Quantitative Model-Checking and Mean-Payoff Expressions

Yaron Velner, Tel Aviv University

Quantitative Model-Checking and Mean-Payoff Expressions

- Talk outline
  - Background on model-checking
  - From Boolean model-checking to quantitative model-checking
  - Quantitative languages
  - Weighted automata
  - The class of mean-payoff expressions
    - Our contribution

Model-Checking

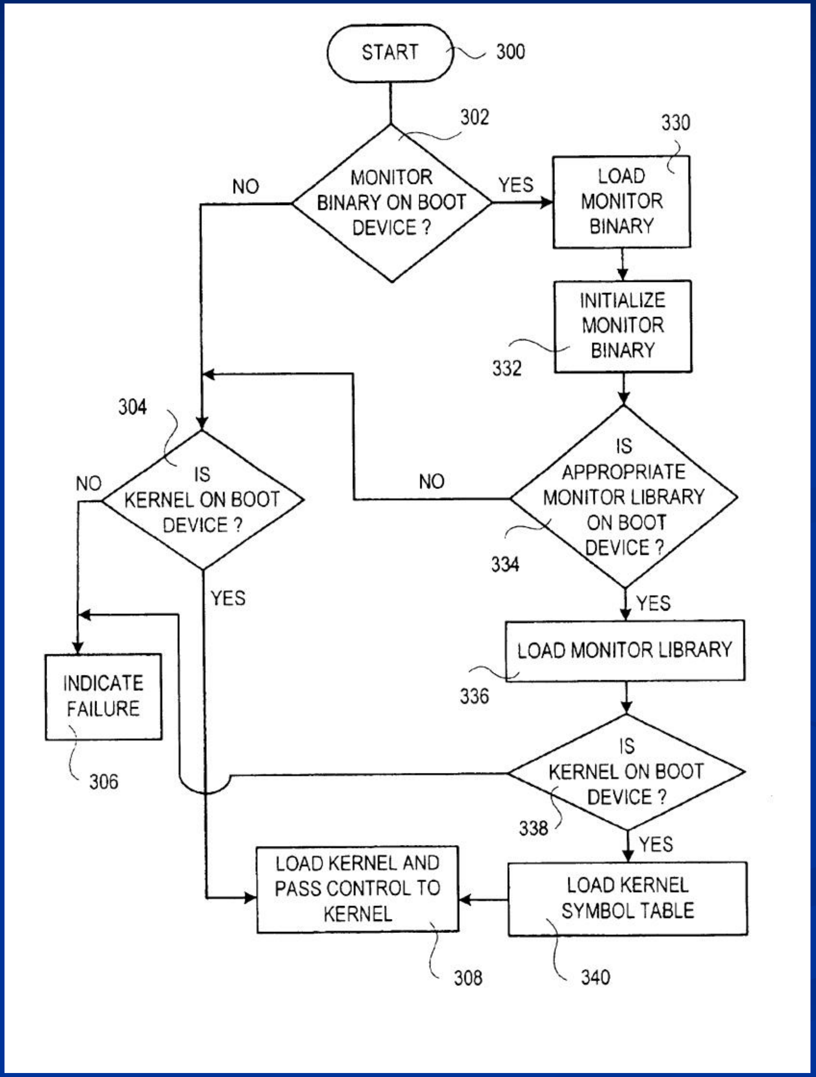
Program

Input

Output

Program

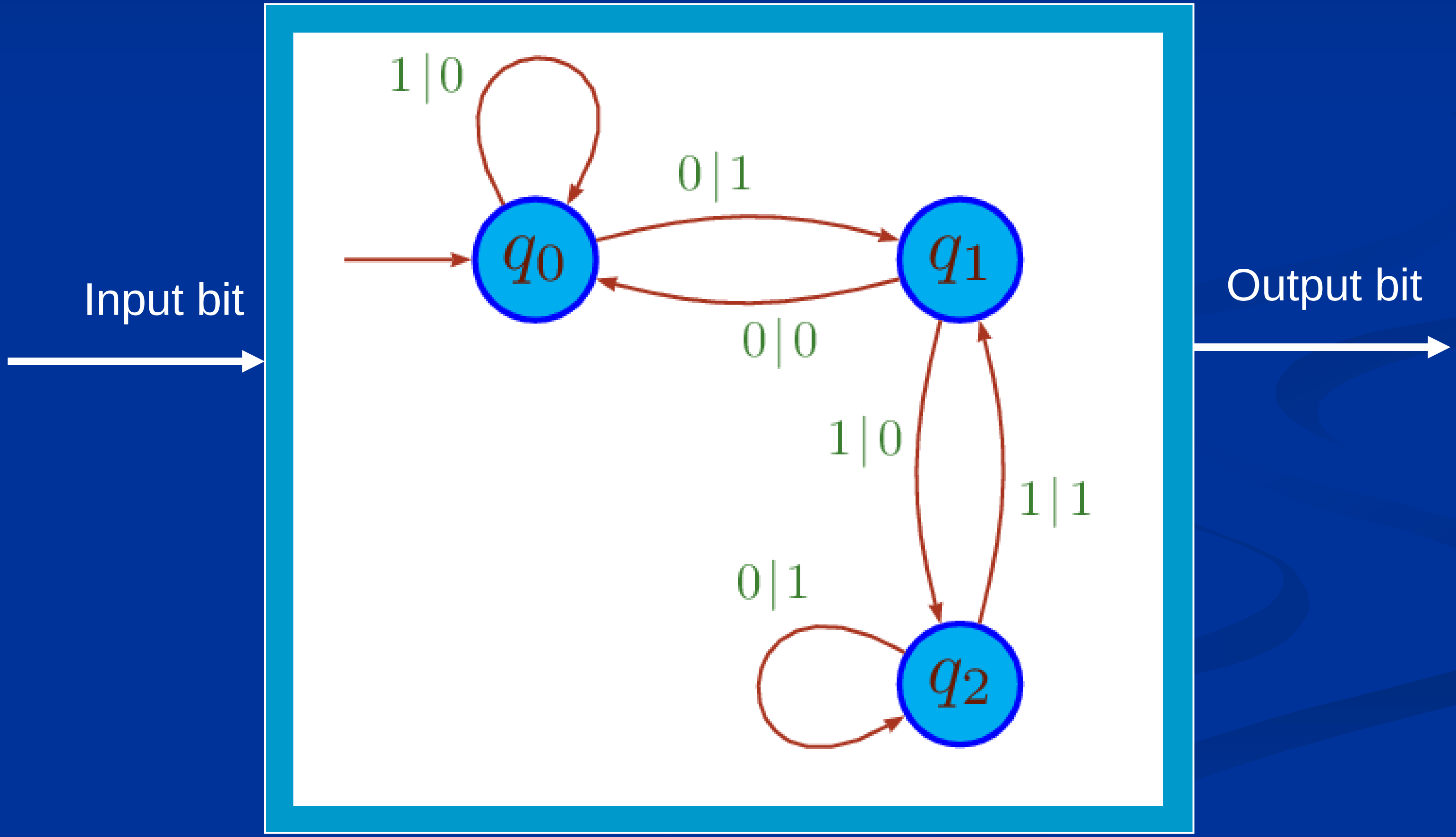
Input



Output

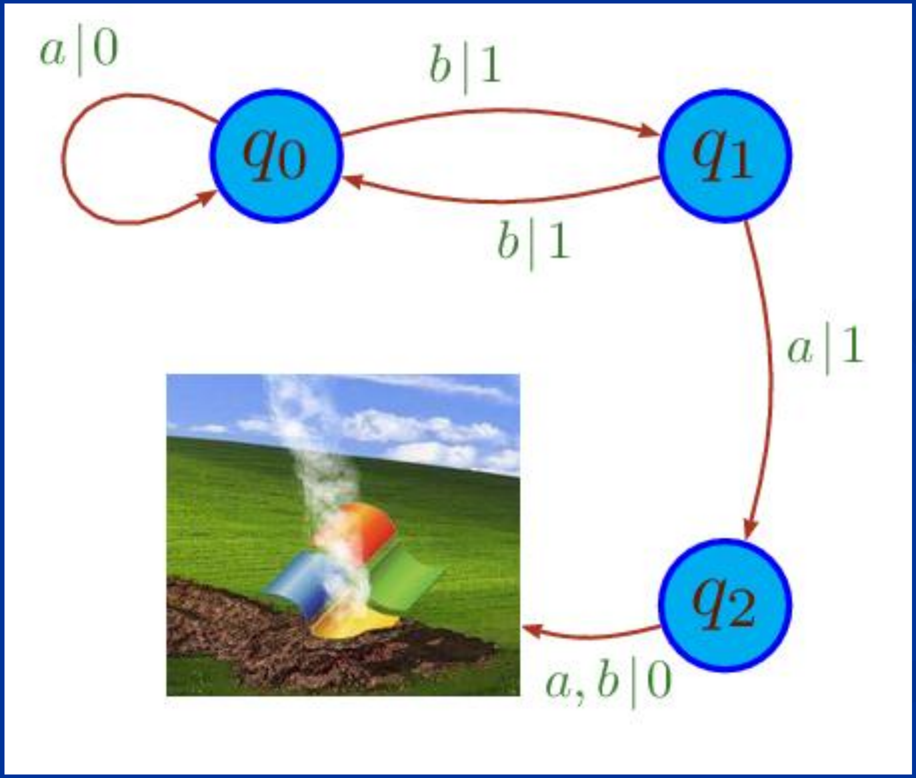
Program

- Finite state machine



Property

- The OS never crashes



