Decisiveness Analysis of Infinite (Dynamic) Probabilistic Models

Alain Finkel[1,2], Serge Haddad[1], Lina Ye[1,3]

(1) Université Paris-Saclay, CNRS, ENS Paris-Saclay, Laboratoire Méthodes Formelles
(2) Institut Universitaire de France (IUF)
(3) CentraleSupélec

Journées du GT Vérif, November 20, 2024

Plan

1. Preliminaries
2. Decisiveness
3. One-counter machines
4. Petri nets
5. Conclusion

Markov Chains

A Markov Chain (MC) $\mathcal{M} = (S, p)$ is defined by:

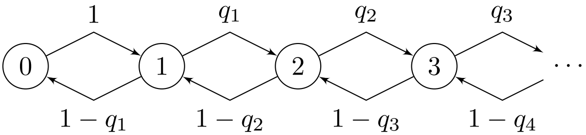
- S, a countable set of states;

$$0 \qquad 1 \qquad 2 \qquad 3 \qquad \cdots$$

Markov Chains

A Markov Chain (MC) $\mathcal{M} = (S, p)$ is defined by:

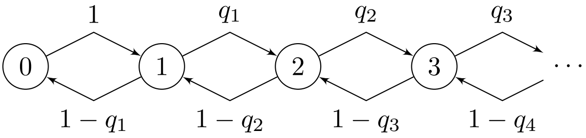
- S, a countable set of states;
- p: $S \to Dist(S)$ ($Dist(S)$: the set of distributions over S).



Markov Chains

A Markov Chain (MC) $\mathcal{M} = (S, p)$ is defined by:

- S, a countable set of states;
- p: $S \to Dist(S)$ ($Dist(S)$: the set of distributions over S).



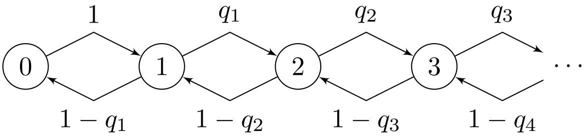
For effectivity, one requires that:

- for all $s \in S$, the support of $p(s)$ is finite and computable;
- p is computable.

Markov Chains

A Markov Chain (MC) $\mathcal{M} = (S, p)$ is defined by:

- S, a countable set of states;
- p: $S \to Dist(S)$ ($Dist(S)$: the set of distributions over S).

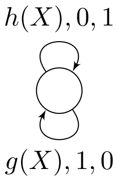


For effectivity, one requires that:

- for all $s \in S$, the support of $p(s)$ is finite and computable;
- p is computable.

Here $i \to q_i$ should be computable.

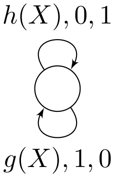
High-Level Models for Markov Chains



Markov chains are issued from non deterministic high-level models by:

- adding (computable) weights for the transitions of the model;

High-Level Models for Markov Chains



1-state and 1-counter machine

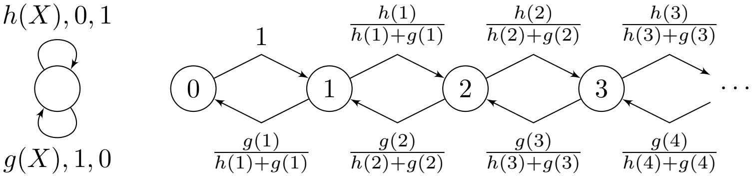
$t_{incre} : q \xrightarrow{h(X),0,1} q$: X'=X+1

$t_{decre} : q \xrightarrow{g(X),1,0} q$: if X>1, then X'=X-1

Markov chains are issued from non deterministic high-level models by:

- adding (computable) weights for the transitions of the model;

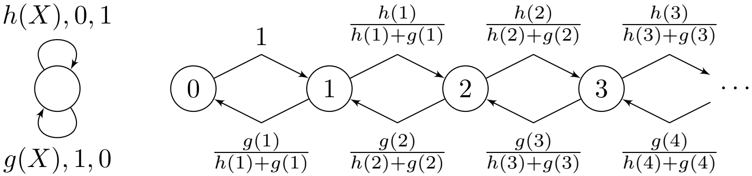
High-Level Models for Markov Chains



Markov chains are issued from non deterministic high-level models by:

- adding (computable) weights for the transitions of the model;
- given a state, getting the probabilities
  by normalization of the weights of the enabled transitions.

High-Level Models for Markov Chains



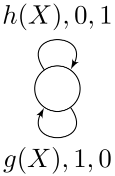
Markov chains are issued from non deterministic high-level models by:

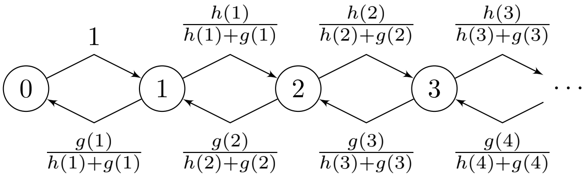
- adding (computable) weights for the transitions of the model;
- given a state, getting the probabilities
  by normalization of the weights of the enabled transitions.

