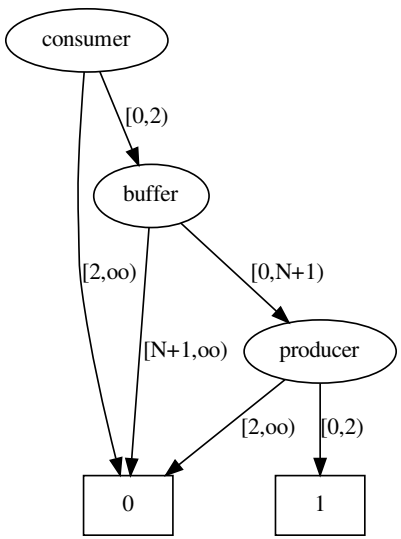
Milano

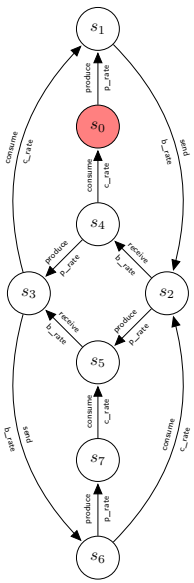
June 26, 2013

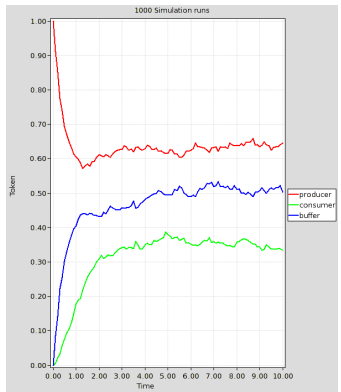
Outline



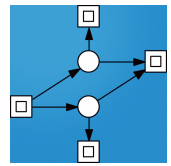


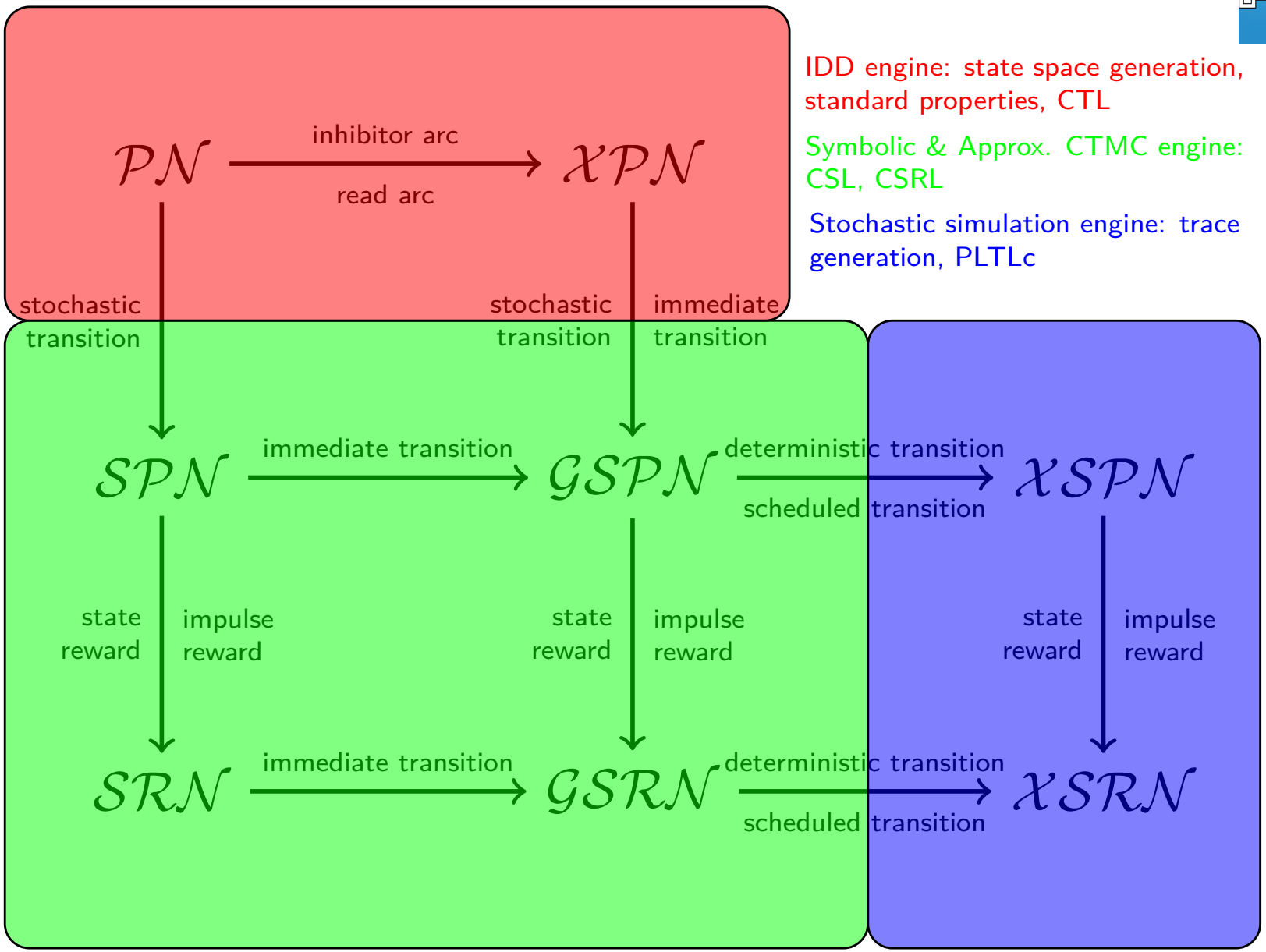






Net Classes



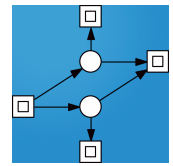


IDD engine: state space generation, standard properties, CTL

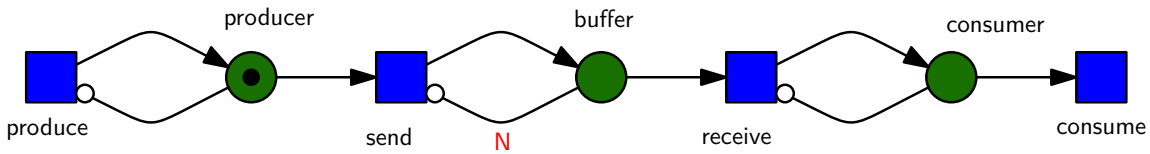
Symbolic & Approx. CTMC engine: CSL, CSRL

Stochastic simulation engine: trace generation, PLTLc

Abstract Net Description Language
(ANDL)

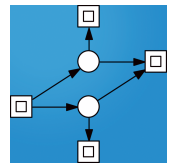


Producer & Consumer