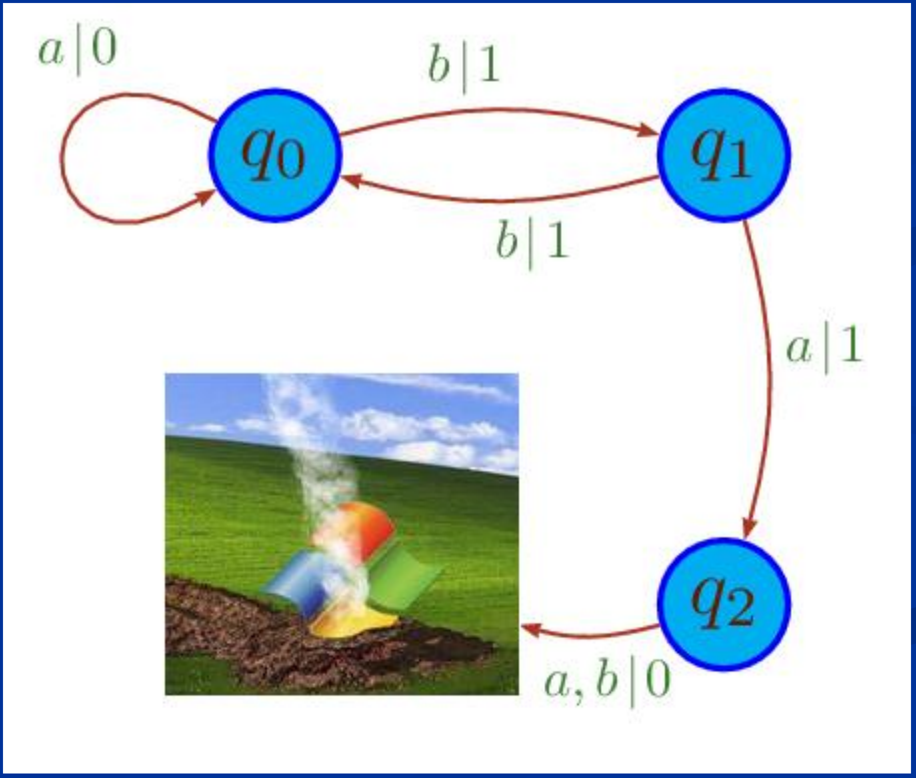
q_2 is never reached

Property

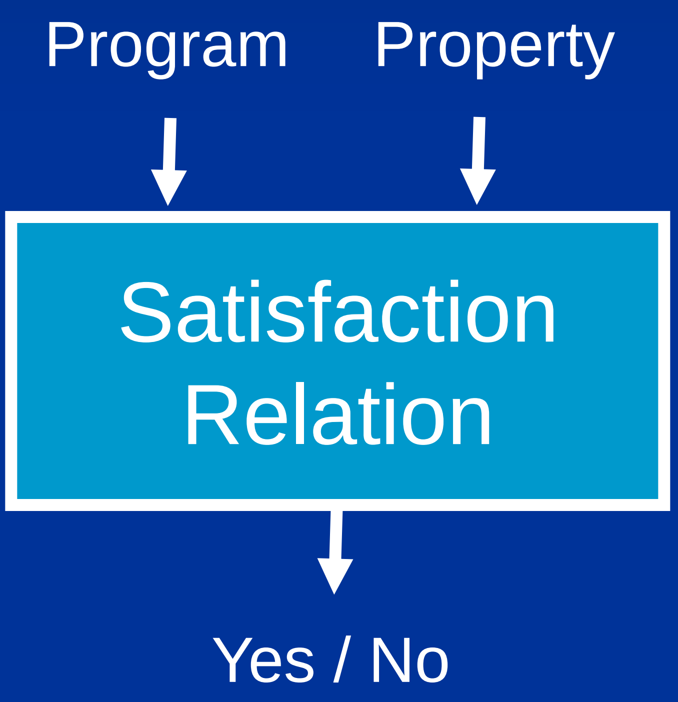
- Property is a set of infinite runs
  - P = "State q_2 is never reached"
    - $q_0\ q_1\ q_0\ (q_0\ q_1)\ (q_0\ q_1)\ (q_0\ q_1)\ldots \in P$
    - $q_0\ q_1\ q_2\ q_3\ (q_3)\ (q_3)\ (q_3)\ldots \notin P$