The weights are *dynamic* (resp. *static*)

if they (resp. do not) depend on the current state.

High-Level Models for Markov Chains





Markov chains are issued from non deterministic high-level models by:

- adding (computable) weights for the transitions of the model;
- given a state, getting the probabilities
  by normalization of the weights of the enabled transitions.

The weights are *dynamic* (resp. *static*)

if they (resp. do not) depend on the current state.

Here the weights are static if h and g are constant.

Computing Reachability Probability

Let $\mathcal{M}$ be a Markov chain, s_0 an (initial) state, and A a subset of states, then $\mathbf{Pr}_{\mathcal{M},s_0}(\mathbf{F}A)$ represents the probability to reach A from s_0.

The Computing Reachability Probability (CRP) problem is defined by:

- Input: effective $\mathcal{M}$, s_0, effective A, and a rational number $\theta > 0$;
- Output: an interval $[low, up]$ such that $up - low \leq \theta$ and $\mathbf{Pr}_{\mathcal{M},s_0}(\mathbf{F}A) \in [low, up]$.

Computing Reachability Probability

Let $\mathcal{M}$ be a Markov chain, s_0 an (initial) state, and A a subset of states, then $\mathbf{Pr}_{\mathcal{M},s_0}(\mathbf{F}A)$ represents the probability to reach A from s_0.

The Computing Reachability Probability (CRP) problem is defined by:

- Input: effective $\mathcal{M}$, s_0, effective A, and a rational number $\theta > 0$;
- Output: an interval $[low, up]$ such that $up - low \leq \theta$ and $\mathbf{Pr}_{\mathcal{M},s_0}(\mathbf{F}A) \in [low, up]$.

How to solve CRP problem of infinite Markov chains?

- ad-hoc algorithms for particular class of probabilistic models, *e.g.*, static *Probabilistic Pushdown Automata (pPDA)* (Brádzil et al, FMSD 2013);
- generic algorithms for probabilistic models satisfying a semantical property, *e.g.*, decisiveness (Abdulla et al, LMCS 2007).

Motivation

Limitations of existing approaches

- models with only constants (static) transition weights cannot model phenomena like congestion in networks;

Our contributions

- models may contain *dynamic* weights;

Motivation

Limitations of existing approaches

- models with only constants (static) transition weights cannot model phenomena like congestion in networks;
- the decisiveness problem for some standard models are not yet studied.

Our contributions

- models may contain *dynamic* weights;
- new decisiveness results for dynamic counter machines and Petri nets.

Plan

Decisiveness

$\mathcal{M}$ is *decisive* w.r.t. $s_0 \in S$ and $A \subseteq S$ if almost surely a run starting from s_0:

- either reaches A;
- or $\overline{\mathbf{EF}A}$, the set of states from which A is unreachable.

Decisiveness

$\mathcal{M}$ is *decisive* w.r.t. $s_0 \in S$ and $A \subseteq S$ if almost surely a run starting from s_0:

- either reaches A;
- or $\overline{\mathbf{EF}A}$, the set of states from which A is unreachable.

The generic algorithm proceeds by a fair exploration of the computation tree,

- maintaining an interval which contains the reachability probability;
- ending when the length of the interval is less than a given θ.

Decisiveness

$\mathcal{M}$ is *decisive* w.r.t. $s_0 \in S$ and $A \subseteq S$ if almost surely a run starting from s_0:

- either reaches A;
- or $\overline{\mathbf{EF}A}$, the set of states from which A is unreachable.

The generic algorithm proceeds by a fair exploration of the computation tree,

- maintaining an interval which contains the reachability probability;
- ending when the length of the interval is less than a given θ.

It stops the exploration along a path when:

- either it reaches A incrementing the lower bound;
- or reaches $\overline{\mathbf{EF}A}$ decrementing the upper bound.

Decisiveness

$\mathcal{M}$ is *decisive* w.r.t. $s_0 \in S$ and $A \subseteq S$ if almost surely a run starting from s_0:

- either reaches A;
- or $\overline{\mathbf{EF}A}$, the set of states from which A is unreachable.

The generic algorithm proceeds by a fair exploration of the computation tree,

- maintaining an interval which contains the reachability probability;
- ending when the length of the interval is less than a given θ.

It stops the exploration along a path when:

- either it reaches A incrementing the lower bound;
- or reaches $\overline{\mathbf{EF}A}$ decrementing the upper bound.

Main applications.

- Static Petri nets when A is an upward closed set.
- "Quasi-Static" Lossy channel systems
  where every message has some probability to be lost at each step.

Recurrence

Let $\mathcal{M} = (S, p)$ be a Markov chain and $s \in S$. Then:

- $\mathcal{M}$ is *irreducible* if for all $s, s' \in S$, $s \rightarrow^* s'$;
- s is *recurrent* if $\mathbf{Pr}_{\mathcal{M},s}(\mathbf{XF}\{s\}) = 1$ otherwise s is *transient*.

