Power of Randomization in Automata on Infinite Strings

A. Prasad Sistla

October 12, 2009

Joint work with Rohit Chadha and Mahesh Viswanathan

OutLine

Automata on Infinite Strings

Probabilistic Automata on Finite Strings

Probabilistic Buchi Automata

Finite State Probabilistic Monitors

Expressiveness results for FPMS

Decidability and Complexity results for FPMs

Decidability and Expressiveness for PBAs

Hierarchical PBAs

Automata on Infinite Strings (Rabin, Buchi, Muller, McNaughton)

- Finite State Automata on Finite Strings:
  DFA $\equiv$ *NFA* $\equiv$ *RExprs*

Automata on Infinite Strings (Rabin, Buchi, Muller, McNaughton)

- Finite State Automata on Finite Strings:
  $DFA \equiv NFA \equiv RExprs$
- Buchi Automata (BA) on Infinite Strings:
  $\mathcal{A} = (\Sigma, Q, \delta, q_0, F)$
  - Σ — input alphabet
  - Q — automaton states
  - $\delta \subseteq Q \times \Sigma \times q$ — transition relation
  - q_0 — start state
  - $F \subseteq Q$ — accepting set

Buchi Automata

- A run r on an infinite input $a = a_0, ...a_i, ...$ is a sequence $(r_0, ..., r_i, ...)$ such that $r_0 = q_0$ and $(r_i, a_i, r_{i+1}) \in \delta$ for $i \geq 0$.

Buchi Automata

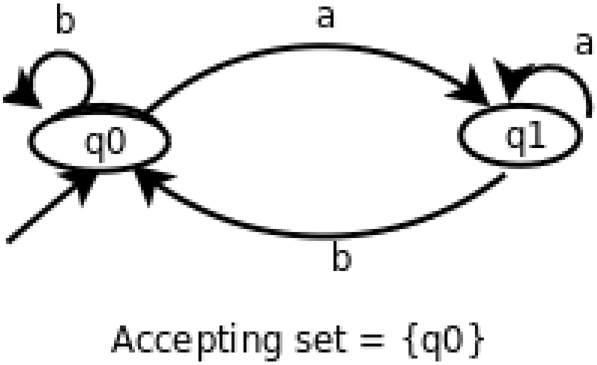
- A run r on an infinite input $a = a_0, ...a_i, ...$ is a sequence $(r_0, ..., r_i, ...)$ such that $r_0 = q_0$ and $(r_i, a_i, r_{i+1}) \in \delta$ for $i \geq 0$.
- r is accepting if some accepting state appears infinitely often.

Buchi Automata

- A run r on an infinite input $a = a_0, ...a_i, ...$ is a sequence $(r_0, ..., r_i, ...)$ such that $r_0 = q_0$ and $(r_i, a_i, r_{i+1}) \in \delta$ for $i \geq 0$.
- r is accepting if some accepting state appears infinitely often.
- $L(\mathcal{A})$ is the set of inputs on which $\mathcal{A}$ has an accepting run.

Buchi Automata

- A run r on an infinite input $a = a_0, \ldots a_i, \ldots$ is a sequence $(r_0, \ldots, r_i, \ldots)$ such that $r_0 = q_0$ and $(r_i, a_i, r_{i+1}) \in \delta$ for $i \geq 0$.
- r is accepting if some accepting state appears infinitely often.
- $L(\mathcal{A})$ is the set of inputs on which $\mathcal{A}$ has an accepting run.
- **An example BA:** Accepts strings in which every a is eventually followed by b.

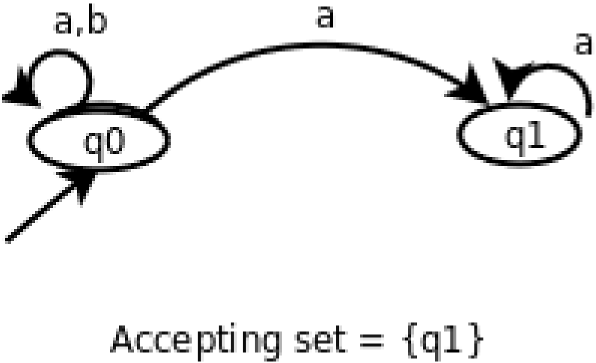


NonDet BA Vs. Det BA

NonDet BA is more powerful than Det BA. The following BA has no equivalent Det BA.

NonDet BA Vs. Det BA

NonDet BA is more powerful than Det BA. The following BA has no equivalent Det BA.



Accepts strings in which eventually only *a* appears.

Rabin Automata

- Rabin Automaton (RA): the acceptance set is of the form $\{(L_1, U_1), ..., (L_k, U_k)\}$ where $L_i, U_i \subseteq Q$.

Rabin Automata

- Rabin Automaton (RA): the acceptance set is of the form $\{(L_1, U_1), ..., (L_k, U_k)\}$ where $L_i, U_i \subseteq Q$.
- A run r is accepting if for some i, no state in L_i appears infinitely often and some state in U_i appears infinitely often.

Rabin Automata

- Rabin Automaton (RA): the acceptance set is of the form $\{(L_1, U_1), ..., (L_k, U_k)\}$ where $L_i, U_i \subseteq Q$.
- A run r is accepting if for some i, no state in L_i appears infinitely often and some state in U_i appears infinitely often.

- *DetRA* $\equiv$ *NonDetRA* $\equiv$ *NonDetBA*
- *DetRA* $\equiv \omega$-Regular languages

Rabin Automata

- Rabin Automaton (RA): the acceptance set is of the form $\{(L_1, U_1), ..., (L_k, U_k)\}$ where $L_i, U_i \subseteq Q$.
- A run r is accepting if for some i, no state in L_i appears infinitely often and some state in U_i appears infinitely often.

- $DetRA \equiv NonDetRA \equiv NonDetBA$
- $DetRA \equiv \omega$-Regular languages
- Det-ω-Regular langs = Langs recognized by Det BA.
- Fact: ω-Regular Langs = Boolean closure of Det-ω-Regular Langs.