Language of a Property

- Property + Program = Language
  - P = "State q_2 is never reached"
  - aaaaa… $\in$L
  - abaaa… $\notin$L



Model-Checking



$\Rightarrow$

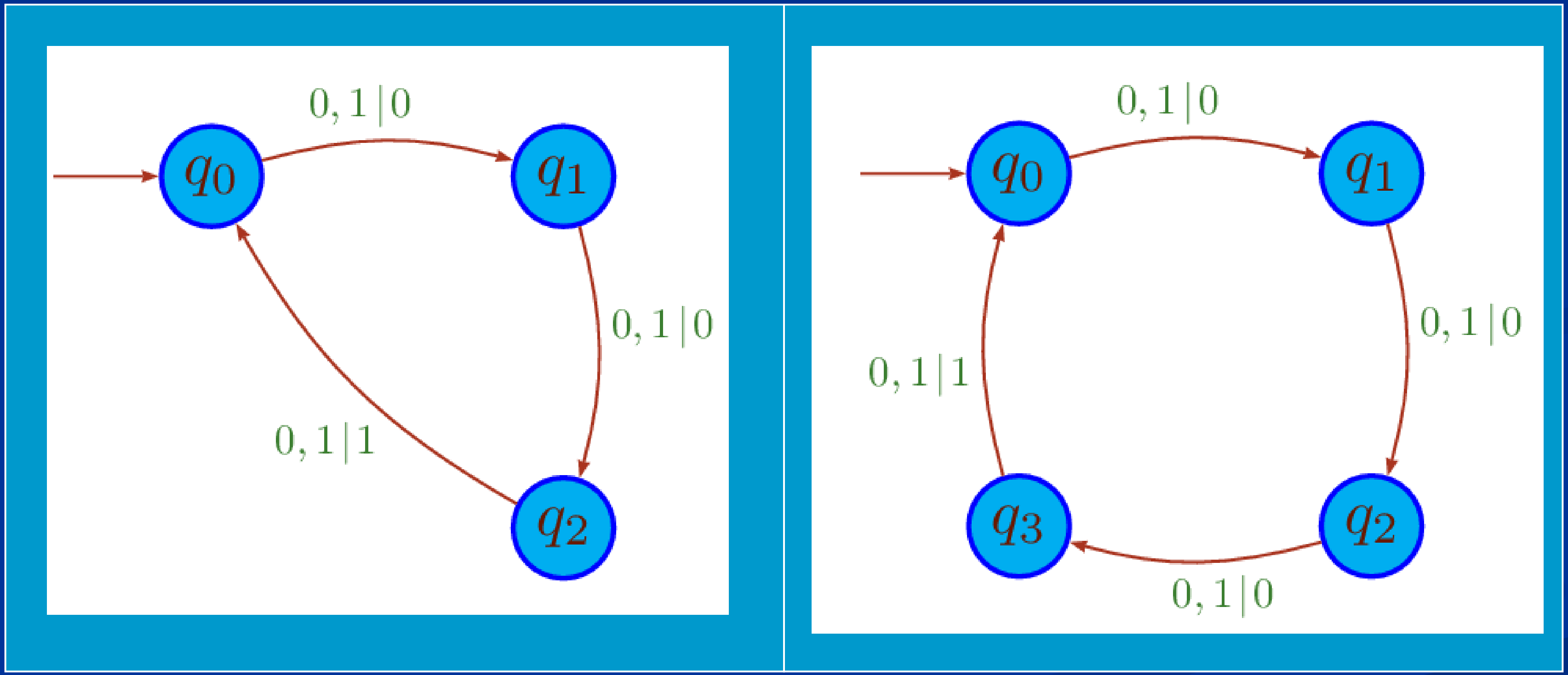
- Property + Program = L
- Is there exists a word $\notin$ L ?

Why is it not enough?

- Property: every website is loaded eventually
- Which implementation is better?