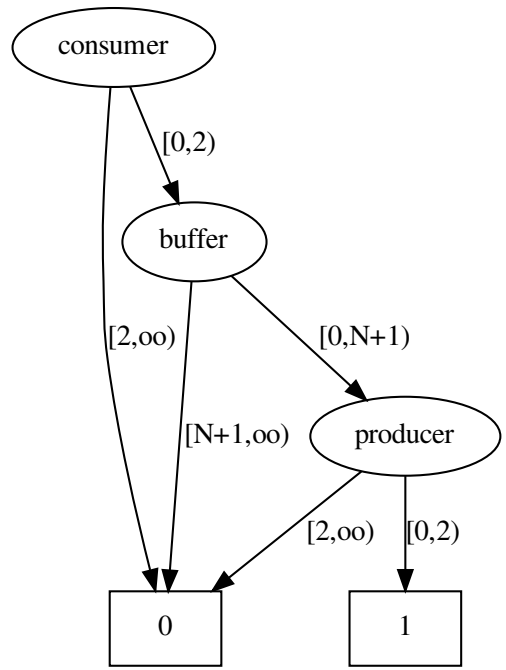



```
spn [procon] {
constants:
  int          N;        // buffer capacity
  double b_rate = 100; // buffer rate
  double p_rate = 150; // production rate
  double c_rate = 50;  // consumption rate
places:
  producer = 1;
  consumer = 0;
  buffer   = 0;
transitions:
  receive : [consumer < 1] : [consumer + 1] & [buffer - 1]   : b_rate;
  send    : [buffer < N]   : [buffer + 1]   & [producer - 1] : b_rate;
  produce : [producer < 1] : [producer + 1]                  : p_rate;
  consume :                : [consumer - 1]                  : c_rate;
}
```