Rabin Automata

- Rabin Automaton (RA): the acceptance set is of the form $\{(L_1, U_1), ..., (L_k, U_k)\}$ where $L_i, U_i \subseteq Q$.
- A run r is accepting if for some i, no state in L_i appears infinitely often and some state in U_i appears infinitely often.

- $DetRA \equiv NonDetRA \equiv NonDetBA$
- $DetRA \equiv \omega$-Regular languages
- Det-ω-Regular langs = Langs recognized by Det BA.
- Fact: ω-Regular Langs = Boolean closure of Det-ω-Regular Langs.
- **Applications:** Verification of concurrent programs, decision procedures for logics: Linear Temporal Logic, S1S, Theory of Linear order, etc.

Decision Problems

- Emptiness: Given BA $\mathcal{A}$, check if $L(\mathcal{A}) \neq \emptyset$. It is in NLOGSPACE and hence is in P.

Decision Problems

- Emptiness: Given BA $\mathcal{A}$, check if $L(\mathcal{A}) \neq \emptyset$. It is in NLOGSPACE and hence is in P.
- Universality; Given a BA $\mathcal{A}$, check if $L(\mathcal{A}) = \Sigma^\omega$. It is **PSPACE**-complete.

Decision Problems

- Emptiness: Given BA $\mathcal{A}$, check if $L(\mathcal{A}) \neq \emptyset$. It is in NLOGSPACE and hence is in P.
- Universality; Given a BA $\mathcal{A}$, check if $L(\mathcal{A}) = \Sigma^\omega$. It is **PSPACE**-complete.
- Language Containment; Given a BAs $\mathcal{A}, \mathcal{B}$, check if $L(\mathcal{A}) \subseteq L(\mathcal{B})$. **PSPACE**-complete.

OutLine

Automata on Infinite Strings

Probabilistic Automata on Finite Strings

Probabilistic Buchi Automata

Finite State Probabilistic Monitors

Expressiveness results for FPMS

Decidability and Complexity results for FPMs

Decidability and Expressiveness for PBAs

Hierarchical PBAs

Probabilistic Finite string Automata (PFA) (Rabin)

- $\mathcal{A} = (\Sigma, Q, \delta, q_0, F)$.
- $\delta : Q \times \Sigma \times Q \rightarrow [0, 1]$ so that $\sum_{q'} \delta(q, a, q') = 1$.

Probabilistic Finite string Automata (PFA) (Rabin)

- $\mathcal{A} = (\Sigma, Q, \delta, q_0, F)$.
- $\delta : Q \times \Sigma \times Q \rightarrow [0, 1]$ so that $\sum_{q'} \delta(q, a, q') = 1$.
- For $a \in \Sigma^*$, $PrOfAcc_{\mathcal{A}}(a)$, called the probability of acceptance of a— is the probability that $\mathcal{A}$ is in some state in F after the input a.

Probabilistic Finite string Automata (PFA) (Rabin)

- $\mathcal{A} = (\Sigma, Q, \delta, q_0, F)$.
- $\delta : Q \times \Sigma \times Q \to [0, 1]$ so that $\sum_{q'} \delta(q, a, q') = 1$.
- For $a \in \Sigma^*$, $PrOfAcc_{\mathcal{A}}(a)$, called the probability of acceptance of a— is the probability that $\mathcal{A}$ is in some state in F after the input a.
- $L_{>\frac{1}{2}}(\mathcal{A}) = \{a \in \Sigma^* : Pr(a) > \frac{1}{2}\}$.

Probabilistic Finite string Automata (PFA) (Rabin)

- $\mathcal{A} = (\Sigma, Q, \delta, q_0, F)$.
- $\delta : Q \times \Sigma \times Q \to [0, 1]$ so that $\sum_{q'} \delta(q, a, q') = 1$.
- For $a \in \Sigma^*$, $PrOfAcc_{\mathcal{A}}(a)$, called the probability of acceptance of a— is the probability that $\mathcal{A}$ is in some state in F after the input a.
- $L_{>\frac{1}{2}}(\mathcal{A}) = \{a \in \Sigma^* : Pr(a) > \frac{1}{2}\}$.
- $L_{>\frac{1}{2}}(\mathcal{A})$ can be a non-regular set (Rabin 1960s).

Probabilistic Finite string Automata (PFA) (Rabin)

- $\mathcal{A} = (\Sigma, Q, \delta, q_0, F)$.
- $\delta : Q \times \Sigma \times Q \rightarrow [0,1]$ so that $\sum_{q'} \delta(q, a, q') = 1$.
- For $a \in \Sigma^*$, $PrOfAcc_{\mathcal{A}}(a)$, called the probability of acceptance of a— is the probability that $\mathcal{A}$ is in some state in F after the input a.
- $L_{>\frac{1}{2}}(\mathcal{A}) = \{a \in \Sigma^* : Pr(a) > \frac{1}{2}\}$.
- $L_{>\frac{1}{2}}(\mathcal{A})$ can be a non-regular set (Rabin 1960s).
- Determining non-emptiness of $L_{>\frac{1}{2}}(\mathcal{A})$ and $L_{\geq\frac{1}{2}}(\mathcal{A})$ are undecidable. Both are **R.E.**-complete. (Paz 1971, Soloma 1973).

OutLine

Automata on Infinite Strings

Probabilistic Automata on Finite Strings

Probabilistic Buchi Automata

Finite State Probabilistic Monitors

Expressiveness results for FPMS

Decidability and Complexity results for FPMs

Decidability and Expressiveness for PBAs

Hierarchical PBAs

Probabilistic Buchi Automata (PBA) (Baier et al 2007)

- $\mathcal{A} = (\Sigma, Q, \delta, q_0, F)$. δ defines probabilities on transitions as in the case of PFAs.
- Consider $a \in \Sigma^\omega$.