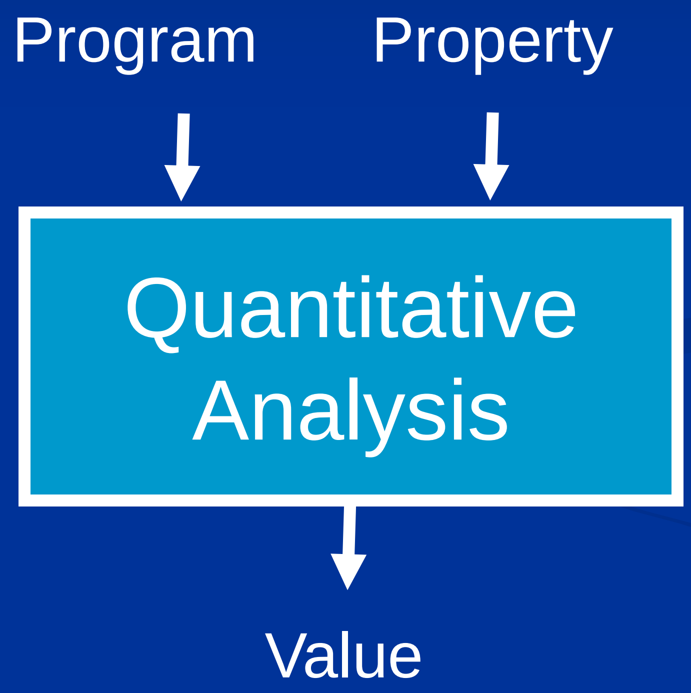
Why is it not enough?

- Property: ‘1’ is always eventually outputted



- Which implementation is better?

Quantitative Model-Checking

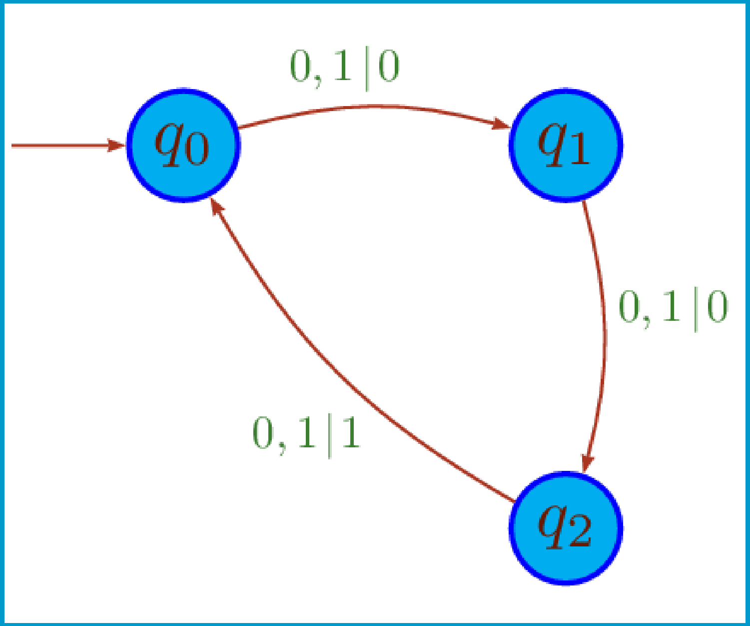


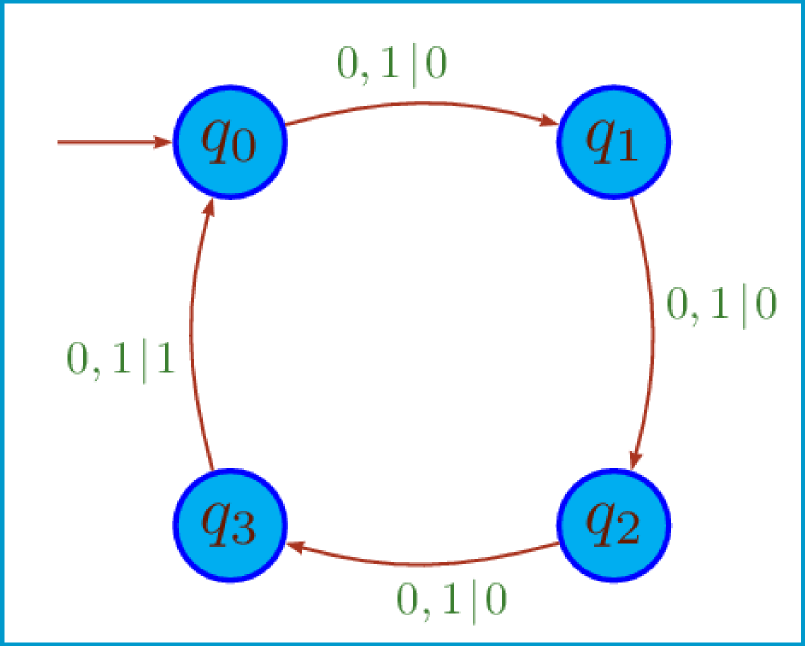
Quantitative Property

- Property : runs $\rightarrow$ R
- Property + Program = Quantitative Language
  - L : infinite words $\rightarrow$ R