Recurrence

Let $\mathcal{M} = (S, p)$ be a Markov chain and $s \in S$. Then:

- $\mathcal{M}$ is *irreducible* if for all $s, s' \in S$, $s \rightarrow^* s'$;
- s is *recurrent* if $\mathbf{Pr}_{\mathcal{M},s}(\mathbf{XF}\{s\}) = 1$ otherwise s is *transient*.

Let $\mathcal{M} = (S, p)$ be an irreducible Markov chain and $s, s' \in S$. Then s is recurrent if and only if s' is recurrent.

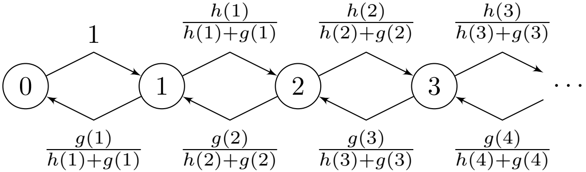
.

Recurrence

Let $\mathcal{M} = (S, p)$ be a Markov chain and $s \in S$. Then:

- $\mathcal{M}$ is *irreducible* if for all $s, s' \in S$, $s \rightarrow^* s'$;
- s is *recurrent* if $\mathbf{Pr}_{\mathcal{M},s}(\mathbf{XF}\{s\}) = 1$ otherwise s is *transient*.

Let $\mathcal{M} = (S, p)$ be an irreducible Markov chain and $s, s' \in S$. Then s is recurrent if and only if s' is recurrent.



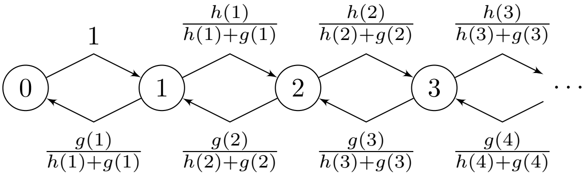
This Markov chain is irreducible. And it is recurrent if and only if $\sum_{n\in\mathbb{N}} \prod_{1\leq m<n} \rho_m = \infty$ with $\rho_m = \frac{1-p_m}{p_m}$ and $p_m = \frac{h(m)}{h(m)+g(m)}$.

Recurrence

Let $\mathcal{M} = (S, p)$ be a Markov chain and $s \in S$. Then:

- $\mathcal{M}$ is *irreducible* if for all $s, s' \in S$, $s \to^* s'$;
- s is *recurrent* if $\mathbf{Pr}_{\mathcal{M},s}(\mathbf{XF}\{s\}) = 1$ otherwise s is *transient*.

Let $\mathcal{M} = (S, p)$ be an irreducible Markov chain and $s, s' \in S$. Then s is recurrent if and only if s' is recurrent.



This Markov chain is irreducible. And it is recurrent if and only if $\sum_{n\in\mathbb{N}} \prod_{1\leq m<n} \rho_m = \infty$ with $\rho_m = \frac{1-p_m}{p_m}$ and $p_m = \frac{h(m)}{h(m)+g(m)}$.

- if $h(m) = g(m) = 1$, then $p_m = \frac{1}{2}$ and $\rho_m = 1$: thus recurrent;
- if $h(m) \geq 2$ and $g(m) = 1$, then one has $\rho_m \leq \frac{1}{2}$: thus not recurrent.

Decisiveness and Recurrence

Let $\mathcal{M}$ be a Markov chain, s_0 be a state and A be a subset of states.

Then $\mathcal{M}_{s_0,A} = (S', p')$ is defined as follows.

- S' is the smallest set containing s_0 and a new state $s_\top$ such that for all $s \rightarrow_{\mathcal{M}} s'$ with $s \in S'$, $s' \notin A$ and $s' \models \mathbf{EF}A$ one has $s' \in S'$;

Decisiveness and Recurrence

Let $\mathcal{M}$ be a Markov chain, s_0 be a state and A be a subset of states.

Then $\mathcal{M}_{s_0,A} = (S', p')$ is defined as follows.

- S' is the smallest set containing s_0 and a new state $s_\top$ such that for all $s \rightarrow_{\mathcal{M}} s'$ with $s \in S'$, $s' \notin A$ and $s' \models \mathbf{EF}A$ one has $s' \in S'$;
- $p'(s_\top, s_0) = 1$;
- for all $s, s' \in S \cap S'$, $p'(s, s') = p(s, s')$;
- for all $s \in S \cap S'$, $p'(s, s_\top) = \sum_{s' \in A \cup \overline{\mathbf{EF}A}} p(s, s')$.

Decisiveness and Recurrence

Let $\mathcal{M}$ be a Markov chain, s_0 be a state and A be a subset of states.

Then $\mathcal{M}_{s_0,A} = (S', p')$ is defined as follows.

