Lecture 17
ω-regular properties

Dr. Dave Parker

Department of Computer Science
University of Oxford

Long-run properties

- Last lecture: regular safety properties
  - e.g. “a message failure never occurs”
  - e.g. “an alarm is only ever triggered by an error”
  - bad prefixes represented by a regular language
  - property always refuted by a finite trace/path
- Liveness properties
  - e.g. "for every request, an acknowledge eventually follows”
  - no finite prefix refutes the property
  - any finite prefix can be extended to a satisfying trace
- Fairness assumptions
  - e.g. “every process that is enabled infinitely often is scheduled infinitely often”
- Need properties of infinite paths

Overview

- ω-regular expressions and ω-regular languages
- Nondeterministic Büchi automata (NBA)
- Deterministic Büchi automata (DBA)
- Deterministic Rabin automata (DRA)
- Deterministic ω-automata and DTMCs

ω-regular expressions

- Regular expressions E over alphabet Σ are given by:
  - $E ::= \emptyset \mid \varepsilon \mid \alpha \mid E + E \mid E.E \mid E^*$ (where $\alpha \in \Sigma$)
- An ω-regular expression takes the form:
  - $G = E_1.(F_1)^\omega + E_2.(F_2)^\omega + \ldots + E_n.(F_n)^\omega$
  - where E_i and F_i are regular expressions with $\varepsilon \notin L(F_i)$
- The language $L(G) \subseteq \Sigma^\omega$ of an ω-regular expression G
  - is $L(E_1).L(F_1)^\omega \cup L(E_2).L(F_2)^\omega + \ldots + L(E_n).L(F_n)^\omega$
  - where $L(E)$ is the language of regular expression E
  - and $L(E)^\omega = \{ w^\omega \mid w \in L(E) \}$
- Example: $(\alpha+\beta+\gamma)^*(\beta+\gamma)^\omega$ for $\Sigma = \{ \alpha, \beta, \gamma \}$

ω-regular languages/properties

- A language $L \subseteq \Sigma^\omega$ over alphabet Σ is an ω-regular language if and only if:
  - $L = L(G)$ for some ω-regular expression G

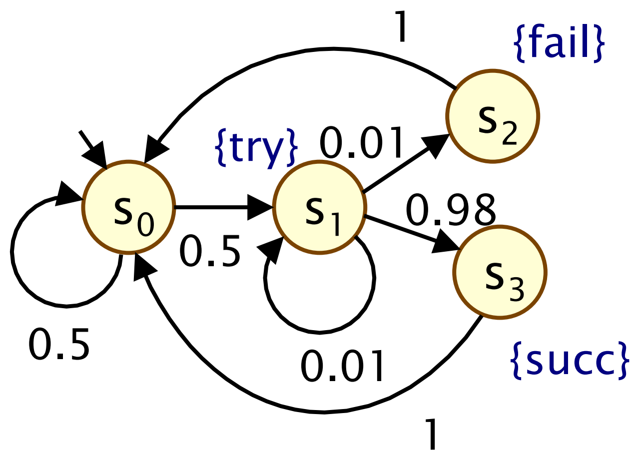
- ω-regular languages are:
  - closed under intersection
  - closed under complementation

- $P \subseteq (2^{AP})^\omega$ is an ω-regular property
  - if P is an ω-regular language over 2^{AP}
  - (where AP is the set of atomic propositions for some model)
  - path ω satisfies P if $\text{trace}(\omega) \in P$
  - NB: any regular safety property is an ω-regular property

Examples

- A message is sent successfully infinitely often
  - $((\neg succ)^*.succ)^\omega$

- Every time the process tries to send a message, it eventually succeeds in sending it
  - $((\neg try)^* + try.(\neg succ)^*.succ)^\omega$



Büchi automata

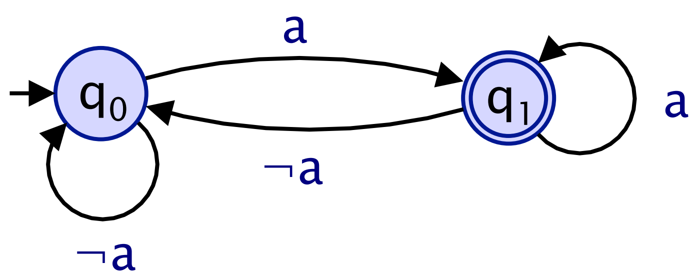
- A nondeterministic Büchi automaton (NBA) is…
  - a tuple $A = (Q, \Sigma, \delta, Q_0, F)$ where:
  - Q is a finite set of states
  - Σ is an alphabet
  - $\delta : Q \times \Sigma \rightarrow 2^Q$ is a transition function
  - $Q_0 \subseteq Q$ is a set of initial states
  - $F \subseteq Q$ is a set of “accept” states
  - i.e. just like a nondeterministic finite automaton (NFA)
- The difference is the accepting condition…