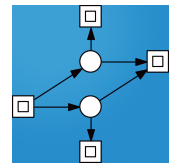
Symbolic Engine



- Interval Decision Diagrams (IDD)
- Three state space generation algorithms:
  - Breadth-First Search (BFS), Transition chaining, Saturation
- Standard properties:
  - dead states, reversibility and liveness
- Variable ordering (place & transition)
  - plain, reverse, lexical, random, from file
  - heuristics, taking the net structure into account
- can handle $\mathcal{PN}$ & $\mathcal{XPN}$



Symbolic Engine

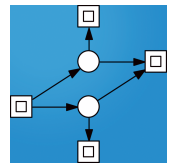


Producer & Consumer

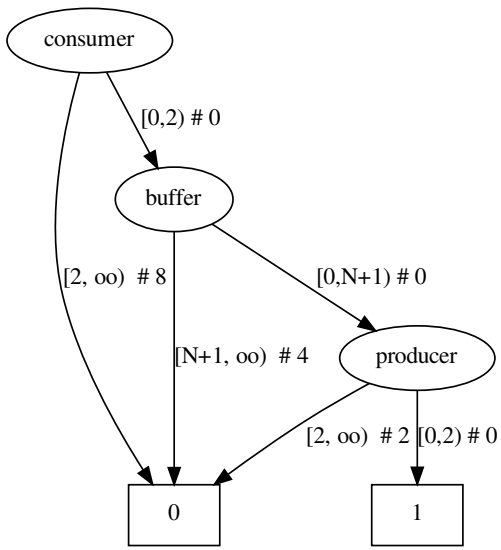
N	$\|states\|$
1	8
10	44
100	404
1,000	4,004
10,000	40,004
100,000	400,004
1,000,000	4,000,004
10,000,000	40,000,004
100,000,000	400,000,004
1,000,000,000	4,000,000,004

$\Rightarrow$ MCC 2012 & MCC 2013

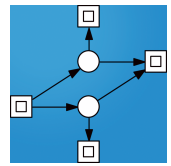
Symbolic CTMC Engine



- build upon the symbolic state space representation (IDD)
- "on-the-fly" computation of the matrix from the IDD
- 2-step approach
- multi-threaded
- transient analysis
  $\Rightarrow$ uniformization method
- steady state analysis
  $\Rightarrow$ iterative methods, e.g. Jacobi, Gauss-Seidel
- can handle $\mathcal{GSPN}$ & $\mathcal{SRN}$
- rate matrix & probability distributions can be exported in csv format



Symbolic CTMC Engine

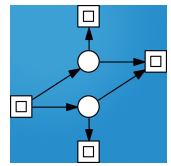


Producer & Consumer (N=1)