Example

- P(run) = - (maximal number of consecutive states that do not output ‘1’)

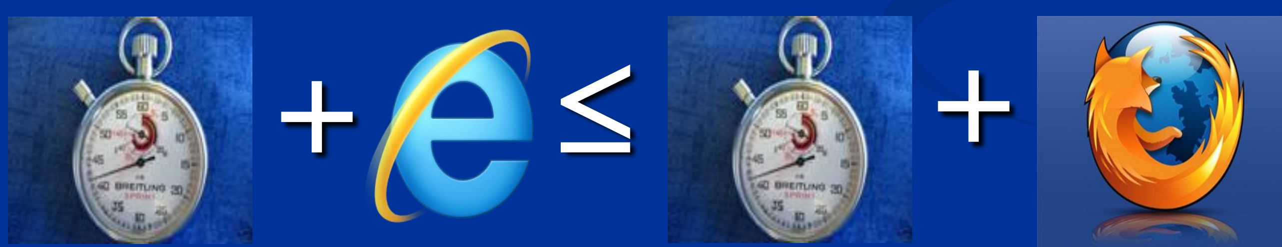




- For every input: $L_2(\text{input}) \leq L_1(\text{input})$

Program Refinement

- Given property P and programs A_1, A_2
  - A_2 refines A_1 if for every input:
    - $(P + A_1)(\text{input}) \leq (P + A_2)(\text{input})$



The quantitative language inclusion problem

- For two quantitative languages L_1 and L_2
  - is $L_1(w) \leq L_2(w)$ for every word w?
- A class of languages is decidable if the inclusion problem is decidable

Closure Operations

- For quantitative languages L_1 and L_2
  - **min**$(L_1, L_2)(w) = \min(L_1(w), L_2(w))$
  - **max**$(L_1, L_2)(w) = \max(L_1(w), L_2(w))$
  - $(-L_1)(w) = 1 - L_1(w)$
  - **sum**$(L_1, L_2)\ (w) = L_1\ (w) + L_2(w)$
- A class of languages is robust if it closed under the above operations

Generalization of Boolean languages

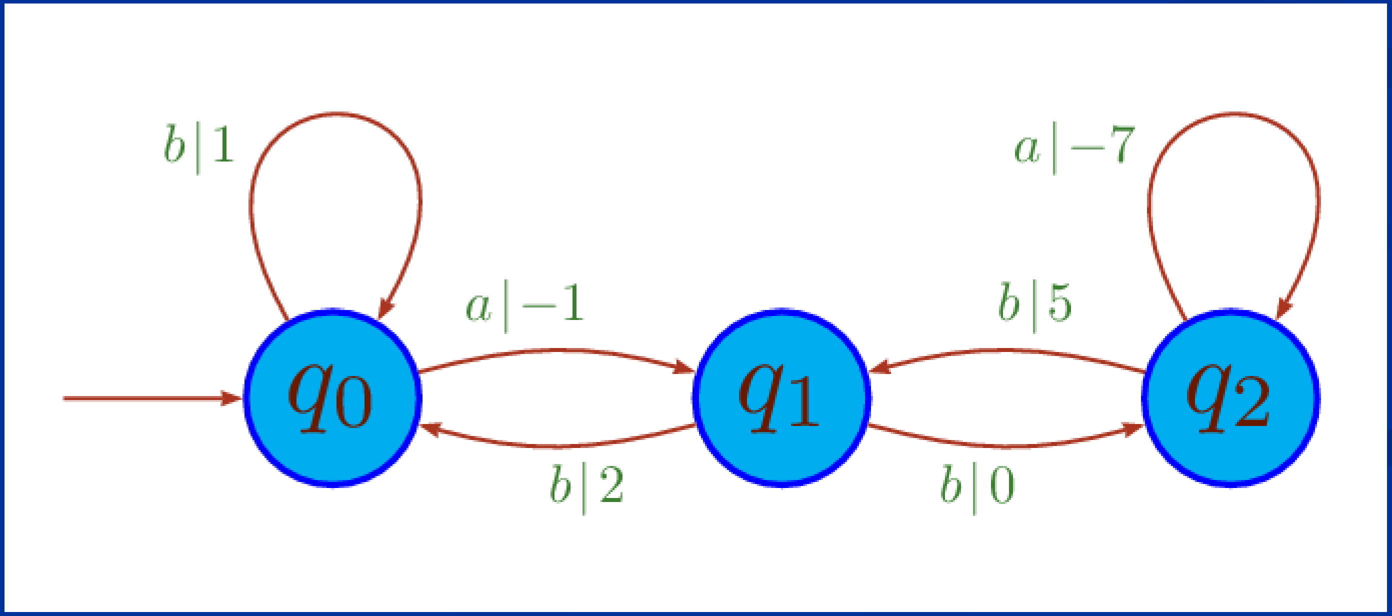
- Boolean language L : infinite word $\rightarrow \{0,1\}$
- $L_1 \cap L_2 = \mathbf{min}(L_1, L_2)$
- $L_1 \cup L_2 = \mathbf{max}(L_1, L_2)$
- $L^c = -L$
- $L_1 \subseteq L_2 \Leftrightarrow L_1 \leq L_2$