Language of an NBA

- Consider a Büchi automaton $A = (Q, \Sigma, \delta, Q_0, F)$
- A run of A on an infinite word $\alpha_1\alpha_2\ldots$ is:
  - an infinite sequence of automata states $q_0q_1\ldots$ such that:
  - $q_0 \in Q_0$ and $q_{i+1} \in \delta(q_i, \alpha_{i+1})$ for all $i \geq 0$
- An accepting run is a run with $q_i \in F$ for infinitely many i
- The language L(A) of A is the set of all infinite words on which there exists an accepting run of A

Example

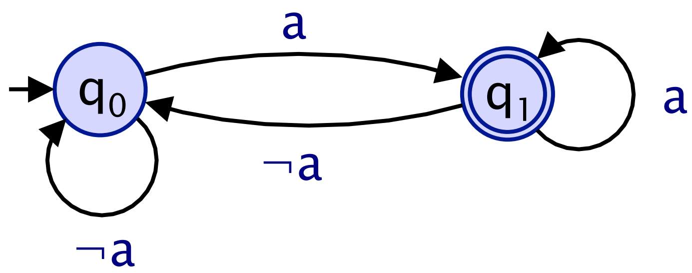
- Infinitely often a



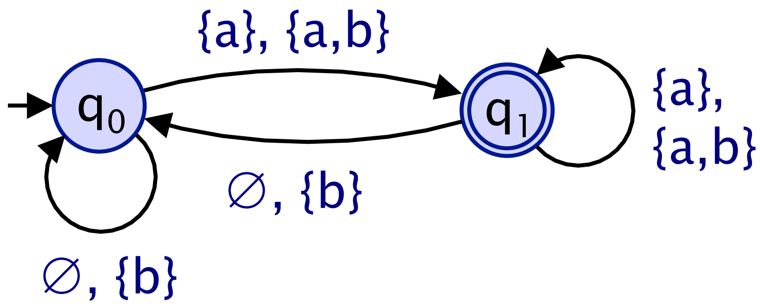
Example…

- As in the last lecture, we use automata to represent languages of the form $L \subseteq (2^{AP})^\omega$

- So, if AP = {a,b}, then:



- …is actually:



Properties of Büchi automata

- ω-regular languages
  - L(A) is an ω-regular language for any NBA A
  - any ω-regular language can be represented by an NBA

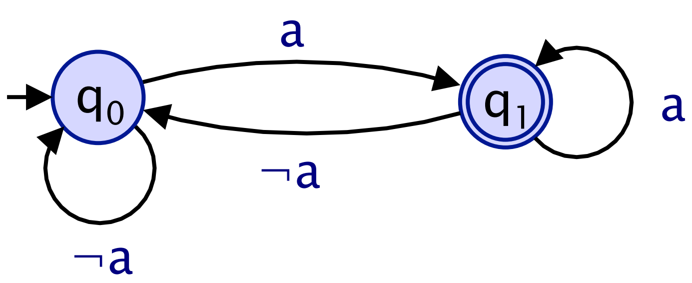
- ω-regular expressions
  - like for finite automata, can construct an NBA from an arbitrary ω-regular expression $E_1.(F_1)^\omega + \ldots + E_n.(F_n)^\omega$
  - i.e. there are operations on NBAs to:
    - construct NBA accepting L^ω for regular language L
    - construct NBA from NFA for (regular) E and NBA for (ω-regular) F
    - construct NBA accepting union $L(A_1) \cup L(A_2)$ for NBA A_1 and A_2