- S' is the smallest set containing s_0 and a new state $s_\top$ such that for all $s \rightarrow_{\mathcal{M}} s'$ with $s \in S'$, $s' \notin A$ and $s' \models \mathbf{EF}A$ one has $s' \in S'$;
- $p'(s_\top, s_0) = 1$;
- for all $s, s' \in S \cap S'$, $p'(s, s') = p(s, s')$;
- for all $s \in S \cap S'$, $p'(s, s_\top) = \sum_{s' \in A \cup \overline{\mathbf{EF}A}} p(s, s')$.

Observations. $\mathcal{M}_{s_0,A}$ is irreducible and ...

$\boxed{\mathcal{M} \text{ is decisive w.r.t. } s_0 \text{ and } A \text{ iff } \mathcal{M}_{s_0,A} \text{ is recurrent.}}$

Plan

Undecidability Decisiveness Results

- Decisiveness w.r.t. a finite set is undecidable in static probabilistic counter machines (pCM).
- Decisiveness w.r.t. a finite set is undecidable in dynamic 1-state, 1-counter pCM.

Undecidability Decisiveness Results

- Decisiveness w.r.t. a finite set is undecidable in static probabilistic counter machines (pCM).
- Decisiveness w.r.t. a finite set is undecidable in dynamic 1-state, 1-counter pCM.

Undecidability Decisiveness Results

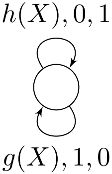
- Decisiveness w.r.t. a finite set is undecidable in dynamic 1-state, 1-counter pCM.

Sketch of proof. By reduction of the Hilbert's tenth problem (undecidable):

Input: $P(X_1, \ldots, X_k)$ an integer polynomial with k variables.
Output: whether there exists $n_1, \ldots, n_k \in \mathbb{N}$ such that $P(n_1, \ldots, n_k) = 0$.

Transform to the following 1-state and 1-counter pCM as input:

$h(n) = \min(1 + P^2(n_1, \ldots, n_k) \mid n_1 + \cdots + n_k \leq n)$ and $g = 1$.



Undecidability Decisiveness Results

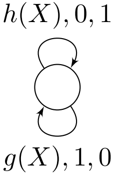
- Decisiveness w.r.t. a finite set is undecidable in dynamic 1-state, 1-counter pCM.

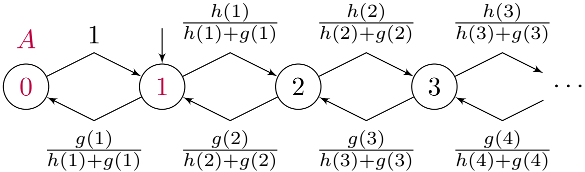
Sketch of proof. By reduction of the Hilbert's tenth problem (undecidable):

Input: $P(X_1, \ldots, X_k)$ an integer polynomial with k variables.
Output: whether there exists $n_1, \ldots, n_k \in \mathbb{N}$ such that $P(n_1, \ldots, n_k) = 0$.

Transform to the following 1-state and 1-counter pCM as input:

$h(n) = \min(1 + P^2(n_1, \ldots, n_k) \mid n_1 + \cdots + n_k \leq n)$ and $g = 1$.



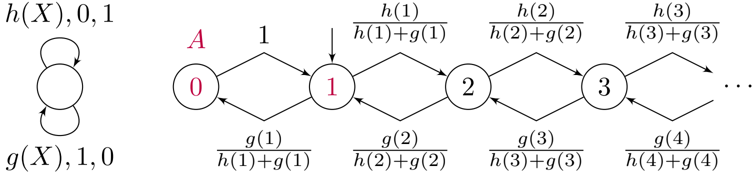


Output: Whether $\mathcal{M}$ is decisive w.r.t. s_0 and A, where $s_0 = 1$ and $A = \{0\}$.

Undecidability Decisiveness Results

- Decisiveness w.r.t. a finite set is undecidable in dynamic 1-state, 1-counter pCM.

Input: $h(n) = \min(1 + P^2(n_1, \ldots, n_k) \mid n_1 + \cdots + n_k \leq n)$ and $g = 1$.



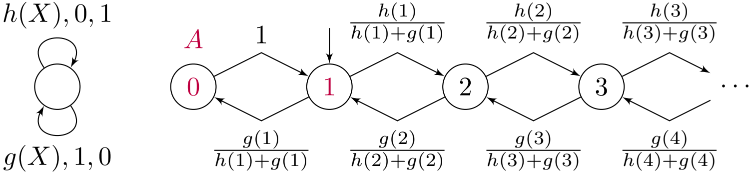
Output: Whether $\mathcal{M}$ is decisive w.r.t. s_0 and A, where $s_0 = 1$ and $A = \{0\}$.