Goal

- Find a robust and decidable class of quantitative languages

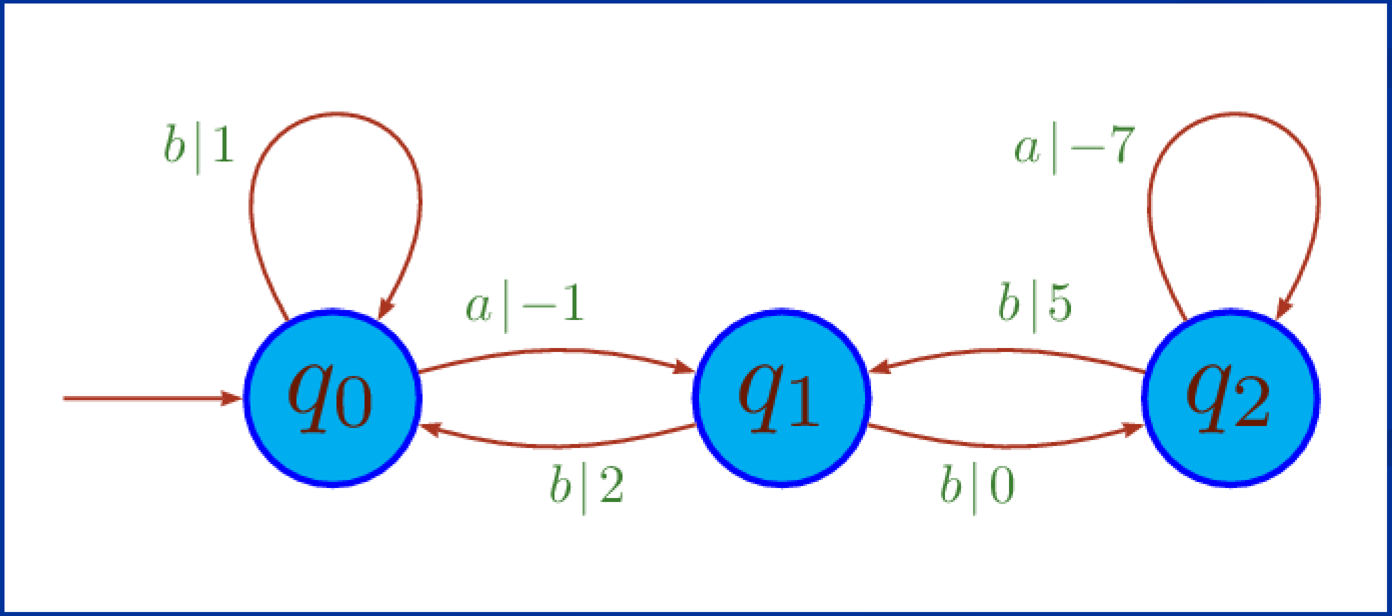
Weighted Automata

- Automaton : word → v_0 v_1 v_2 …



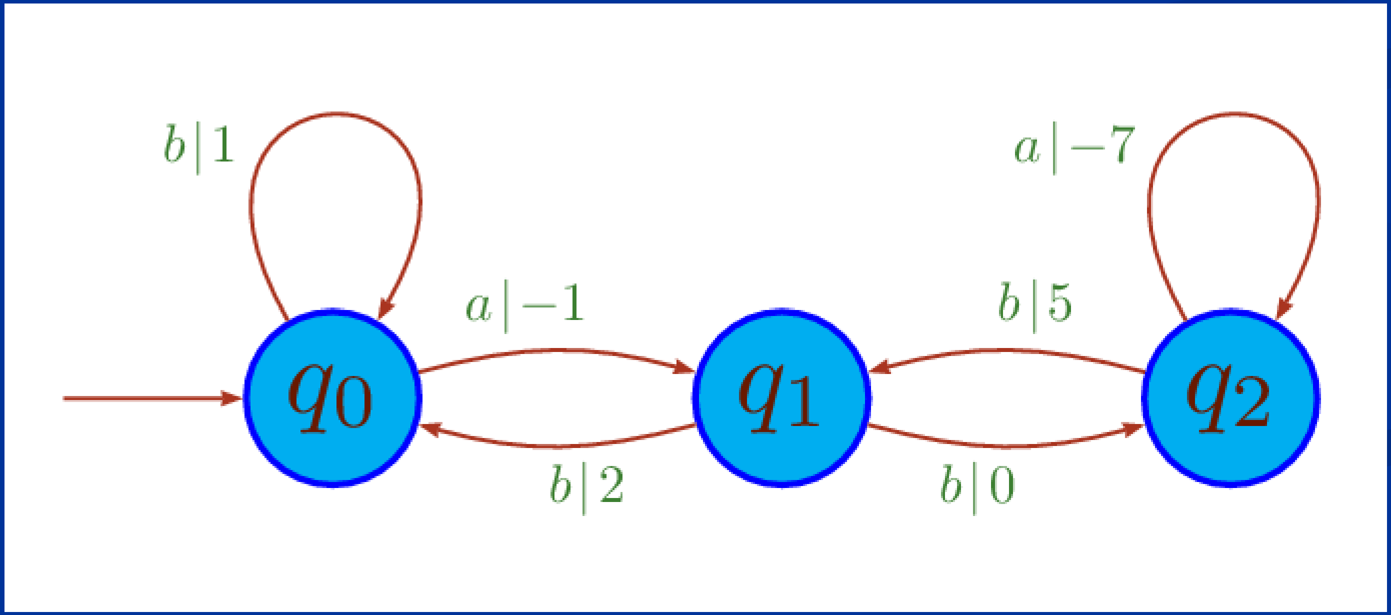
Weighted Automata

- Automaton : word → $v_0\ v_1\ v_2 \ldots$
  - Max automaton: $A(w) = \mathbf{max}\{v_0, v_1, v_2 \ldots\}$



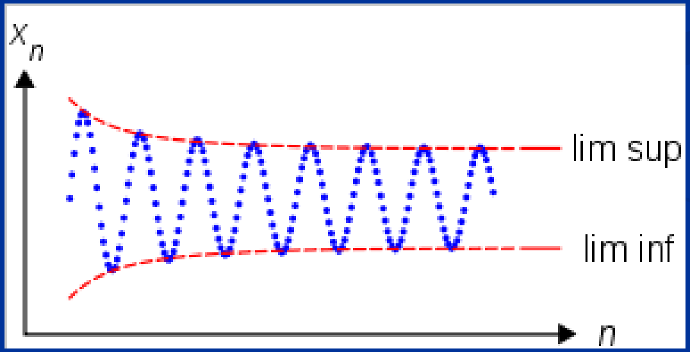
Weighted Automata

- Automaton : word → v_0 v_1 v_2 …
  - LimAvg automaton: A(w) = $\liminf_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} v_i$



Weighted Automata

- Automaton : word $\rightarrow$ v_0 v_1 v_2 …
  - LimAvg automaton: $A(w) = \liminf_{n\rightarrow\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} v_i$



The Class of LimAvg Automata

- Decidable
- Not robust

Mean-payoff expressions

- Chatterjee, Doyen, Edelsbrunner, Henzinger and Rannou 2010:
  - E := LimAvg(A) | **min**(E,E) | **max**(E,E) | **sum**(E,E) | -E
  - Example:
    - **min**(LimAvg(A_1), LimAvg(A_2)) + **max**(LimAvg(A_3), -LimAvg(A_4))
- This class is robust
- Is it decidable?

Mean-payoff expressions

- Chatterjee, Doyen, Edelsbrunner, Henzinger and Rannou 2010:
  - The class of mean-payoff expressions is decidable
  - The inclusion problem $\in$ 4EXPTIME $(2^{2^{2^{2^{poly(n)}}}})$

Mean-payoff expressions

- Chatterjee, Doyen, Edelsbrunner, Henzinger and Rannou 2010:
  - The class of mean-payoff expressions is decidable
  - The inclusion problem $\in$ 4EXPTIME $(2^{2^{2^{2^{poly(n)}}}})$
- Our contribution:
  - The inclusion problem is PSPACE complete

Thanks