Probabilistic Buchi Automata (PBA) (Baier et al 2007)

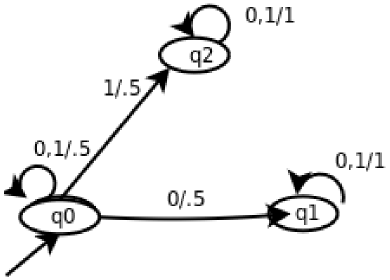
- $\mathcal{A} = (\Sigma, Q, \delta, q_0, F)$. δ defines probabilities on transitions as in the case of PFAs.
- Consider $a \in \Sigma^\omega$.
- Let $Inf(F) \subseteq Q^\omega$ be the set of sequences having some state of F appearing infinitely often.

Probabilistic Buchi Automata (PBA) (Baier et al 2007)

- $\mathcal{A} = (\Sigma, Q, \delta, q_0, F)$. δ defines probabilities on transitions as in the case of PFAs.
- Consider $a \in \Sigma^\omega$.
- Let $Inf(F) \subseteq Q^\omega$ be the set of sequences having some state of F appearing infinitely often.
- Define $PrOfAcc_{\mathcal{A}}(a)$ to be the probability that a run of $\mathcal{A}$ on a is in $Inf(F)$.

Probabilistic Buchi Automata (PBA) (Baier et al 2007)

- $\mathcal{A} = (\Sigma, Q, \delta, q_0, F)$. δ defines probabilities on transitions as in the case of PFAs.
- Consider $a \in \Sigma^\omega$.
- Let $Inf(F) \subseteq Q^\omega$ be the set of sequences having some state of F appearing infinitely often.
- Define $PrOfAcc_{\mathcal{A}}(a)$ to be the probability that a run of $\mathcal{A}$ on a is in $Inf(F)$.
- Example. $\Sigma = \{0, 1\}$.



Accepting set = {q0, q1}

Formal defn. of Acceptance Probability

- Fix $a \in \Sigma^\omega$. Define the probability space (Q^ω, Δ, ϕ) where Δ is the event space and ϕ is the probability measure on it.

Formal defn. of Acceptance Probability

- Fix $a \in \Sigma^\omega$. Define the probability space (Q^ω, Δ, ϕ) where Δ is the event space and ϕ is the probability measure on it.

- Δ is the σ-algebra generated by the sets $S_u = uQ^\omega$ for $u \in Q^*$.

Formal defn. of Acceptance Probability

- Fix $a \in \Sigma^\omega$. Define the probability space (Q^ω, Δ, ϕ) where Δ is the event space and ϕ is the probability measure on it.
- Δ is the σ-algebra generated by the sets $S_u = uQ^\omega$ for $u \in Q^*$.
- Definition of ϕ: Fix any $u = (r_0, .., r_m)$.

Formal defn. of Acceptance Probability

- Fix $a \in \Sigma^\omega$. Define the probability space (Q^ω, Δ, ϕ) where Δ is the event space and ϕ is the probability measure on it.

- Δ is the σ-algebra generated by the sets $S_u = uQ^\omega$ for $u \in Q^*$.

- Definition of ϕ: Fix any $u = (r_0, .., r_m)$.
  If $r_0 = q_0$ then
  $\phi(S_u) = \delta(r_0, a_0, r_1,) \times ...\delta(r_i, a_i, r_{i+1}) \times ...\delta(r_m, a_m, r_{m+1})$.

Formal defn. of Acceptance Probability

- Fix $a \in \Sigma^\omega$. Define the probability space (Q^ω, Δ, ϕ) where Δ is the event space and ϕ is the probability measure on it.
- Δ is the σ-algebra generated by the sets $S_u = uQ^\omega$ for $u \in Q^*$.
- Definition of ϕ: Fix any $u = (r_0, .., r_m)$.
  If $r_0 = q_0$ then
  $\phi(S_u) = \delta(r_0, a_0, r_1,) \times ...\delta(r_i, a_i, r_{i+1}) \times ...\delta(r_m, a_m, r_{m+1})$.
  If $r_0 \neq q_0$ then $\phi(S_u) = 0$.

Formal defn. of Acceptance Probability

- Fix $a \in \Sigma^\omega$. Define the probability space (Q^ω, Δ, ϕ) where Δ is the event space and ϕ is the probability measure on it.
- Δ is the σ-algebra generated by the sets $S_u = uQ^\omega$ for $u \in Q^*$.
- Definition of ϕ: Fix any $u = (r_0, .., r_m)$.
  If $r_0 = q_0$ then
  $\phi(S_u) = \delta(r_0, a_0, r_1,) \times ...\delta(r_i, a_i, r_{i+1}) \times ...\delta(r_m, a_m, r_{m+1})$.
  If $r_0 \neq q_0$ then $\phi(S_u) = 0$.
- $PrOfAcc_{\mathcal{A}}(a) = \phi(Inf(F))$. (Note $Inf(F) \in \Delta$).

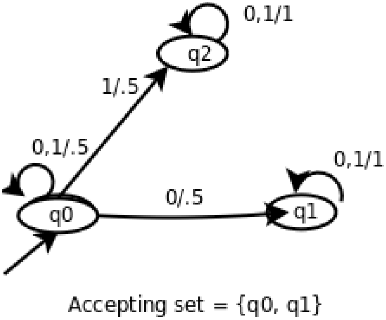
OutLine

Finite State Probabilistic Monitors (FPM)

- A FPM is a PBA with a designated **reject** state, an **absorbing** state. All other states are accepting states. Let $PrOfRej_{\mathcal{A}}(a) = 1 - PrOfAcc_{\mathcal{A}}(a)$.

Finite State Probabilistic Monitors (FPM)

- A FPM is a PBA with a designated **reject** state, an **absorbing** state. All other states are accepting states. Let $PrOfRej_{\mathcal{A}}(a) = 1 - PrOfAcc_{\mathcal{A}}(a)$.