- if there exists $n_1, \ldots, n_k \in \mathbb{N}$ s.t. $P(n_1, \ldots, n_k) = 0$, $\forall n \geq n_0 = n_1 + \ldots + n_k$, $\frac{h(n)}{h(n)+g(n)} = \frac{g(n)}{h(n)+g(n)} = \frac{1}{2}$, thus $\mathcal{M}$ is recurrent implying $\mathbf{Pr}_{\mathcal{M},1}(\mathbf{F}\{0\}) = 1$ and so decisive;
- otherwise $h(n) \geq 2$ and $\mathbf{Pr}_{\mathcal{M},1}(\mathbf{F}\{0\}) < 1$, so not decisive.

Undecidability Decisiveness Results

- Decisiveness w.r.t. a finite set is undecidable in dynamic 1-state, 1-counter pCM.

Input: $h(n) = \min(1 + P^2(n_1, \dots, n_k) \mid n_1 + \cdots + n_k \leq n)$ and $g = 1$.



Output: Whether $\mathcal{M}$ is decisive w.r.t. s_0 and A, where $s_0 = 1$ and $A = \{0\}$.

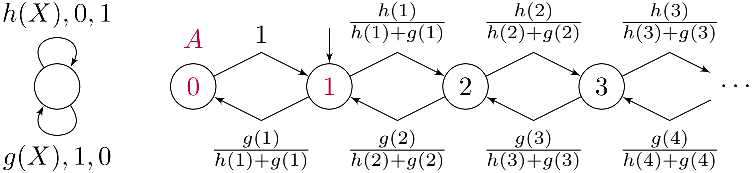
We must add restrictions on the counter machine and on the kind of weights.

The natural candidates for weights are polynomials.

Undecidability Decisiveness Results

- Decisiveness w.r.t. a finite set is undecidable in dynamic 1-state, 1-counter pCM.

Input: $h(n) = \min(1 + P^2(n_1, \ldots, n_k) \mid n_1 + \cdots + n_k \leq n)$ and $g = 1$.



Output: Whether $\mathcal{M}$ is decisive w.r.t. s_0 and A, where $s_0 = 1$ and $A = \{0\}$.

The decisiveness problem w.r.t. s_0 and finite A for polynomial 1-state, 1-counter pCM is decidable in linear time.

From QBD to Homogeneous 1-Counter Machine

Quasi Birth-Death Process (QBD) is a probabilistic model widely used and analyzed in performance evaluation.

It is equivalent to a probabilistic 1-counter machine with the following restrictions.

- Counter updates are incrementations and decrementations.
- For all states q, q', positive integers n, n' and $\Delta \in \{-1, 0, 1\}$, $\mathbf{Pr}((q,n),(q',n+\Delta)) = \mathbf{Pr}((q,n'),(q',n'+\Delta))$

From QBD to Homogeneous 1-Counter Machine

An Homogenous 1-counter Machine (HCM) is an extension of QBD where:

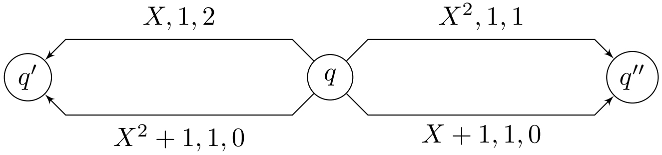
- the weights are polynomials whose single variable X is the counter value;
- but the coefficients of M are still constant.

From QBD to Homogeneous 1-Counter Machine

An Homogenous 1-counter Machine (HCM) is an extension of QBD where:

- the weights are polynomials whose single variable X is the counter value;
- but the coefficients of M are still constant.

Illustration.



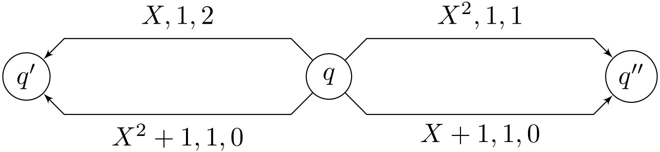
Here $M[q, q'] = M[q, q''] = 1/2$

From QBD to Homogeneous 1-Counter Machine

An Homogenous 1-counter Machine (HCM) is an extension of QBD where:

- the weights are polynomials whose single variable X is the counter value;
- but the coefficients of M are still constant.

Illustration.



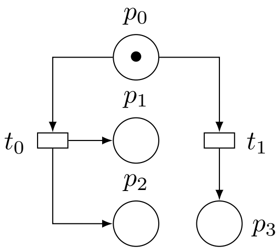
Here $M[q, q'] = M[q, q''] = 1/2$

The decisiveness problem of an HCM with M is irreducible is decidable in polynomial time (CONCUR2023).

The decisiveness problem of an HCM is decidable in polynomial time (new).