Büchi automata and LTL

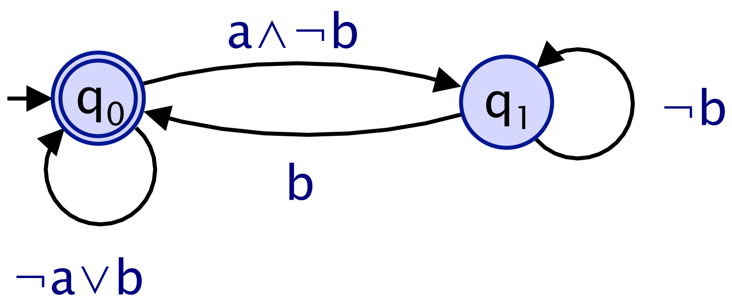
- LTL formulae
  - $\psi ::= \text{true} \mid a \mid \psi \wedge \psi \mid \neg\psi \mid X\,\psi \mid \psi\, U\, \psi$
  - where $a \in AP$ is an atomic proposition
- Can convert any LTL formula ψ into an NBA A over 2^{AP}
  - i.e. $\omega \vDash \psi \Leftrightarrow trace(\omega) \in L(A)$ for any path ω
- LTL-to-NBA translation (see e.g. [VW94], [DGV99])
  - construct a generalized NBA (multiple sets of accept states)
  - based on decomposition of LTL formula into subformulae
  - can convert GNBA into an equivalent NBA
  - various optimisations to the basic techniques developed
  - not covered here; see e.g. section 5.2 of [BK08]

Büchi automata and LTL

- GF a   (“infinitely often a”)



- G(a → F b)   (“b always eventually follows a”)



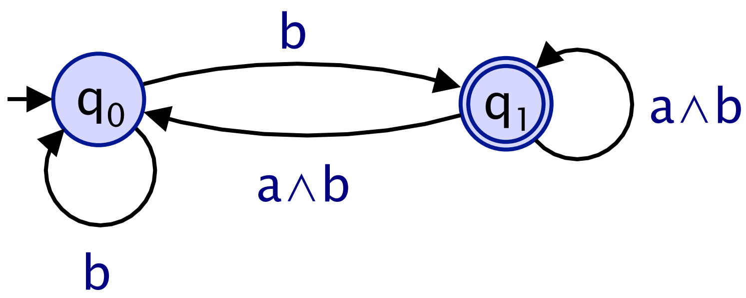
Deterministic Büchi automata

- Like for finite automata…
- A NBA is deterministic if:
  - $|Q_0|=1$
  - $|\delta(q, \alpha)| \leq 1$ for all $q \in Q$ and $\alpha \in \Sigma$
  - i.e. one initial state and no nondeterministic successors
- A deterministic Büchi automaton (DBA) is total if:
  - $|\delta(q, \alpha)| = 1$ for all $q \in Q$ and $\alpha \in \Sigma$
  - i.e. unique successor states
- But, NBA can not always be determinised…
  - i.e. NBA are strictly more expressive than DBA

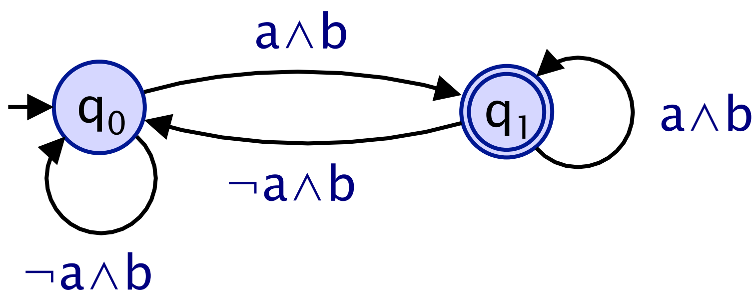
NBA and DBA

- NBA and DBA for the LTL formula G b ∧ GF a

NBA:



DBA:

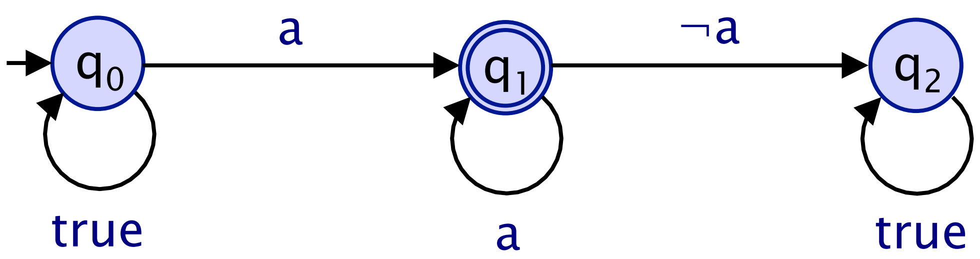


No DBA possible

- Consider the ω-regular expression $(\alpha+\beta)^*\alpha^\omega$ over $\Sigma=\{\alpha,\beta\}$
  - i.e. words containing only finitely many instances of β
  - there is no deterministic Büchi automata accepting this

