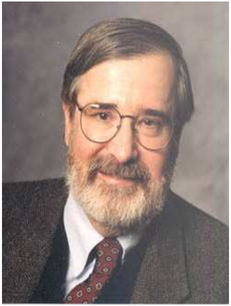
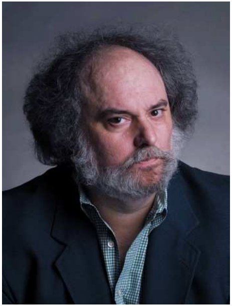
CS 228 : Logic in Computer Science

Krishna. S

Linear Temporal Logic

Model Checking

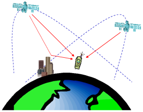
  

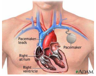
- Year 2007 : ACM confers the Turing Award to the pioneers of Model Checking: Ed Clarke, Allen Emerson, and Joseph Sifakis
- `https://amturing.acm.org/award_winners/clarke_1167964`

Model checking

- Model checking has evolved in last 25 years into a widely used verification and debugging technique for software and hardware.
- Model checking used (and further developed) by companies/institutes such as IBM, Intel, NASA, Cadence, Microsoft, and Siemens, and has culminated in many freely downloadable software tools that allow automated verification.

What is Model Checking?



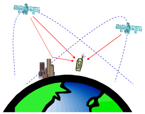


System → specification

satisfy?

good/bad properties

What is Model Checking?



System

satisfy?

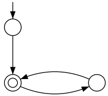
specification

System Model

model-checking

$\models ?$

Spec Model



ϕ

Model Checker as a Black Box

- Inputs to Model checker : A finite state system M, and a property P to be checked.
- Question : Does M satisfy P?
- Possible Outputs
  - Yes, M satisfies P
  - No, here is a counter example!.

What are Models?

Transition Systems

- States labeled with propositions
- Transition relation between states
- Action-labeled transitions to facilitate composition

What are Properties?

Example properties

- Can the system reach a deadlock?
- Can two processes ever be together in a critical section?
- On termination, does a program provide correct output?

Notations for Infinite Words

- Σ is a finite alphabet
- Σ^* set of finite words over Σ
- An infinite word is written as $\alpha = \alpha(0)\alpha(1)\alpha(2)\ldots$, where $\alpha(i) \in \Sigma$
- Such words are called ω-words
- a^ω, $a^7.b^\omega$

Transition Systems

A Transition System is a tuple $(S, Act, \rightarrow, I, AP, L)$ where

- S is a set of states
- Act is a set of actions
- $s \xrightarrow{\alpha} s'$ in $S \times Act \times S$ is the transition relation
- $I \subseteq S$ is the set of initial states
- AP is the set of atomic propositions
- $L : S \rightarrow 2^{AP}$ is the labeling function

Traces of Transition Systems

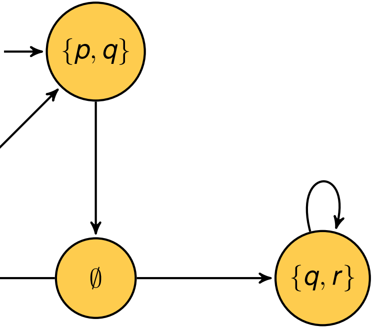
- Labels of the locations represent values of all observable propositions $\in AP$
- Captures system state
- Focus on sequences $L(s_0)L(s_1)\ldots$ of labels of locations
- Such sequences are called traces
- Assuming transition systems have no terminal states,
  - Traces are infinite words over 2^{AP}
  - Traces $\in (2^{AP})^\omega$
  - Go to the example slide and define traces

Traces of Transition Systems

Given a transition system $TS = (S, Act, \rightarrow, I, AP, L)$ without terminal states,

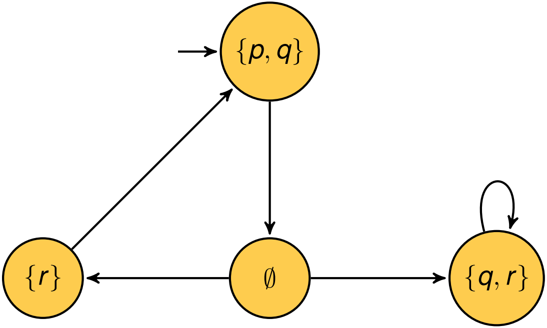
- All maximal executions/paths are infinite
- Path $\pi = s_0 s_1 s_2 \ldots$, $trace(\pi) = L(s_0)L(s_1)\ldots$
- For a set Π of paths, $Trace(\Pi) = \{trace(\pi) \mid \pi \in \Pi\}$
- For a location s, $Traces(s) = Trace(Paths(s))$
- $Traces(TS) = \bigcup_{s \in I} Traces(s)$