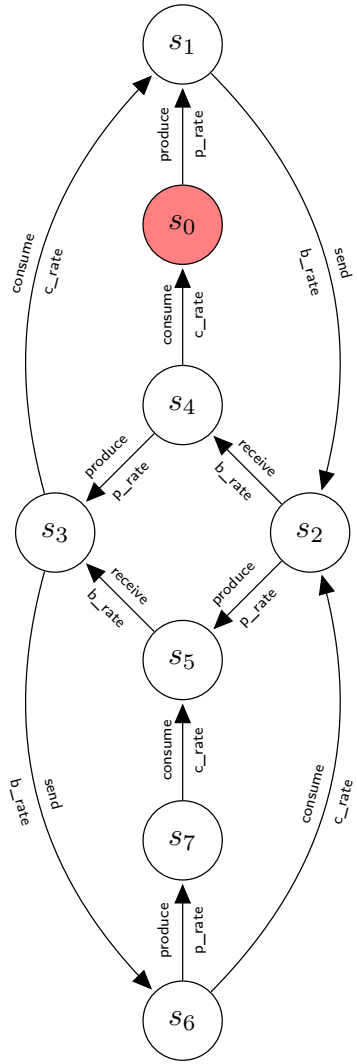
$\text{state}_{\text{idx}}$	consumer	buffer	producer	$\pi(10)$
0				0.000000418703894
1			1	0.202416456144621
2		1		0.002023964192866
3		1	1	0.402740107364029
4	1			0.000020039273592
5	1		1	0.202372017366254
6	1	1		0.002003683340260
7	1	1	1	0.200343931382354

$\Rightarrow$ CMSB 2010 & QEST 2011

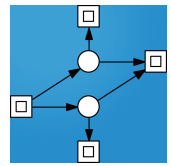
Approximative CTMC Engine



- combines BFS explicit state space construction & transient analysis using adaptive uniformization
- prune states with probability below threshold Δ $\Rightarrow$ finite subset of states
- 1-step approach
- can handle $\mathcal{GSPN}$
- weighted token values at certain time points can be exported in CSV format



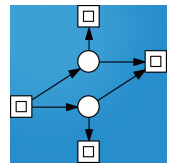
Approximative CTMC Engine



Producer & Consumer (N=1, $\Delta = 10^{-5}$)

$\text{state}_{\text{idx}}$	consumer	buffer	producer	$\pi(10)$
0				~~0.000000418703894~~
1			1	0.202416456144621
2		1		0.002023964192866
3		1	1	0.402740107364029
4	1			0.000020039273592
5	1		1	0.202372017366254
6	1	1		0.002003683340260
7	1	1	1	0.200343931382354

Approximative CTMC Engine

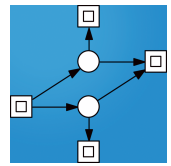


Producer & Consumer ($\Delta = 10^{-11}$)

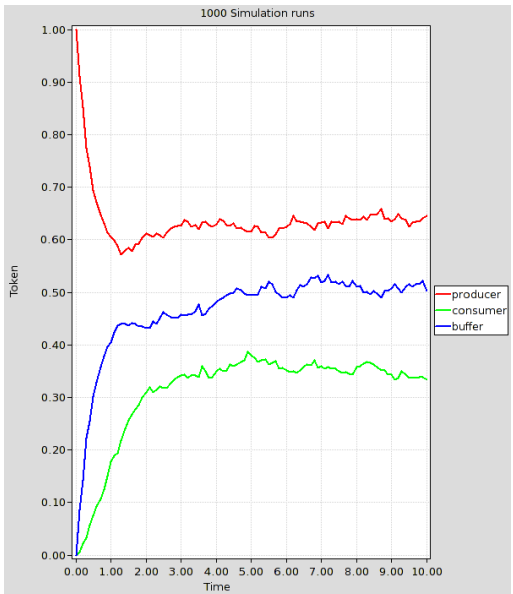
N	$\lvert states \rvert$	% of $\lvert states \rvert$
1	8	100%
10	44	100%
100	404	100%
1,000	4,004	42%
10,000	40,004	4%
100,000	400,004	$<$1%
1,000,000	4,000,004	$\ll$1%
1,000,000,000	4,000,000,004	$\ll$1%