Plan

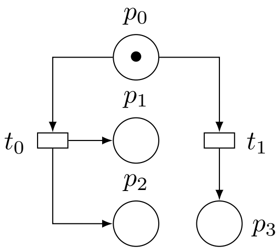
Petri Nets and two-counter machines



A petri net is a tuple $N = (P, T, F, m_0)$, where

- P is a finite set of places;
- T is a finite set of transitions;
- $P \cap T = \emptyset$
- $F \subseteq (P \times T) \cup (T \times P)$ is the flow relation
- $m_0 : P \to \mathbb{N}$ is the initial marking

Petri Nets and two-counter machines



A petri net is a tuple $N = (P, T, F, m_0)$, where

- P is a finite set of places;
- T is a finite set of transitions;
- $P \cap T = \emptyset$
- $F \subseteq (P \times T) \cup (T \times P)$ is the flow relation
- $m_0 : P \to \mathbb{N}$ is the initial marking

A variant of 2-*counter machine* $\mathcal{C}$ is defined by two counters $\{c_1, c_2\}$ and a set of instructions $\{0, \ldots, n\}$, where the instruction n is $\mathbf{halt}$, and for all $i < n$, the instruction i is

- either (1) $c_j \leftarrow c_j + 1; \mathbf{goto}\ i'$ with $1 \leq j \leq 2$ and $0 \leq i' \leq n$
- or (2) $\mathbf{if}\ c_j > 0\ \mathbf{then}\ c_j \leftarrow c_j - 1; \mathbf{goto}\ i', \mathbf{else\ goto}\ i''$ with $1 \leq j \leq d$ and $0 \leq i', i'' \leq n$

The halting problem asks, given $\mathcal{C}$ and $v_1, v_2 \in \mathbb{N}$, whether $\mathcal{C}$ eventually halts.

Polynomial Probabilistic Petri Nets: Decisiveness

The decisiveness problem of polynomial pPNs
w.r.t. an upward closed set is undecidable.

Polynomial Probabilistic Petri Nets: Decisiveness

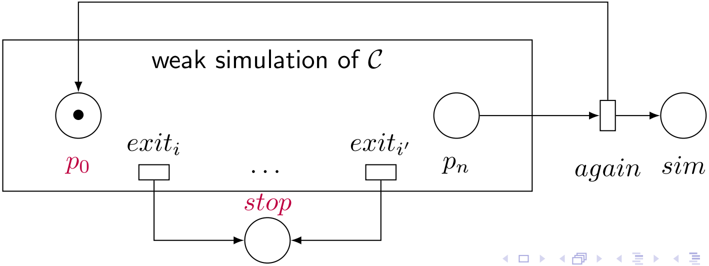
The decisiveness problem of polynomial pPNs
w.r.t. an upward closed set is undecidable.

Sketch of Proof.

By reduction of the halting problem for a *normalized* counter machine $\mathcal{C}$.

A normalized CM resets the counters at the start and the end of the computation.

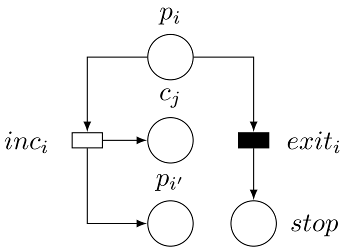
The probabilistic Petri net infinitely repeats a weak simulation for $\mathcal{C}$
incrementing a counter of simulations sim,
which is the single variable of the polynomial weights
with at each instruction some (variable) probability to exit the simulation.



Sketch of Proof (continued)

Simulation of an incrementation

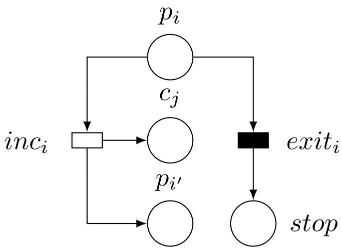
$$i : c_j \leftarrow c_j + 1; \mathbf{goto}\ i'$$



Sketch of Proof (continued)

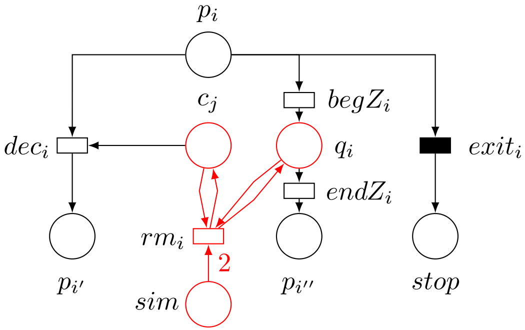
Simulation of an incrementation

$i : c_j \leftarrow c_j + 1; \textbf{goto } i'$



Simulation of a decrementation

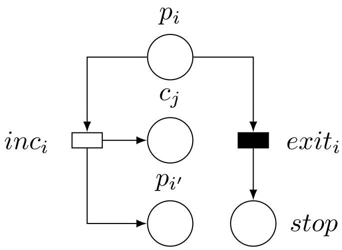
$i : \textbf{if } c_j > 0 \textbf{ then}$
$\quad c_j \leftarrow c_j - 1; \textbf{goto } i'$
$\textbf{else}$
$\quad \textbf{goto } i''$



Sketch of Proof (continued)

