Model Checking Markov Chains against AFW specifications

Linus Cooper[1], Sasha Rubin[1], Moshe Vardi[2]

[1] University of Sydney
[2] Rice University

AAAI Spring Symposium 2023

Context

Model Checking

- Formal verification of a model against a specification.
- M is the model being verified.
- A is a specification of some behaviour M might exhibit.
- Here, M is a Markov chain and A is an alternating automaton on finite words (AFW). M generates infinite words, while A either accepts or rejects finite words.

Problem

How to compare an infinite word against an acceptor of finite words?

- We say that A accepts an infinite word x if A accepts *some prefix* of x.
- We call this the *existential semantics*.
- Now we can ask: what is the probability that M generates a word accepted by A?
- Decision problems: is this probability 0? is it 1?
- Another natural definition (acceptance condition is if *every* prefix is accepted) turns out to be dual to this one.

Characterization

- $P > 0$ if and only if some finite string is accepted by A and can be generated by M.
- $P = 1$ if and only if for every finite string x generated by M, there is some continuation xy generated by M accepted by A.
- Intuitively, since it is never possible to end up in a 'dead' state, one will eventually reach an accepting state with probability 1.

Previous work

- (Vardi 1985) showed that model checking of NBWs (nondeterministic Büchi automata, acceptors of infinite words) is PSPACE-complete.
- (Courcoubetis and Yannakakis 1988) study LTL: also PSPACE-complete.
- (Bustan, Rubin and Vardi 2004) look at ABWs - also PSPACE-complete.
- In all of these cases $P = 0$ and $P = 1$ have the same complexity.

This work

- So AFWs - also PSPACE-complete for both problems?
- Idea - convert AFW to equivalent DFW, then use a straightforward graph traversal
  - Removing alternation causes a double-exponential blowup, for an overall 2EXPTIME (can be improved to EXPSPACE)
- Idea - convert AFW to equivalent ABW (after accounting for existential semantics)
  - This conversion can require an exponential blowup in automaton size, producing an EXPSPACE algorithm.
- It turns out that the existential semantics are crucial and the problem is substantially harder...

Results

- The emptiness problem is PSPACE-complete.
- The universality problem is EXPSPACE-complete.
- Unlike the cases for infinite word specifications where both problems are PSPACE-complete, our problem has an asymmetry causing an exponential gap.
- Intuitively, based on the characterization, the emptiness problem is easier because one only needs to exhibit one string to show $P > 0$ instead of reasoning about all strings.

Proof sketches

- Emptiness in PSPACE: use nondeterminism to guess a word generated by M and accepted by A, showing it is NPSPACE, then apply Savitch's theorem
- Emptiness PSPACE-hard: standard AFW emptiness, which is PSPACE-complete, reduces to probabilistic emptiness.
- Universality in EXPSPACE: convert AFW to ABW or to DFW.
- Universality EXPSPACE-hard: we build an AFW that recognises invalid runs of an EXPSPACE TM. To do so we use a specific encoding of runs that marks each tape cell with its position (in binary). The AFW uses existential choice to select a time t and the k-th cell of the tape. It then uses universal choice to validate that its contents are consistent with the k-th cell at time $t+1$. The end of the trace serves as a "marker" that the AFW uses to synchronize.