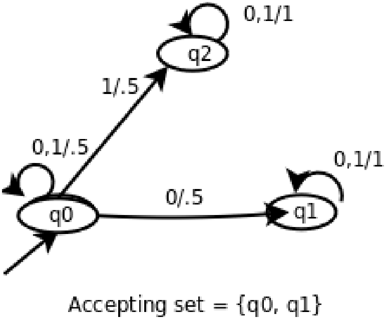


Automaton B

- Any input $a \in \{0,1\}^\omega$ is the binary representation of a number $val(a) \in [0,1]$.

Finite State Probabilistic Monitors (FPM)

- A FPM is a PBA with a designated **reject** state, an **absorbing** state. All other states are accepting states. Let $PrOfRej_{\mathcal{A}}(a) = 1 - PrOfAcc_{\mathcal{A}}(a)$.



Automaton B

- Any input $a \in \{0,1\}^{\omega}$ is the binary representation of a number $val(a) \in [0,1]$.
- Observe $PrOfRej_B(a) = val(a)$.

Applications

- As monitors for monitoring safety as well as some liveness properties.

Applications

- As monitors for monitoring safety as well as some liveness properties.
- Modeling open systems that can fail.

Applications

- As monitors for monitoring safety as well as some liveness properties.
- Modeling open systems that can fail.
- Model checking safety properties of open finite state probabilistic programs.

Properties of Infinite Executions

- Language $C \subseteq \Sigma^\omega$ is a safety property, if it is limit closed. That is, for any $a \in \Sigma^\omega$, if $prefixes(a) \subseteq (Prefixes(C))$ then $a \in C$.
  Example: Set of sequences in which every 1 is preceded by 0.

Properties of Infinite Executions

- Language $C \subseteq \Sigma^\omega$ is a safety property, if it is limit closed. That is, for any $a \in \Sigma^\omega$, if $prefixes(a) \subseteq (Prefixes(C))$ then $a \in C$.
  Example: Set of sequences in which every 1 is preceded by 0.
- Only Safety properties can be monitored using deterministic monitors.

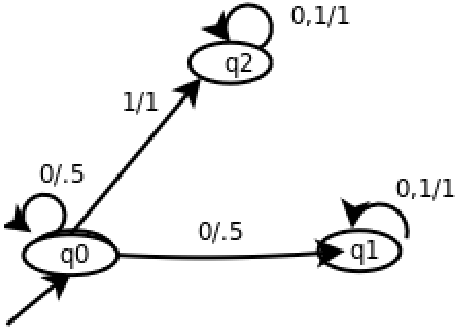
Properties of Infinite Executions

- Language $C \subseteq \Sigma^\omega$ is a safety property, if it is limit closed. That is, for any $a \in \Sigma^\omega$, if $\textit{prefixes}(a) \subseteq (\textit{Prefixes}(C))$ then $a \in C$.
  Example: Set of sequences in which every 1 is preceded by 0.
- Only Safety properties can be monitored using deterministic monitors.
- $C \subseteq \Sigma^\omega$ is an almost safety property if it is a countable union of safety properties.
  Example: Set of sequences in which 1 appears at least 3 times.

Monitoring Non-safety Properties

Monitoring Non-safety Properties

The following FPM monitors (i.e., accepts) the set of sequences in which 1 appears eventually. Not a safety property!



q1--- rejecting state

Classes of Monitorable Languages

Consider an alphabet Σ and a language $\mathcal{L} \subseteq \Sigma^\omega$.

- $\mathcal{L}$ is Monitorable with Strong Acceptance if there is a FPM $\mathcal{A}$ such that $\mathcal{L}$ is the set of strings rejected by $\mathcal{A}$ with probability 0. MSA is the class of all such languages.

Classes of Monitorable Languages

Consider an alphabet Σ and a language $\mathcal{L} \subseteq \Sigma^\omega$.

- $\mathcal{L}$ is Monitorable with Strong Acceptance if there is a FPM $\mathcal{A}$ such that $\mathcal{L}$ is the set of strings rejected by $\mathcal{A}$ with probability 0. MSA is the class of all such languages.
- $\mathcal{L}$ is Monitorable with Weak Acceptance if there is a FPM $\mathcal{A}$ such that $\mathcal{L}$ is the set of strings rejected by $\mathcal{A}$ with probability < 1 ,i.e., accepted with non-zero prob. MWA is the class of all such languages.

Classes of Monitorable Languages

Consider an alphabet Σ and a language $\mathcal{L} \subseteq \Sigma^\omega$.

- $\mathcal{L}$ is Monitorable with Strong Acceptance if there is a FPM $\mathcal{A}$ such that $\mathcal{L}$ is the set of strings rejected by $\mathcal{A}$ with probability 0. MSA is the class of all such languages.
- $\mathcal{L}$ is Monitorable with Weak Acceptance if there is a FPM $\mathcal{A}$ such that $\mathcal{L}$ is the set of strings rejected by $\mathcal{A}$ with probability < 1 ,i.e., accepted with non-zero prob. MWA is the class of all such languages.
- $\mathcal{L}$ is Monitorable with Strict Cut-off if there is a FPM $\mathcal{A}$ such that $\mathcal{L}$ is the set of strings rejected by $\mathcal{A}$ with probability $< \frac{1}{2}$. MSC is the class of all such languages.

Classes of Monitorable Languages

Consider an alphabet Σ and a language $\mathcal{L} \subseteq \Sigma^\omega$.