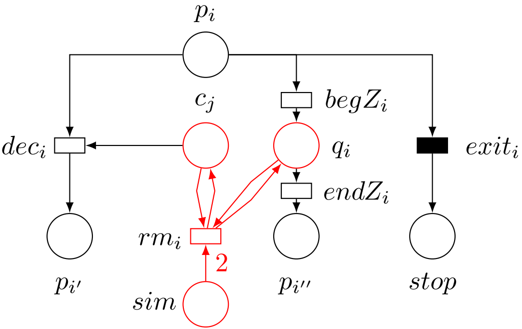
Simulation of an incrementation

$$i : c_j \leftarrow c_j + 1; \textbf{goto } i'$$



Simulation of a decrementation

i : **if** $c_j > 0$ **then**
$c_j \leftarrow c_j - 1;$ **goto** i'
else
goto i''



When cheating the net is punished by a possible decrementation of sim.

Sketch of Proof (continued)

Due to the choice of the polynomial weights, when sim goes to infinity,

- If i is an incrementation, $W(exit_i) = o(W(inc_i))$;
- If i is a decrementation, $W(exit_i) = o(W(begZ_i))$ and $W(begZ_i) = o(W(dec_i))$.

Thus the more the simulations are achieved without cheating

the less probable the net will stop or cheat.

Sketch of Proof (continued)

Due to the choice of the polynomial weights, when sim goes to infinity,

- If i is an incrementation, $W(exit_i) = o(W(inc_i))$;
- If i is a decrementation, $W(exit_i) = o(W(begZ_i))$ and $W(begZ_i) = o(W(dec_i))$.

Thus the more the simulations are achieved without cheating

the less probable the net will stop or cheat.

Assume that $\mathcal{C}$ halts.

The infinite path corresponding to the repetition of the correct simulation of $\mathcal{C}$

has a non null probability.

Thus the net is not decisive w.r.t. ${}^{\uparrow}stop$.

Sketch of Proof (ended)

Assume that $\mathcal{C}$ does not halt.

The set of paths that do not stop can be (countably) partionned as follows.

- For all $n \in \mathbb{N}$, $\mathcal{P}_n$, the set of of paths that perform exactly n simulations and never stop during the n^{th} simulation;
- $\mathcal{P}_\infty$, the set of paths that perform an infinite number of simulations.

Sketch of Proof (ended)

Assume that $\mathcal{C}$ does not halt.

The set of paths that do not stop can be (countably) partionned as follows.

- For all $n \in \mathbb{N}$, $\mathcal{P}_n$, the set of of paths that perform exactly n simulations and never stop during the n^{th} simulation;
- $\mathcal{P}_\infty$, the set of paths that perform an infinite number of simulations.

• A path in $\mathcal{P}_n$ has at most n tokens in sim implying that the probability to stop during an instruction is lower bounded by some constant.
Thus $\mathcal{P}_n$ has a null probability to avoid to mark $stop$.

Sketch of Proof (ended)

Assume that $\mathcal{C}$ does not halt.

The set of paths that do not stop can be (countably) partionned as follows.

- For all $n \in \mathbb{N}$, $\mathcal{P}_n$, the set of of paths that perform exactly n simulations and never stop during the n^{th} simulation;
- $\mathcal{P}_\infty$, the set of paths that perform an infinite number of simulations.

• A path in $\mathcal{P}_n$ has at most n tokens in sim implying that the probability to stop during an instruction is lower bounded by some constant. Thus $\mathcal{P}_n$ has a null probability to avoid to mark $stop$.

• If $\mathcal{P}_\infty$ has a non null probability, then one proves that almost surely a path in $\mathcal{P}_\infty$ achieves infinitely often a simulation with one token in sim and thus reaches $^{\uparrow}stop$: a contradiction.

Sketch of Proof (ended)

Assume that $\mathcal{C}$ does not halt.

The set of paths that do not stop can be (countably) partionned as follows.

- For all $n \in \mathbb{N}$, $\mathcal{P}_n$, the set of of paths that perform exactly n simulations and never stop during the n^{th} simulation;
- $\mathcal{P}_\infty$, the set of paths that perform an infinite number of simulations.

• A path in $\mathcal{P}_n$ has at most n tokens in sim implying that the probability to stop during an instruction is lower bounded by some constant.

Thus $\mathcal{P}_n$ has a null probability to avoid to mark $stop$.

• If $\mathcal{P}_\infty$ has a non null probability, then one proves that almost surely a path in $\mathcal{P}_\infty$ achieves infinitely often a simulation with one token in sim and thus reaches ${}^{\uparrow}stop$: a contradiction.

So the net is decisive w.r.t. ${}^{\uparrow}stop$.

Probabilistic Regular Petri Nets

A marked Petri net is *regular* if the language of its firing sequences is regular.

One can decide whether a Petri net is regular in EXPSPACE.