$\Rightarrow$ CMSB 2010 & QEST 2011

Stochastic Simulation Engine

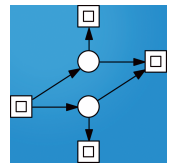


- two simulation algorithms
  - direct method by Gillespie
  - next reaction method by Gibson & Bruck
- computes traces through the state space (CTMC, non Markovian)
- constant memory consumption
- multi-threaded
- can handle $\mathcal{XSPN}$ & $\mathcal{XSRN}$
- traces can be exported in CSV format



$\Rightarrow$ CMSB 2010 & QEST 2011 & BioPPN 2012

Computation Tree Logic
(CTL)

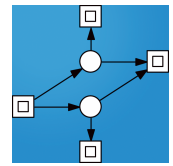


- efficient symbolic implementation of the standard CTL model checking algorithm
- supported for $\mathcal{XPN}$ using the symbolic engine

Example

$$\mathbf{A}\,\mathbf{F}(\mathit{buffer} = \mathsf{N})$$

Continuous Stochastic Logic
(CSL)

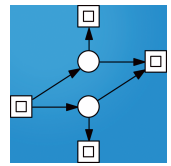


- stochastic extension of CTL
- $\mathcal{P}_{\circ p}$ replaces $\mathbf{E}$ and $\mathbf{A}$
- operator $\mathcal{S}_{\circ p}$ to reason about steady state behaviour
- $\mathcal{P}_{=?}$ and $\mathcal{S}_{=?}$ to estimate the probability
- temporal operators are decorated with time intervals
- supported for $\mathcal{SPN}$ & $\mathcal{GSPN}$ using the symbolic CTMC engine
- approximative CTMC engine supports only unnested, time-bounded formulas

Example

$$\mathcal{P}_{=?}[\mathbf{F}^{[0,t]}(\mathit{buffer} = \mathsf{N})]$$

Reward measures

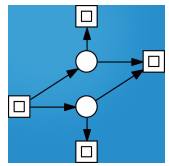


- extensions to reason about expectations of rewards
- operator $\mathcal{R}_{\bowtie r}$ with
  - $[\mathbf{C} <= t]$ the cumulative reward
  - $[\mathbf{I} = t]$ the instantaneous reward
  - $[\mathbf{F}(sp)]$ the cumulative reward until the first occurrence of a state satisfying sp
  - $[\mathcal{S}]$ expected reward at steady state
- $\mathcal{R}_{=?}$ to estimate the reward measure

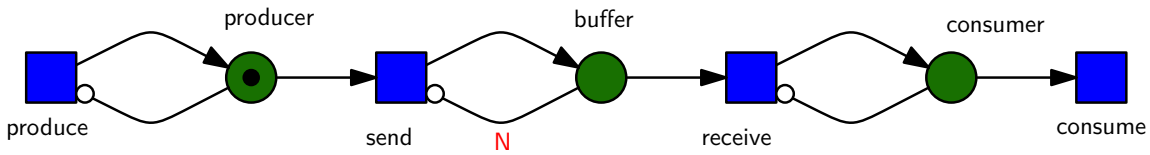
Example

$$\mathcal{R}_{=?}^{[r2r]}[\mathbf{C} <= 10]$$

Reward measures



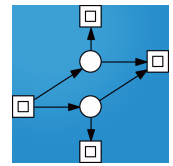
Producer & Consumer




```
spn [procon] {
constants:
  int          N;        // buffer capacity
  double b_rate = 100;   // buffer rate
  double p_rate = 150;   // production rate
  double c_rate = 50;    // consumption rate
places:
  producer = 1;
  consumer = 0;
  buffer   = 0;
transitions:
  receive : [consumer < 1] : [consumer + 1] & [buffer - 1]    : b_rate;
  send    : [buffer < N]   : [buffer + 1]   & [producer - 1] : b_rate;
  produce : [producer < 1] : [producer + 1]                  : p_rate;
  consume :                : [consumer - 1]                  : c_rate;
}
```

```
rewards [ r2r ] {
  /* states where the
     consumer is ready to
     receive, have
     a reward of 1
  */
  consumer > 0 : 1 ;
}
```