- $\mathcal{L}$ is Monitorable with Strong Acceptance if there is a FPM $\mathcal{A}$ such that $\mathcal{L}$ is the set of strings rejected by $\mathcal{A}$ with probability 0. MSA is the class of all such languages.
- $\mathcal{L}$ is Monitorable with Weak Acceptance if there is a FPM $\mathcal{A}$ such that $\mathcal{L}$ is the set of strings rejected by $\mathcal{A}$ with probability < 1 ,i.e., accepted with non-zero prob. MWA is the class of all such languages.
- $\mathcal{L}$ is Monitorable with Strict Cut-off if there is a FPM $\mathcal{A}$ such that $\mathcal{L}$ is the set of strings rejected by $\mathcal{A}$ with probability $< \frac{1}{2}$. MSC is the class of all such languages.
- $\mathcal{L}$ is Monitorable with Non-strict Cut-off if there is a FPM $\mathcal{A}$ such that $\mathcal{L}$ is the set of strings rejected by $\mathcal{A}$ with probability $\leq \frac{1}{2}$. MNC is the class of all such languages.

OutLine

Automata on Infinite Strings

Probabilistic Automata on Finite Strings

Probabilistic Buchi Automata

Finite State Probabilistic Monitors

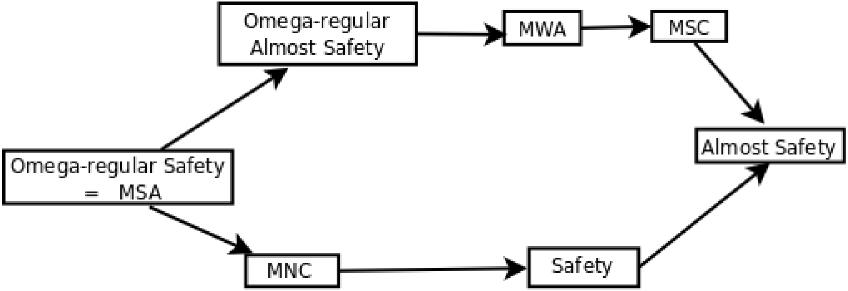
Expressiveness results for FPMS

Decidability and Complexity results for FPMs

Decidability and Expressiveness for PBAs

Hierarchical PBAs

Expressiveness results



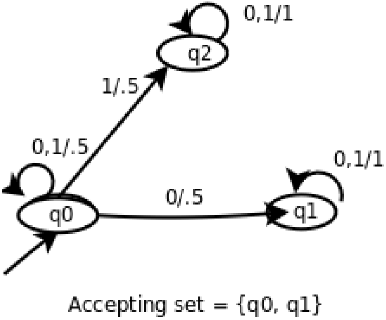
Containment Lattice among Language Classes

Monitoring Non-ω-regular Languages

- For any $\mathcal{A}, \mathcal{B}$, can construct $\mathcal{C}$ such that for any input *a*, $PrOfRej_{\mathcal{C}}(a) = PrOfRej_{\mathcal{A}}(a) \times PrOfRej_{\mathcal{B}}(a)$.

Monitoring Non-ω-regular Languages

- For any $\mathcal{A}, \mathcal{B}$, can construct $\mathcal{C}$ such that for any input a, $PrOfRej_{\mathcal{C}}(a) = PrOfRej_{\mathcal{A}}(a) \times PrOfRej_{\mathcal{B}}(a)$.

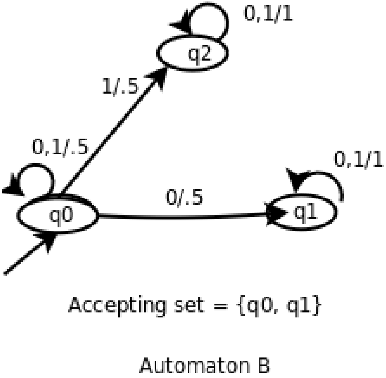


Automaton B

- For any $a \in \{0,1\}^\omega$, $PrOfRej_B(a) = val(a)$.

Monitoring Non-ω-regular Languages

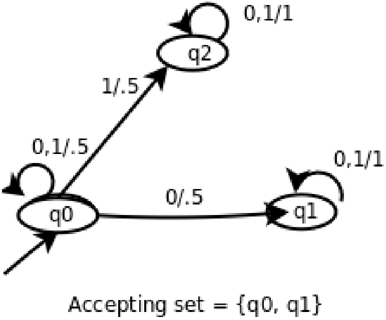
- For any $\mathcal{A}, \mathcal{B}$, can construct $\mathcal{C}$ such that for any input a, $PrOfRej_{\mathcal{C}}(a) = PrOfRej_{\mathcal{A}}(a) \times PrOfRej_{\mathcal{B}}(a)$.



- For any $a \in \{0,1\}^{\omega}$, $PrOfRej_B(a) = val(a)$.
- Construct FPM $\mathcal{C}$, $PrOfRej_{\mathcal{C}}(a) = (PrOfRej_B(a))^2$.

Monitoring Non-ω-regular Languages

- For any $\mathcal{A}, \mathcal{B}$, can construct $\mathcal{C}$ such that for any input a, $PrOfRej_{\mathcal{C}}(a) = PrOfRej_{\mathcal{A}}(a) \times PrOfRej_{\mathcal{B}}(a)$.

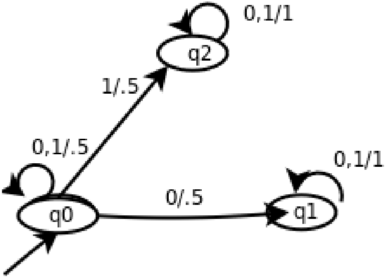


Automaton B

- For any $a \in \{0,1\}^\omega$, $PrOfRej_B(a) = val(a)$.
- Construct FPM $\mathcal{C}$, $PrOfRej_{\mathcal{C}}(a) = (PrOfRej_B(a))^2$.
- Let L be the set of inputs rejected by $\mathcal{C}$ with prob $\leq \frac{1}{2}$.

Monitoring Non-ω-regular Languages

- For any $\mathcal{A}, \mathcal{B}$, can construct $\mathcal{C}$ such that for any input a, $PrOfRej_{\mathcal{C}}(a) = PrOfRej_{\mathcal{A}}(a) \times PrOfRej_{\mathcal{B}}(a)$.