(Demri JCSS 2013, Blockelet & Schmitz MFCS 2011)

Probabilistic Regular Petri Nets

A marked Petri net is *regular* if the language of its firing sequences is regular.

One can decide whether a Petri net is regular in EXPSPACE.

(Demri JCSS 2013, Blockelet & Schmitz MFCS 2011)

> Probabilistic Regular Petri Nets are decisive w.r.t. a finite set whatever the weights.

Probabilistic Regular Petri Nets

A marked Petri net is *regular* if the language of its firing sequences is regular.

One can decide whether a Petri net is regular in EXPSPACE.

(Demri JCSS 2013, Blockelet & Schmitz MFCS 2011)

Probabilistic Regular Petri Nets are decisive w.r.t. a finite set whatever the weights.

Sketch of Proof. Based on the following property

(Ginzburg and Yoeli JCSS 1980)

Let $(\mathcal{N}, \mathbf{m}_0)$ be a marked regular Petri net.

There exists a bound $B(\mathcal{N}, \mathbf{m}_0)$ such that:

- for all $\mathbf{m}_1$ reachable from $\mathbf{m}_0$,
- and all $\mathbf{m}_2$ with some place p fullfilling $\mathbf{m}_2(p) + B(\mathcal{N}, \mathbf{m}_0) < \mathbf{m}_1(p)$,
- $\mathbf{m}_2$ is unreachable from $\mathbf{m}_1$.

Probabilistic Regular Petri Nets

Sketch of Proof (continued)

A finite graph is built as follows, suppose $A = \{m_1\}$

- Push on the stack $\mathbf{m}_0$.
- While the stack is not empty, pop from the stack some marking $\mathbf{m}$. Compute the set of transition firings $\mathbf{m} \xrightarrow{t} \mathbf{m}'$. Push on the stack $\mathbf{m}'$ if:
  1. $\mathbf{m}'$ is not already present in the graph,
  2. and $\mathbf{m}' \neq \mathbf{m}_1$,
  3. and for all $p \in P$, $\mathbf{m}_1(p) + B(\mathcal{N}, \mathbf{m}_0) \geq \mathbf{m}'(p)$.

Probabilistic Regular Petri Nets

Sketch of Proof (continued)

A finite graph is built as follows, suppose $A = \{m_1\}$

- Push on the stack $\mathbf{m}_0$.
- While the stack is not empty, pop from the stack some marking $\mathbf{m}$. Compute the set of transition firings $\mathbf{m} \xrightarrow{t} \mathbf{m}'$. Push on the stack $\mathbf{m}'$ if:
  1. $\mathbf{m}'$ is not already present in the graph,
  2. and $\mathbf{m}' \neq \mathbf{m}_1$,
  3. and for all $p \in P$, $\mathbf{m}_1(p) + B(\mathcal{N}, \mathbf{m}_0) \geq \mathbf{m}'(p)$.

Due to the third condition, this algorithm terminates. On the finite graph, one keeps the weights and adds loops for states without successors. Two types of bottom strongly connected components (BSCC)

- the BSCC consists of m_1
- a BSCC from which one cannot reach m_1

Probabilistic Regular Petri Nets

Sketch of Proof (continued)

A finite graph is built as follows, suppose $A = \{m_1\}$

- Push on the stack $\mathbf{m}_0$.
- While the stack is not empty, pop from the stack some marking $\mathbf{m}$. Compute the set of transition firings $\mathbf{m} \xrightarrow{t} \mathbf{m}'$. Push on the stack $\mathbf{m}'$ if:
  1. $\mathbf{m}'$ is not already present in the graph,
  2. and $\mathbf{m}' \neq \mathbf{m}_1$,
  3. and for all $p \in P$, $\mathbf{m}_1(p) + B(\mathcal{N}, \mathbf{m}_0) \geq \mathbf{m}'(p)$.

Due to the third condition, this algorithm terminates. On the finite graph, one keeps the weights and adds loops for states without successors. Two types of bottom strongly connected components (BSCC)

- the BSCC consists of m_1
- a BSCC from which one cannot reach m_1

As one can reach some BSCC almost surely, thus the net is decisive w.r.t. m_1.

Plan

Conclusion

Contributions

- Study the relationship between decisiveness and recurrent;
- Obtain decidability results of decisiveness w.r.t. subclasses of dynamic probabilistic counter machines;
- Demonstrate decidability results of decisiveness w.r.t. subclasses of dynamic probabilistic Petri nets.

Conclusion

Contributions

- Study the relationship between decisiveness and recurrent;
- Obtain decidability results of decisiveness w.r.t. subclasses of dynamic probabilistic counter machines;
- Demonstrate decidability results of decisiveness w.r.t. subclasses of dynamic probabilistic Petri nets.

Perspectives

- Study the decidability of decisiveness of static pPN w.r.t. arbitrary finite set;
- Establish sufficient conditions for decisiveness for models with undecidability of decisiveness;
- Examine the relationship between two properties: decisiveness and divergence.