Continuous Stochastic Reward Logic (CSRL)



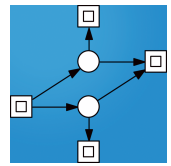
- temporal operators are decorated with a time and a reward interval
- supported for $\mathcal{SRN}$ using the symbolic engine
- supported for $\mathcal{XSRN}$ using the simulative engine, but only unnested formulas

Example

$$\mathcal{P}_{=?}^{[r2r]}[\mathbf{F}_{[t,t/2]}^{[0,t]}(\mathit{buffer} = \mathsf{N})]$$

Probabilistic Linear-time Temporal Logic with numerical constraints

(PLTLc)

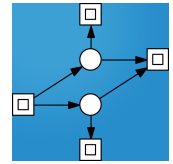


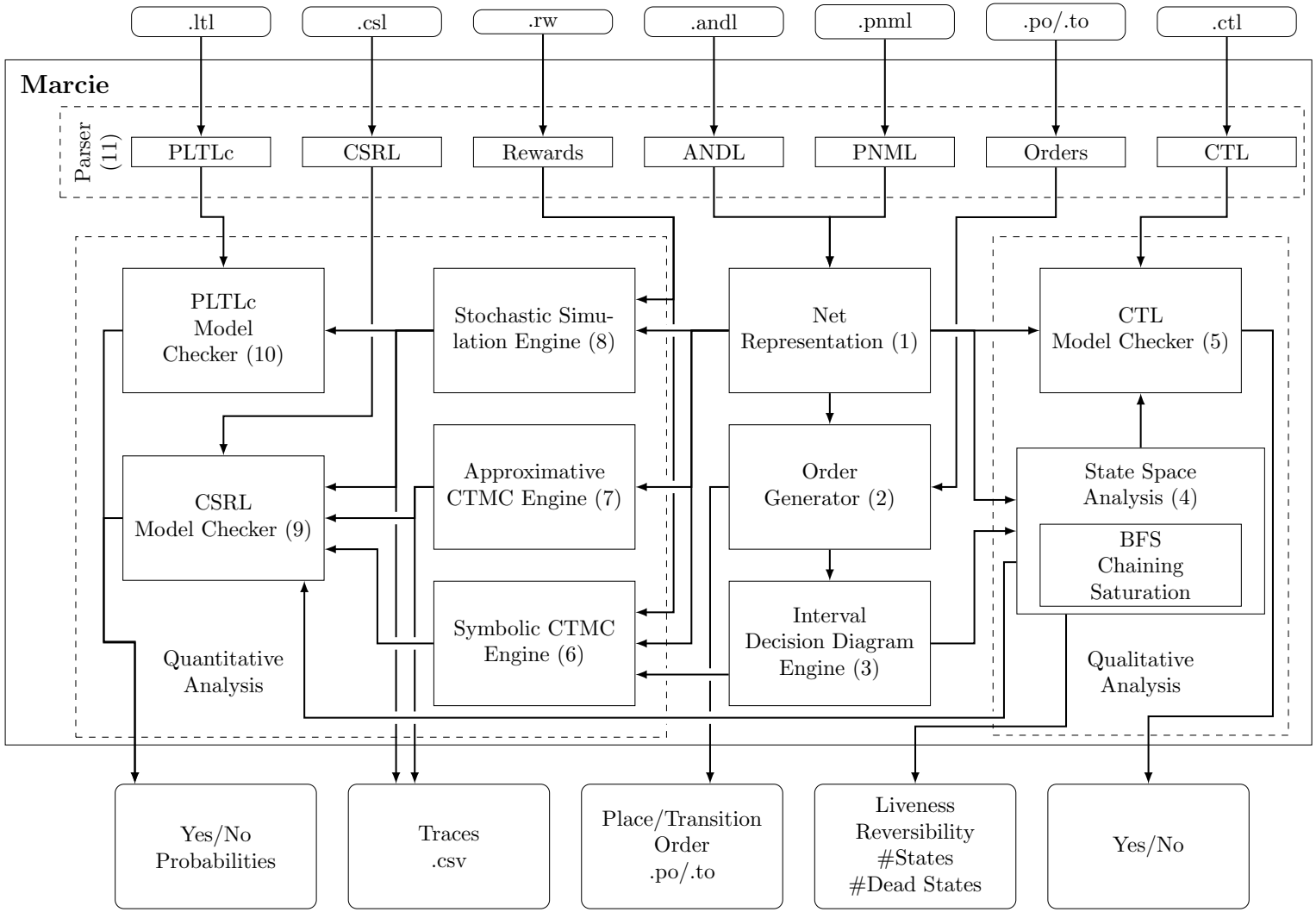
- formulas on the future of paths through the state space/CTMC
- numerical constraints are added by including free variables
- LTLc formulas return the domain $\mathcal{D}_\phi \subset \mathbb{N}^n$ of the free variables so that ϕ becomes true
- PLTLc computes probabilistic interpretation of the domains for the free variables
- supported for $\mathcal{XSPN}$ using the simulative engine