Accepting set = {q0, q1}

Automaton B

- For any $a \in \{0,1\}^\omega$, $PrOfRej_B(a) = val(a)$.
- Construct FPM $\mathcal{C}$, $PrOfRej_{\mathcal{C}}(a) = (PrOfRej_B(a))^2$.
- Let L be the set of inputs rejected by $\mathcal{C}$ with prob $\leq \frac{1}{2}$.
- $L = \{a : val(a) \leq \frac{1}{\sqrt{2}}\}$. $L \in MNC$ and not ω-regular.

OutLine

Decidability and Complexity results

- Given a FPM $\mathcal{A}$, determining if there is at least one input *a* such that $PrOfRej_{\mathcal{A}}(a) < 1$ is **PSPACE**-complete.

Decidability and Complexity results

- Given a FPM $\mathcal{A}$, determining if there is at least one input *a* such that $PrOfRej_{\mathcal{A}}(a) < 1$ is **PSPACE**-complete.
- **Upperbound proof:** Such an input exists, if there is a reachable state *q* from which the reject state is never reached on some input.

Decidability and Complexity results

- Given a FPM $\mathcal{A}$, determining if there is at least one input *a* such that $PrOfRej_{\mathcal{A}}(a) < 1$ is **PSPACE**-complete.
- **Upperbound proof:** Such an input exists, if there is a reachable state *q* from which the reject state is never reached on some input.
- Given a FPM $\mathcal{A}$, determining if there is at least one input *a* such that $PrOfRej_{\mathcal{A}}(a) < \frac{1}{2}$ is **R.E.**-complete.

Decidability and Complexity results

- Given a FPM $\mathcal{A}$, determining if there is at least one input *a* such that $PrOfRej_{\mathcal{A}}(a) < 1$ is **PSPACE**-complete.
- **Upperbound proof:** Such an input exists, if there is a reachable state *q* from which the reject state is never reached on some input.
- Given a FPM $\mathcal{A}$, determining if there is at least one input *a* such that $PrOfRej_{\mathcal{A}}(a) < \frac{1}{2}$ is **R.E.**-complete.
- **Upperbound proof:** Such an input exists, if there is an ultimately periodic input γ, i.e., of the form $\alpha(\beta)^{\omega}$, such that $PrOfRej_{\mathcal{A}}(\gamma) < \frac{1}{2}$.

Decidability and Complexity results

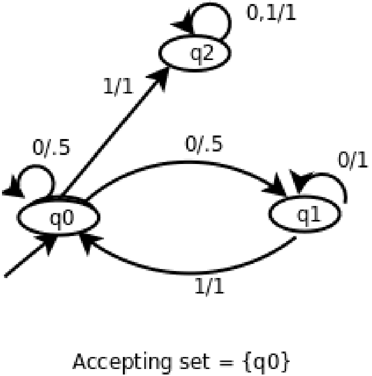
- Given a FPM $\mathcal{A}$, determining if there is at least one input *a* such that $PrOfRej_{\mathcal{A}}(a) < 1$ is **PSPACE**-complete.
- **Upperbound proof:** Such an input exists, if there is a reachable state *q* from which the reject state is never reached on some input.
- Given a FPM $\mathcal{A}$, determining if there is at least one input *a* such that $PrOfRej_{\mathcal{A}}(a) < \frac{1}{2}$ is **R.E.**-complete.
- **Upperbound proof:** Such an input exists, if there is an ultimately periodic input γ, i.e., of the form $\alpha(\beta)^{\omega}$, such that $PrOfRej_{\mathcal{A}}(\gamma) < \frac{1}{2}$.
- Given a FPM $\mathcal{A}$, determining if there is at least one input *a* such that $PrOfRej_{\mathcal{A}}(a) \leq \frac{1}{2}$ is **co-R.E.**-complete.

Summary of Complexity and Decidability results

	EMPTINESS	UNIVERSALITY
Msa	**PSPACE**-complete	**NL**-complete
Mwa	**PSPACE**-complete	**PSPACE**-complete
Msc	**co-R.E.**-complete	Π^1_1-complete
Mnc	**R.E.**-complete	**co-R.E.**-complete

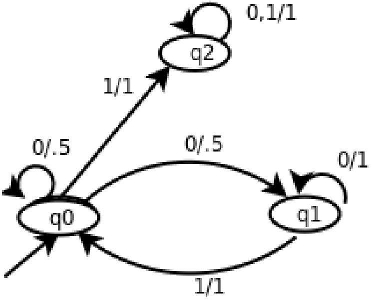
OutLine

Decidability and Expressiveness for PBAs



- What inputs are accepted with non-zero probability?

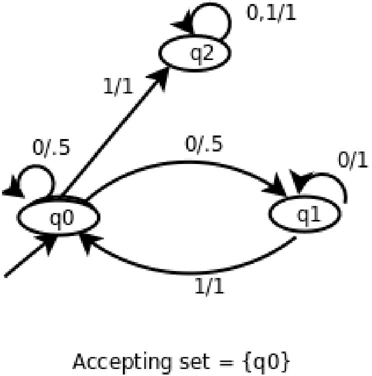
Decidability and Expressiveness for PBAs



Accepting set = {q0}

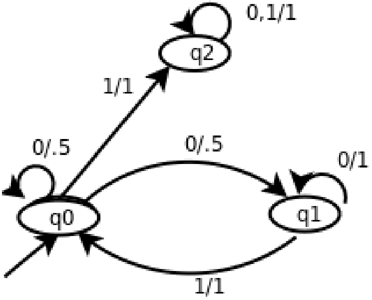
- What inputs are accepted with non-zero probability?
- Should have infinite number of 1s.

Decidability and Expressiveness for PBAs



- What inputs are accepted with non-zero probability?
- Should have infinite number of 1s.
- Consider 010101...10.... Accepted with Prob 0.
- Consider $010^2 10^3 1 \ldots 10^i \ldots$. Accepted with Prob > 0.