- In particular, take $\alpha = \{a\}$ and $\beta = \varnothing$, i.e. $\Sigma=2^{AP}$, $AP=\{a\}$
  - $(\alpha+\beta)^*\alpha^\omega$ represents the LTL formula FG a

- FG a is represented by the following NBA:



- But there is no DBA for FG a

Deterministic Rabin automata

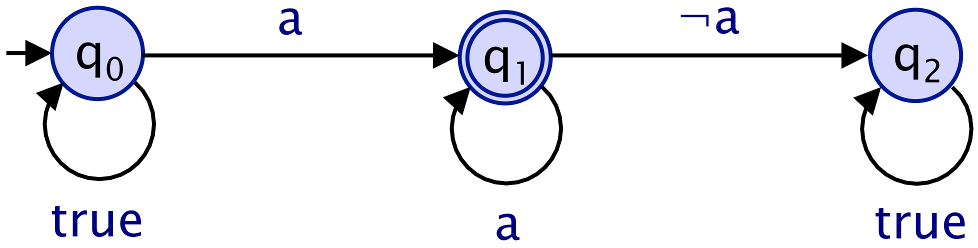
- A deterministic Rabin automaton (DRA) is…
  - a tuple $A = (Q, \Sigma, \delta, q_0, Acc)$ where:
  - Q is a finite set of states
  - Σ is an alphabet
  - $\delta : Q \times \Sigma \rightarrow Q$ is a transition function
  - $q_0 \in Q$ is an initial state
  - $Acc \subseteq 2^Q \times 2^Q$ is an acceptance condition
- The acceptance condition is a set of pairs of state sets
  - $Acc = \{ (L_i, K_i) \mid 1 \leq i \leq k \}$

Deterministic Rabin automata

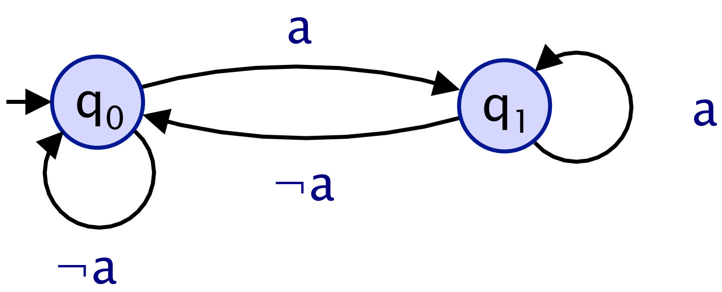
- A run of a word on a DRA is accepting iff:
  - for some pair (L_i, K_i), the states in L_i are visited finitely often and (some of) the states in K_i are visited infinitely often
  - or in LTL: $\bigvee_{1 \leq i \leq k} (FG \neg L_i \wedge GF K_i)$
- Hence:
  - a deterministic Büchi automaton is a special case of a deterministic Rabin automaton where $Acc = \{ (\varnothing, \{F\}) \}$

FG a

- NBA for FG a (no DBA exists)



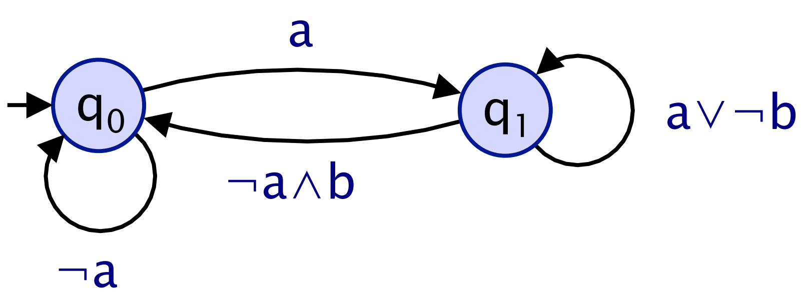
- DRA for FG a



  - where acceptance condition is $Acc = \{ (\{q_0\},\{q_1\}) \}$

Example – DRA

- Another example of a DRA (over alphabet $2^{\{a,b\}}$)