Example

$$\mathcal{P}_{=?}[\mathbf{F}^{[0,t]}(\mathit{buffer} = \$\mathsf{x})]$$

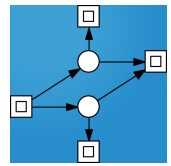
Architecture





Qualitative

Comparison

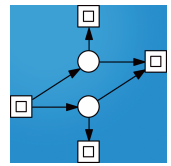


	MARCIE	Möbius	MRMC	Smart	Prism
Net classes	$\mathcal{XPN}$	$(\mathcal{XSPN})$	—	$\mathcal{GSPN}$	$\mathcal{SPN}$
State space generation	BFS, Chaining, Satura-tion	BFS	—	BFS, Chaining, Satura-tion	BFS
Orders	heuristics	plain	—	plain	plain
Standard properties	✓	—	—	(✓)	(✓)
Model checker	CTL	—	—	CTL	—

Entries in round brackets suggest a look in the tool's manual for further details.

Numerical

Comparison

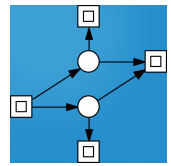


	MARCIE	Möbius	MRMC	Smart	Prism
Net classes	$\mathcal{GSPN}$	$(\mathcal{XSPN})$	$(\mathcal{SPN})$	$\mathcal{GSPN}$	$\mathcal{SPN}$
Transient	✓	✓	✓	✓	✓
Steady state	✓	✓	✓	✓	✓
Rewards	✓	✓	✓	—	✓
Model checker	CSRL	—	(CSRL)	—	CSL
Multi threading	(✓)	—	—	—	—

Entries in round brackets suggest a look in the tool's manual for further details.

Simulative

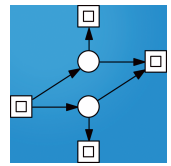
Comparison



	MARCIE	Möbius	MRMC	Smart	Prism
Net classes	$\mathcal{XSPN}$	$\mathcal{XSPN}$	$(\mathcal{SPN})$	—	$\mathcal{SPN}$
Transient	✓	✓	✓	—	✓
Steady state	✓	✓	✓	—	—
Rewards	✓	✓	—	—	✓
Model checker	(CSRL), PLTLc	—	(CSL)	—	(CSL)
Multi threading	✓	✓	—	—	—

