Equivalence of μᵖ-Calculus and p-Automata

Claudia Cauli and Nir Piterman {cc488, np183}@le.ac.uk University of Leicester, Dept. of Informatics PhD 1st year

We model stochastic systems through **Markov chains** and specify and evaluate their properties using probabilistic temporal logics and automata. In general, **logics** offer a clearer syntax and **automata** provide better performance in terms of computability. Therefore, it is important to define classes of logics and automata that have the **same expressive power** and can be used interchangeably. Here, we focus on two such formalisms known as **μᵖ-calculus** and **p-automata**.

Background

μᵖ-Calculus

The μᵖ-calculus [1] is a probabilistic temporal logic. Formulas are built up from the combination of:

- **Atomic propositions** $p, \neg p$
- **Boolean connectives** $\vee, \wedge$
- **Next operator** $\bigcirc\varphi$
- **Probabilistic quantification** $[\varphi]_J$
- **Fixpoints** $\mu X.\varphi,\ \nu X.\varphi$

Using the fixpoint operators this logic can express *finite* and *infinite* iterations of properties:

Least fixpoint μ Finitely many iterations
Greatest fixpoint ν Infinitely many iterations

Formulas φ contained inside a probabilistic quantification are associated with a probability value in [0,1]. The operator $[\cdot]_J$ checks whether the value of the formula meets the **bound J** (of the form $\geq x$ or $> x$), and gets the value 1 or 0 accordingly. Therefore, top-level formulas are *qualitative*: either *true* or *false*.

When a μᵖ-calculus formula is true on a Markov chain, we say that *the Markov chain **satisfies** the formula*.

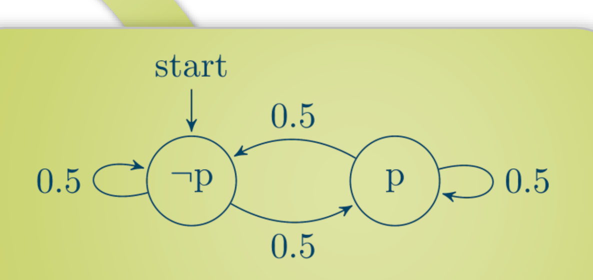

Fig. 1. Markov chain.

$[\bigcirc p]_{\geq 0.5}$ *The probability of reaching p in one step is* ≥ 0.5
$[\bigcirc\bigcirc p]_{\geq 0.5}$ *The probability of reaching p in two steps is* ≥ 0.5
$[p \vee \bigcirc p]_{\geq 0.5}$ *The probability of p either now or in one step is* ≥ 0.5

Fig. 2. μᵖ-Calculus formulas true on the Markov chain of Fig. 1.

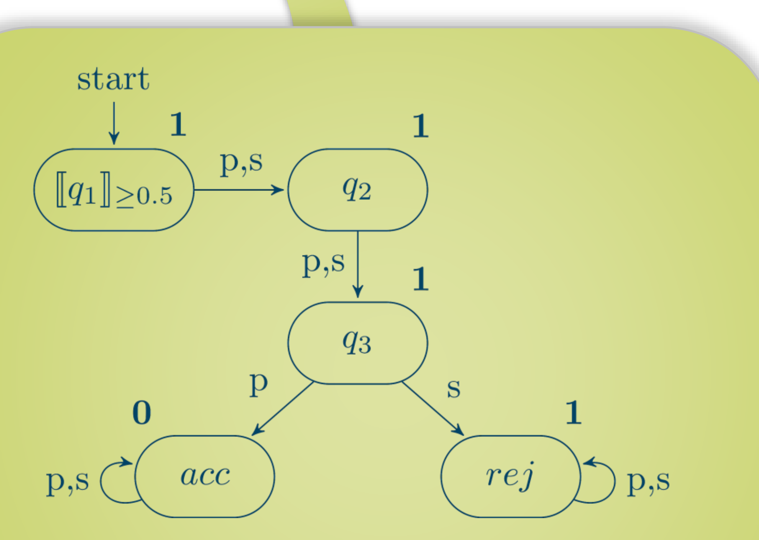

Fig. 3. A graph representing a p-automaton with: alphabet { p,s }, states { q1, q2, q3, acc, rej }, transitions=edges, initial condition ⟦ q1 ⟧≥0.5.

Markov Chains

A Markov chain is a probabilistic transition system defined by the four components:

1) Set of **Locations**; 2) **Initial** location; 3) **Probability** function; 4) **Labelling** function.