Decidability and Expressiveness for PBAs



Accepting set = {q0}

- What inputs are accepted with non-zero probability?
- Should have infinite number of 1s.
- Consider 010101...10.... Accepted with Prob 0.
- Consider $010^2 10^3 1 \ldots 10^i \ldots$. Accepted with Prob > 0.
- Acceptance probability $(1 - \frac{1}{2}) \times (1 - \frac{1}{2^2}) \times \ldots (1 - \frac{1}{2^i}) \times \ldots$

Decidability Results

Given a PBA $\mathcal{A}$:

- deciding if it accepts at least one input with non-zero probability is undecidable, is π_2^0-complete.

Decidability Results

Given a PBA $\mathcal{A}$:

- deciding if it accepts at least one input with non-zero probability is undecidable, is π_2^0-complete.
- deciding if it accepts at least one input with probability 1 is **PSPACE**-complete.

Decidability Results

Given a PBA $\mathcal{A}$:

- deciding if it accepts at least one input with non-zero probability is undecidable, is π_2^0-complete.
- deciding if it accepts at least one input with probability 1 is **PSPACE**-complete.
- deciding if it accepts all inputs with probability 1 is **PSPACE**-complete.

Decidability Results

Given a PBA $\mathcal{A}$:

- deciding if it accepts at least one input with non-zero probability is undecidable, is π_2^0-complete.
- deciding if it accepts at least one input with probability 1 is **PSPACE**-complete.
- deciding if it accepts all inputs with probability 1 is **PSPACE**-complete.
- Reduce to emptiness and universality problems for FPMs.

Expressiveness Results

- Probable Semantics: $\mathcal{L}_{>0}(\mathcal{A}) = \{a : \mathit{PrOfAcc}_{\mathcal{A}}(a) > 0\}$.
  Also $\mathbb{L}(\mathrm{PBA}^{>0}) = \{\mathcal{L}_{>0}(\mathcal{A}) : \mathit{PBA}\ \mathcal{A}\}$.

Expressiveness Results

- Probable Semantics: $\mathcal{L}_{>0}(\mathcal{A}) = \{a : \mathit{PrOfAcc}_{\mathcal{A}}(a) > 0\}$.
  Also $\mathbb{L}(\mathrm{PBA}^{>0}) = \{\mathcal{L}_{>0}(\mathcal{A}) : \mathit{PBA}\ \mathcal{A}\}$.
- $\mathbb{L}(\mathrm{PBA}^{>0})$ is closed under $\cup$ (union), $\cap$ and $\neg$ (Baier et al).
  Like ω-regular langs, i.e., langs accepted by non-det BA.

Expressiveness Results

- Probable Semantics: $\mathcal{L}_{>0}(\mathcal{A}) = \{a : PrOfAcc_{\mathcal{A}}(a) > 0\}$.
  Also $\mathbb{L}(\mathrm{PBA}^{>0}) = \{\mathcal{L}_{>0}(\mathcal{A}) : PBA\ \mathcal{A}\}$.
- $\mathbb{L}(\mathrm{PBA}^{>0})$ is closed under $\cup$ (union), $\cap$ and $\neg$ (Baier et al).
  Like ω-regular langs, i.e., langs accepted by non-det BA.
- Almsot Sure Semantics:
  $\mathcal{L}_{=1}(\mathcal{A}) = \{a : PrOfAcc_{\mathcal{A}}(a) = 1\}$.
  Also $\mathbb{L}(\mathrm{PBA}^{=1}) = \{\mathcal{L}_{=1}(\mathcal{A}) : PBA\ \mathcal{A}\}$.

Expressiveness Results

- Probable Semantics: $\mathcal{L}_{>0}(\mathcal{A}) = \{a : PrOfAcc_{\mathcal{A}}(a) > 0\}$.
  Also $\mathbb{L}(\mathrm{PBA}^{>0}) = \{\mathcal{L}_{>0}(\mathcal{A}) : PBA\ \mathcal{A}\}$.
- $\mathbb{L}(\mathrm{PBA}^{>0})$ is closed under $\cup$ (union), $\cap$ and $\neg$ (Baier et al). Like ω-regular langs, i.e., langs accepted by non-det BA.
- Almsot Sure Semantics:
  $\mathcal{L}_{=1}(\mathcal{A}) = \{a : PrOfAcc_{\mathcal{A}}(a) = 1\}$.
  Also $\mathbb{L}(\mathrm{PBA}^{=1}) = \{\mathcal{L}_{=1}(\mathcal{A}) : PBA\ \mathcal{A}\}$.
- $\mathbb{L}(\mathrm{PBA}^{=1})$ is closed under $\cup$ and $\cap$, but not under $\neg$. Like the Det-ω-Regular langs, i.e., languages accepted by det. BAs.

Expressiveness Results

- Probable Semantics: $\mathcal{L}_{>0}(\mathcal{A}) = \{a : \mathit{PrOfAcc}_{\mathcal{A}}(a) > 0\}$.
  Also $\mathbb{L}(\mathrm{PBA}^{>0}) = \{\mathcal{L}_{>0}(\mathcal{A}) : \mathit{PBA}\ \mathcal{A}\}$.
- $\mathbb{L}(\mathrm{PBA}^{>0})$ is closed under $\cup$ (union), $\cap$ and $\neg$ (Baier et al).
  Like ω-regular langs, i.e., langs accepted by non-det BA.
- Almsot Sure Semantics:
  $\mathcal{L}_{=1}(\mathcal{A}) = \{a : \mathit{PrOfAcc}_{\mathcal{A}}(a) = 1\}$.
  Also $\mathbb{L}(\mathrm{PBA}^{=1}) = \{\mathcal{L}_{=1}(\mathcal{A}) : \mathit{PBA}\ \mathcal{A}\}$.
- $\mathbb{L}(\mathrm{PBA}^{=1})$ is closed under $\cup$ and $\cap$, but not under $\neg$. Like the Det-ω-Regular langs, i.e., languages accepted by det. BAs.
- $\mathbb{L}(\mathrm{PBA}^{>0})$ is exactly the boolean closure of $\mathbb{L}(\mathrm{PBA}^{=1})$. Just like classes of languages accepted by non-det. BAs and det.BAs.