  - where acceptance condition is Acc = { $(\{q_1\},\{q_0\})$ }

- In LTL: $G(a \rightarrow F(\neg a \wedge b)) \wedge FG\ \neg a$

Properties of DRA

- Any ω-regular language can represented by a DRA
  - (and L(A) is an ω-regular language for any DRA A)
- i.e. DRA and NBA are equally expressive
  - (but NBA may be more compact)
  - and DRA are strictly more expressive than DBA
- Any NBA can be converted to an equivalent DRA [Saf88]
  - size of the resulting DRA is $2^{O(n\log n)}$

Deterministic ω-automata and DTMCs

- Let A be a DBA or DRA over the alphabet 2^{AP}
  - i.e. $L(A) \subseteq (2^{AP})^\omega$ identifies a set of paths in a DTMC

- Let $Prob^D(s, A)$ denote the corresponding probability
  - from state s in a discrete-time Markov chain D
  - i.e. $Prob^D(s, A) = Pr^D_s\{\ \omega \in Path(s) \mid trace(\omega) \in L(A)\ \}$

- Like for finite automata (i.e. DFA), we can evaluate $Prob^D(s, A)$ by constructing a product of D and A
  - which records the state of both the DTMC and the automaton

Product DTMC for a DBA

- For a DTMC $D = (S, s_{init}, \mathbf{P}, L)$
- and a (total) DBA $A = (Q, \Sigma, \delta, q_0, F)$

- The product DTMC $D \otimes A$ is:
  - the DTMC $(S \times Q, (s_{init}, q_{init}), \mathbf{P}', L')$ where:

$$q_{init} = \delta(q_0, L(s_{init}))$$

$$\mathbf{P}'((s_1,q_1),(s_2,q_2)) = \begin{cases} \mathbf{P}(s_1,s_2) & \text{if } q_2 = \delta(q_1, L(s_2)) \\ 0 & \text{otherwise} \end{cases}$$

$$L'(s,q) = \{ \text{ accept } \} \text{ if } q \in F \text{ and } L'(s,q) = \emptyset \text{ otherwise}$$

- Since A is deterministic
  - unique mappings between paths of D, A and $D \otimes A$
  - probabilities of paths are preserved

Product DTMC for a DBA

- For DTMC D and DBA A

$$\mathrm{Prob}^{D}(s, A) = \mathrm{Prob}^{D\otimes A}((s,q_s), \text{GF accept})$$

  - where $q_s = \delta(q_0, L(s))$

- Hence:

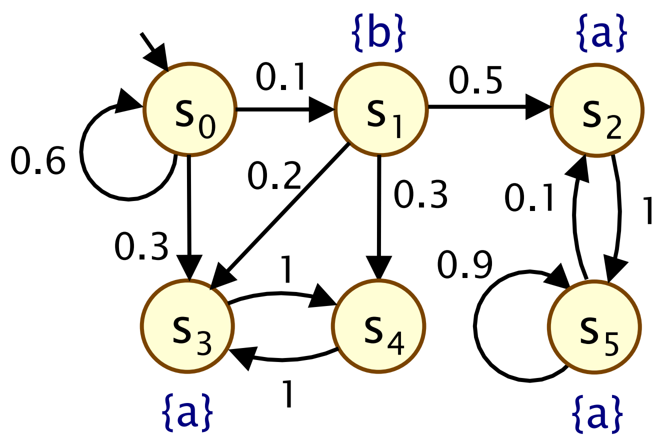
$$\mathrm{Prob}^{D}(s, A) = \mathrm{Prob}^{D\otimes A}((s,q_s), \text{F } T_{GFaccept})$$

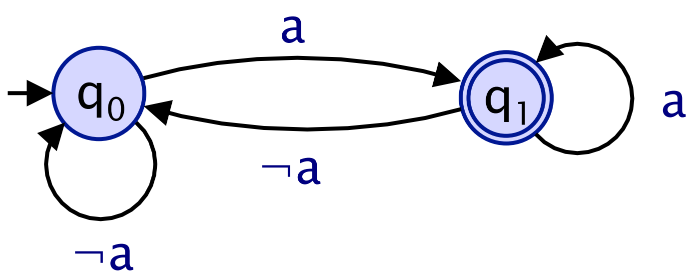
  - where $T_{GFaccept}$ = union of $D\otimes A$ BSCCs T with $T \cap Sat(accept) \neq \emptyset$