The probability of moving from one location to each of its successors is a number in **[0,1]**. The sum of probabilities over all successors must be equal to 1.

p-Automata

A p-automaton [2] is an automaton that *reads a Markov chain as input and decides whether to **accept** it or not*. It is characterised by five components:

1. **States** are the elementary blocks and, to handle probabilities, may be enclosed in a *probabilistic quantification* $[\![\cdot]\!]_J$.
2. **Alphabet** contains *symbols* that are read by the automaton, triggering a specific transition.
3. **Transitions** allow the automaton to move from one state to a *Boolean (and/or) combination* of them, depending on the symbol read.
4. **Initial condition** is a state, or a *combination* thereof, from which the automaton begins its computation.
5. **Acceptance** assigns a *number* to each state. Only states that are marked by an *even* number can be visited *infinitely often*.

Equivalence

μᵖ-Calculus → p-Automata

For every μᵖ-calculus formula we can construct a p-automaton that accepts exactly those Markov chains that satisfy the formula [3].

The components of the automaton resulting from the conversion are:

1. **States** originate from *sub-formulas* of the form: propositions, negated propositions, next, and quantified next; plus *accepting* and *rejecting* states.
2. **Alphabet** is the powerset of *propositions* appearing in the formula.
3. **Transitions** preserve the *Boolean connectives* ($\vee, \wedge$) and unfold the *next* operators into their nested sub-formulas.
4. **Initial condition** derives from the main formula without fixpoints.
5. **Acceptance** reflects the *type* of fixpoints that enclose the sub-formula/state ($\mu \leftrightarrow$ odd, $\nu \leftrightarrow$ even) and their potential *nesting*. Accepting and rejecting states are assigned numbers 0 and 1, respectively.

$\mu X.(p \vee [\bigcirc X]_{\geq 0.5})$

"Eventually reach a p within single steps whose probability is ≥ 0.5"

Fig. 4. μᵖ-Calculus formula that either reads a p or performs a step and starts again. Since μ allows finitely many repetitions, a p must be reached eventually.

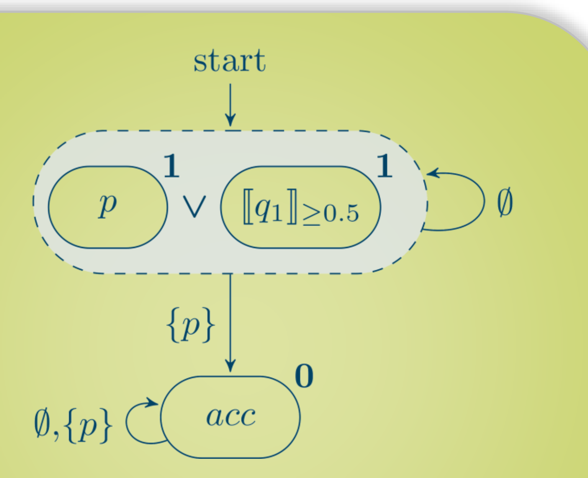

Fig. 5. p-automaton that either reads a p or checks again. Since the topmost states can be visited only finitely many times, a p must be read eventually.

p-Automata → μᵖ-Calculus

For every p-automaton we can construct a μᵖ-calculus formula satisfied in exactly those Markov chains accepted by the automaton [3].

The translation exploits the parallel between components of the automaton and elements of μᵖ-calculus formulas:

- **Propositions** are taken from the *alphabet*.
- **Boolean connectives** match the and/or combinations of states defined by the *transitions* and *initial condition*.
- **Next** reflects the automaton's *transitions*.
- **Probabilistic quantification** is placed corresponding to *bounded states* of the automaton.
- **Fixpoints** are decided by looking at those states that are *visited indefinitely* and their acceptance *number*.

We have summarised the analogies that allow the **translation** from μᵖ-calculus to p-automata and backwards. The mutual correspondence of the two languages implies their **equivalence** in expressive power; thus, lifting the well-known connection between logics and automata theory to a **probabilistic** scenario.

References

[1] Castro P., Kilmurray C., and Piterman N., *Tractable Probabilistic μ-calculus that Expresses Probabilistic Temporal Logics*, 2015
[2] Chatterjee K. and Piterman N., *Obligation Blackwell Games and p-Automata*, 2017
[3] Cauli C. and Piterman N., *Equivalence of μᵖ-Calculus and p-Automata*, 2017

PDF