OutLine

Automata on Infinite Strings

Probabilistic Automata on Finite Strings

Probabilistic Buchi Automata

Finite State Probabilistic Monitors

Expressiveness results for FPMS

Decidability and Complexity results for FPMs

Decidability and Expressiveness for PBAs

Hierarchical PBAs

Hierarchical PBAs (HPBA)

PBA $\mathcal{A}$ is called HPBA, if

- Its states are divided into levels.

Hierarchical PBAs (HPBA)

PBA $\mathcal{A}$ is called HPBA, if

- Its states are divided into levels.
- On an input symbol, at most one next state is at the same level, all others are at higher levels.

Hierarchical PBAs (HPBA)

PBA $\mathcal{A}$ is called HPBA, if

- Its states are divided into levels.
- On an input symbol, at most one next state is at the same level, all others are at higher levels.

$$\mathbb{L}(\mathrm{HPBA}^{>0}) = \{\mathcal{L}_{>0}(\mathcal{A}) \,:\, \textit{HPBA}\ \mathcal{A}\}.$$

Hierarchical PBAs (HPBA)

PBA $\mathcal{A}$ is called HPBA, if

- Its states are divided into levels.
- On an input symbol, at most one next state is at the same level, all others are at higher levels.

$$\mathbb{L}(\mathrm{HPBA}^{>0}) = \{\mathcal{L}_{>0}(\mathcal{A}) : \textit{HPBA } \mathcal{A}\}.$$

$$\mathbb{L}(\mathrm{HPBA}^{=1}) = \{\mathcal{L}_{=1}(\mathcal{A}) : \textit{HPBA } \mathcal{A}\}.$$

Hierarchical PBAs (HPBA)

PBA $\mathcal{A}$ is called HPBA, if

- Its states are divided into levels.
- On an input symbol, at most one next state is at the same level, all others are at higher levels.

$$\mathbb{L}(\mathrm{HPBA}^{>0}) = \{\mathcal{L}_{>0}(\mathcal{A}) : \textit{HPBA } \mathcal{A}\}.$$
$$\mathbb{L}(\mathrm{HPBA}^{=1}) = \{\mathcal{L}_{=1}(\mathcal{A}) : \textit{HPBA } \mathcal{A}\}.$$

- $\mathbb{L}(\mathrm{HPBA}^{>0}) = \omega$-Regular.

Hierarchical PBAs (HPBA)

PBA $\mathcal{A}$ is called HPBA, if

- Its states are divided into levels.
- On an input symbol, at most one next state is at the same level, all others are at higher levels.

$$\mathbb{L}(\mathrm{HPBA}^{>0}) = \{\mathcal{L}_{>0}(\mathcal{A}) : \textit{HPBA } \mathcal{A}\}.$$
$$\mathbb{L}(\mathrm{HPBA}^{=1}) = \{\mathcal{L}_{=1}(\mathcal{A}) : \textit{HPBA } \mathcal{A}\}.$$

- $\mathbb{L}(\mathrm{HPBA}^{>0}) = \omega$-Regular.
- $\mathbb{L}(\mathrm{HPBA}^{=1})$ is exactly the class of langs accepted by det. BAs.

Hierarchical PBAs (HPBA)

PBA $\mathcal{A}$ is called HPBA, if

- Its states are divided into levels.
- On an input symbol, at most one next state is at the same level, all others are at higher levels.

$$\mathbb{L}(\mathrm{HPBA}^{>0}) = \{\mathcal{L}_{>0}(\mathcal{A}) : \textit{HPBA } \mathcal{A}\}.$$
$$\mathbb{L}(\mathrm{HPBA}^{=1}) = \{\mathcal{L}_{=1}(\mathcal{A}) : \textit{HPBA } \mathcal{A}\}.$$

- $\mathbb{L}(\mathrm{HPBA}^{>0}) = \omega$-Regular.
- $\mathbb{L}(\mathrm{HPBA}^{=1})$ is exactly the class of langs accepted by det. BAs.
- Determining emptiness and universality of $\mathcal{L}_{>0}(\mathcal{A})$ is in **NL**and **PSPACE**-complete, respectively.

Hierarchical PBAs (HPBA)

PBA $\mathcal{A}$ is called HPBA, if

- Its states are divided into levels.
- On an input symbol, at most one next state is at the same level, all others are at higher levels.

$$\mathbb{L}(\mathrm{HPBA}^{>0}) = \{\mathcal{L}_{>0}(\mathcal{A}) : \textit{HPBA } \mathcal{A}\}.$$

$$\mathbb{L}(\mathrm{HPBA}^{=1}) = \{\mathcal{L}_{=1}(\mathcal{A}) : \textit{HPBA } \mathcal{A}\}.$$

- $\mathbb{L}(\mathrm{HPBA}^{>0}) = \omega$-Regular.
- $\mathbb{L}(\mathrm{HPBA}^{=1})$ is exactly the class of langs accepted by det. BAs.
- Determining emptiness and universality of $\mathcal{L}_{>0}(\mathcal{A})$ is in **NL**and **PSPACE**-complete, respectively.
- Determining emptiness and universality of $\mathcal{L}_{=1}(\mathcal{A})$ is **PSPACE**-complete and in **NL**, respectively.

Conclusions and Future Work

- Applications to non-deterministic probabilistic programs.

Conclusions and Future Work

- Applications to non-deterministic probabilistic programs.
- Explore relationships to Partially Observable Markov Decision Processes. (POMDPs)

Conclusions and Future Work

- Applications to non-deterministic probabilistic programs.
- Explore relationships to Partially Observable Markov Decision Processes. (POMDPs)
- Explore power of randomization in other computation models on infinite inputs.