- Reduces to computing BSCCs and reachability probabilities

Example

- Compute $Prob(s_0, GF\ a)$
  - property can be represented as a DBA

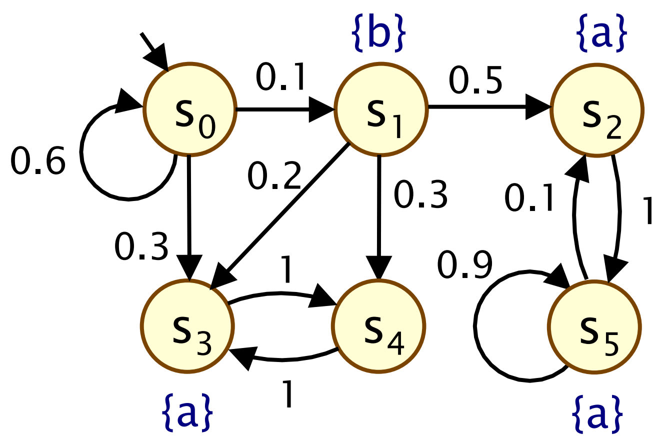


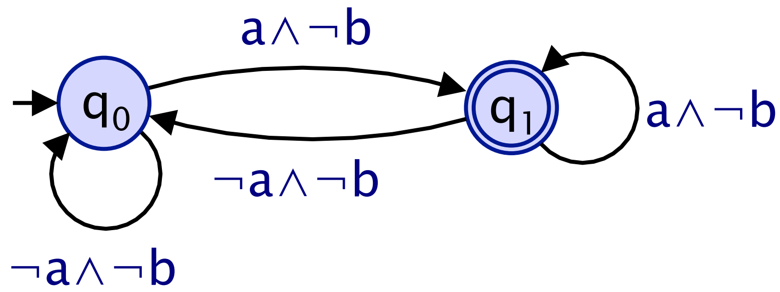


- Result: 1

Example 2

- Compute Prob(s_0, G ¬b ∧ GF a)
  - property can be represented as a DBA





- Result: 0.75

Product DTMC for a DRA

- For a DTMC $D = (S, s_{init}, \mathbf{P}, L)$
- and a (total) DRA $A = (Q, \Sigma, \delta, q_0, Acc)$
  - where $Acc = \{ (L_i, K_i) \mid 1 \leq i \leq k \}$

- The product DTMC $D \otimes A$ is:
  - the DTMC $(S \times Q, (s_{init}, q_{init}), \mathbf{P}', L')$ where:

    $q_{init} = \delta(q_0, L(s_{init}))$

    $$\mathbf{P}'((s_1,q_1),(s_2,q_2)) = \begin{cases} \mathbf{P}(s_1,s_2) & \text{if } q_2 = \delta(q_1, L(s_2)) \\ 0 & \text{otherwise} \end{cases}$$

    $l_i \in L'(s,q)$ if $q \in L_i$ and $k_i \in L'(s,q)$ if $q \in K_i$

    (i.e. state sets of acceptance condition used as labels)

- (same product as for DBA, except for state labelling)

Product DTMC for a DRA

- For DTMC D and DRA A

$$\text{Prob}^{D}(s, A) = \text{Prob}^{D\otimes A}((s,q_s), \vee_{1\leq i\leq k} (FG\ \neg l_i \wedge GF\ k_i))$$

  - where $q_s = \delta(q_0,L(s))$
- Hence:

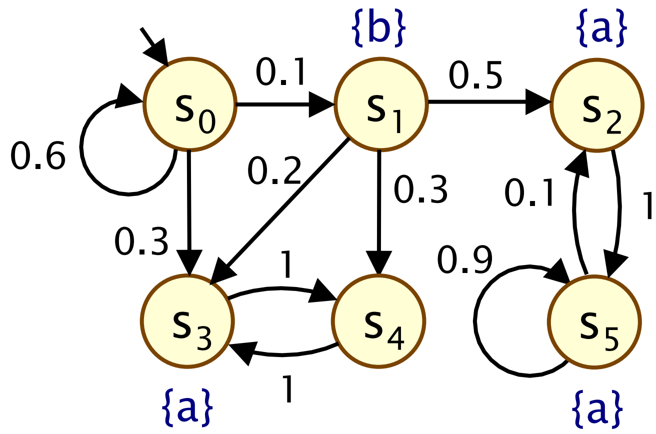
$$\text{Prob}^{D}(s, A) = \text{Prob}^{D\otimes A}((s,q_s), F\ T_{Acc})$$