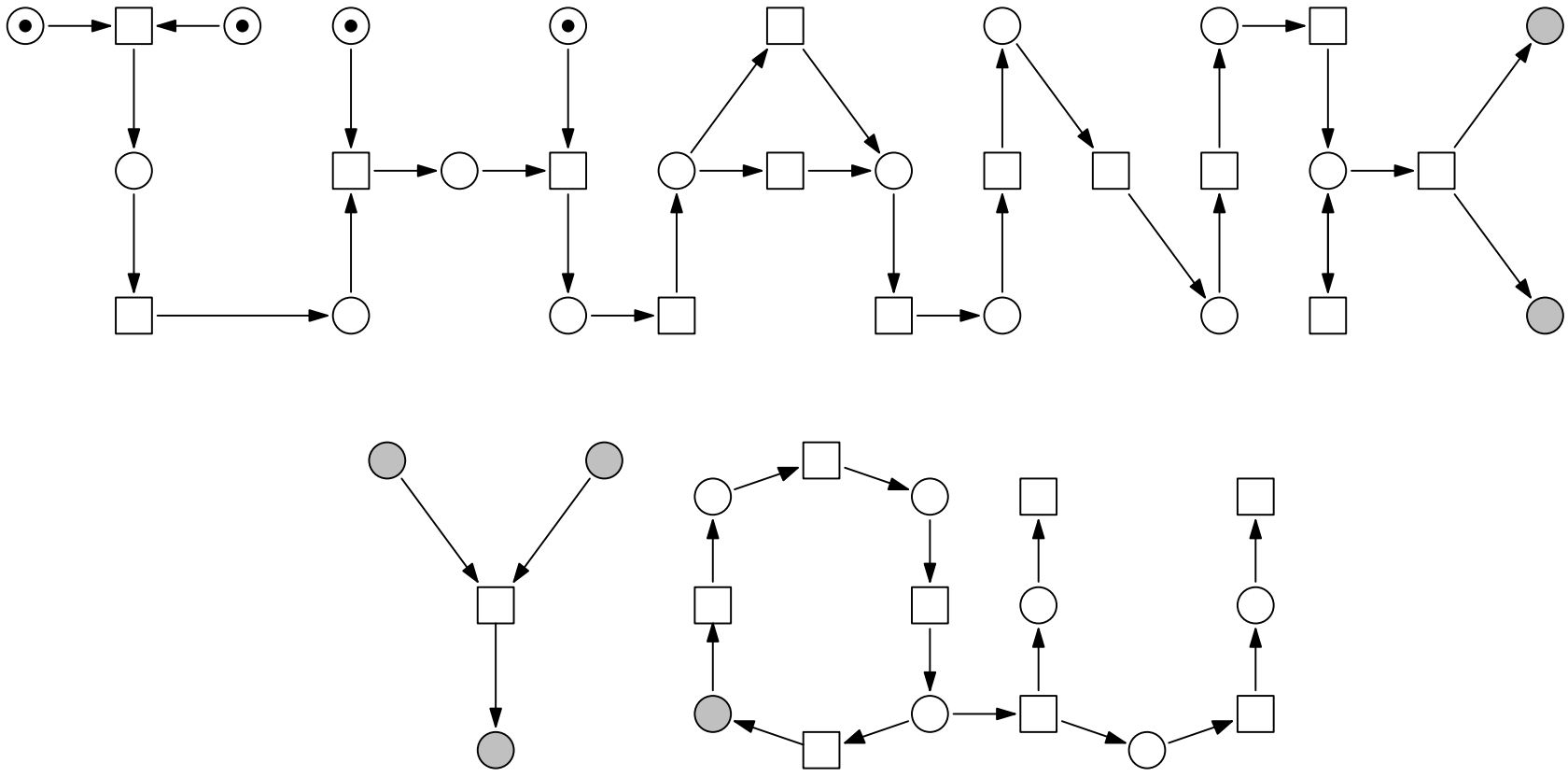
Entries in round brackets suggest a look in the tool's manual for further details.

Installation



- free of charge for academic and non-commercial use
- available for Linux and Mac OS X
- contains all dependencies; no other libraries needed
- MARCIE's website for more information, manual and examples

http://www-dssz.informatik.tu-cottbus.de/marcie.html



http://www-dssz.informatik.tu-cottbus.de/marcie.html