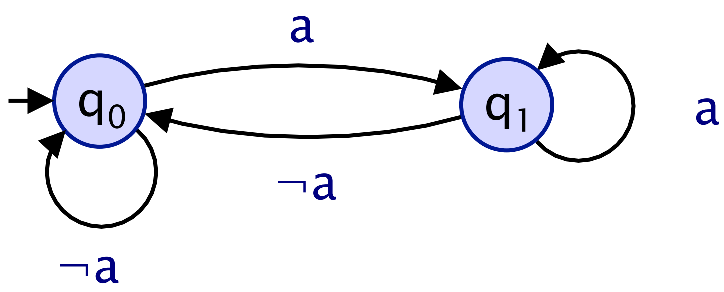
  - where T_{Acc} is the union of all accepting BSCCs in $D\otimes A$
  - an accepting BSCC T of $D\otimes A$ is such that, for some $1\leq i\leq k$:
    - $q \vDash \neg l_i$ for all $(s,q) \in T$ and $q \vDash k_i$ for some $(s,q) \in T$
    - i.e. $T \cap (S \times L_i) = \varnothing$ and $T \cap (S \times K_i) \neq \varnothing$

- Reduces to computing BSCCs and reachability probabilities

Example 3

- Compute Prob(s_0, FG a)
  - property can be represented as a DRA



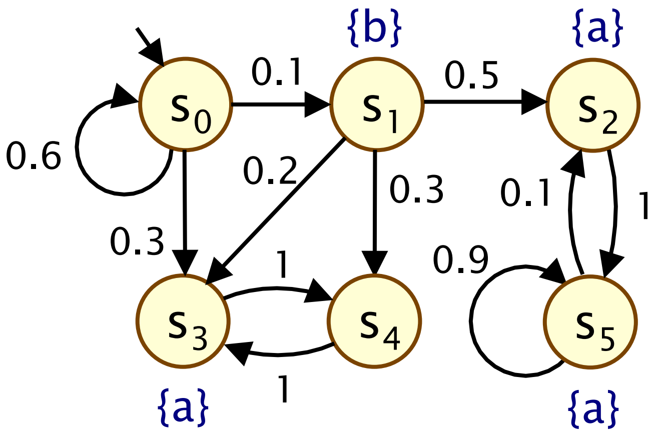


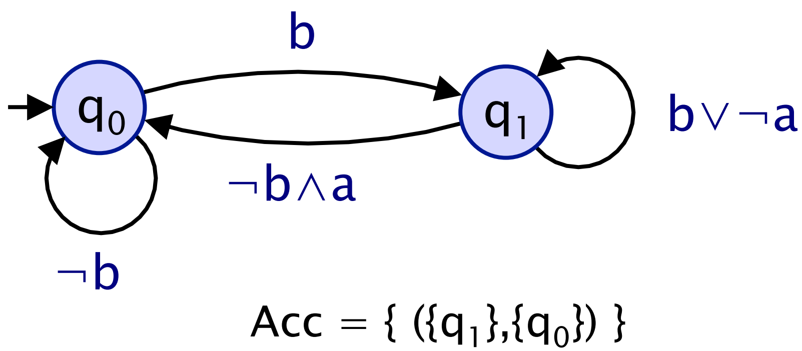
Acc = { ({q_0},{q_1}) }

- Result: 0.125

Example 4

- Compute Prob(s_0, G(b → F(¬b∧a)) ∧ FG ¬b)
  - property can be represented as a DRA





- Result: 1

Summing up…

- ω-regular expressions and ω-regular languages
  - languages of infinite words: $E_1.(F_1)^\omega + E_2.(F_2)^\omega + \ldots + E_n.(F_n)^\omega$
- Nondeterministic Büchi automata (NBA)
  - accepting runs visit a state in F infinitely often
  - can represent any ω-regular language by an NBA
  - can translate any LTL formula into equivalent NBA
- Deterministic Büchi automata (DBA)
  - strictly less expressive than NBA (e.g. no NBA for FG a)
- Deterministic Rabin automata (DRA)
  - generalised acceptance condition: $\{ (L_i, K_i) \mid 1 \leq i \leq k \}$
  - as expressive as NBA; can convert any NBA to a DRA
- Deterministic ω-automata and DTMCs
  - product DTMC + BSCC computation + reachability