Example Traces



$AP = \{p, q, r, t\}$

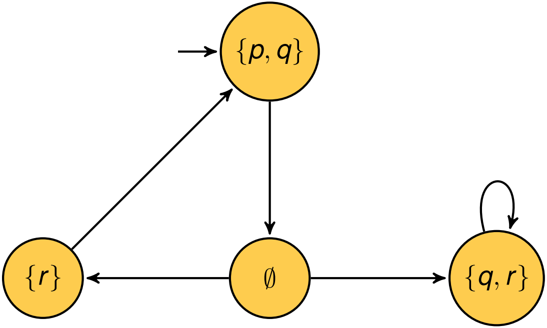
Example Traces



$AP = \{p, q, r, t\}$

- $\{p, q\}\emptyset\{q, r\}^\omega$

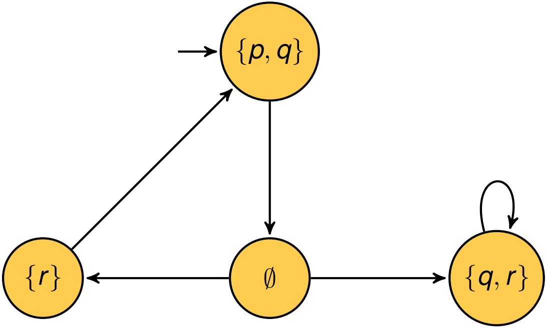
Example Traces



$AP = \{p, q, r, t\}$

- $\{p, q\}\emptyset\,\{q, r\}^\omega$
- $(\{p, q\}\emptyset\{r\})^\omega$

Example Traces



$AP = \{p, q, r, t\}$

- $\{p, q\}\emptyset\ \{q, r\}^\omega$
- $(\{p, q\}\emptyset\{r\})^\omega$
- $(\{p, q\}\emptyset\{r\})^*\ \{p, q\}\emptyset\ \{q, r\}^\omega$

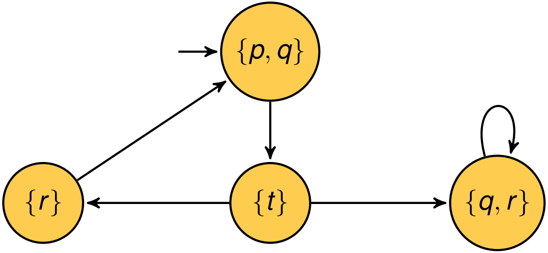
Linear Time Properties

- Linear-time properties specify traces that a TS must have
- A LT property P over AP is a subset of $(2^{AP})^\omega$
- TS over AP satisfies a LT property P over AP

$$TS \models P \text{ iff } Traces(TS) \subseteq P$$

- $s \in S$ satisfies LT property P (denoted $s \models P$) iff $Traces(s) \subseteq P$

Specifying Traces



- Whenever p is true, r will eventually become true
  - $\{A_0 A_1 A_2 \cdots \mid \forall i \geqslant 0, p \in A_i \to \exists j \geqslant i, r \in A_j\}$
- q is true infinitely often
  - $\{A_0 A_1 A_2 \cdots \mid \forall i \geqslant 0, \exists j \geqslant i, q \in A_j\}$
- Whenever r is true, so is q
  - $\{A_0 A_1 \cdots \mid \forall i \geqslant 0, r \in A_i \to q \in A_i\}$

Syntax of Linear Temporal Logic

Given *AP*, a set of propositions,

Syntax of Linear Temporal Logic

Given *AP*, a set of propositions,

- Propositional logic formulae over *AP*
  - $a \in AP$ (atomic propositions)
  - $\neg\varphi$, $\varphi \wedge \psi$, $\varphi \vee \psi$

Syntax of Linear Temporal Logic

Given *AP*, a set of propositions,

- Propositional logic formulae over *AP*
  - $a \in AP$ (atomic propositions)
  - $\neg\varphi$, $\varphi \wedge \psi$, $\varphi \vee \psi$
- Temporal Operators
  - $\bigcirc\varphi$ (Next φ)
  - $\varphi \, \mathsf{U} \psi$ (φ holds until a ψ-state is reached)
- LTL : Logic for describing LT properties

Semantics (On the board)

LTL formulae φ over AP interpreted over words $w \in \Sigma^\omega$, $\Sigma = 2^{AP}$,
$w \models \varphi$

CS 228 : Logic in Computer Science

Krishna. S

Derived Operators

- *true* $= \varphi \vee \neg\varphi$
- *false* $= \neg$*true*
- $\Diamond\varphi =$ *true* $\mathsf{U}\varphi$ (Eventually φ)
- $\Box\varphi = \neg\Diamond\neg\varphi$ (Forever φ)

Precedence

- Unary Operators bind stronger than Binary
- $\bigcirc$ and $\neg$ equally strong
- U takes precedence over $\wedge, \vee, \rightarrow$
  - $a \vee b\,\mathsf{U}c \equiv a \vee (b\,\mathsf{U}c)$
  - $\bigcirc a\,\mathsf{U}\neg b \equiv (\bigcirc a)\,\mathsf{U}(\neg b)$

Examples

- Whenever the traffic light is red, it cannot become green immediately:

Examples

- Whenever the traffic light is red, it cannot become green immediately:
  $\Box(\mathit{red} \rightarrow \neg \bigcirc \mathit{green})$

Examples

- Whenever the traffic light is red, it cannot become green immediately:
  $\Box(\mathit{red} \rightarrow \neg \bigcirc \mathit{green})$
- Eventually the traffic light will become yellow

Examples

- Whenever the traffic light is red, it cannot become green immediately:
  $\Box(\mathit{red} \rightarrow \neg \bigcirc \mathit{green})$
- Eventually the traffic light will become yellow
  $\Diamond \mathit{yellow}$

Examples

- Whenever the traffic light is red, it cannot become green immediately:
  $\Box(red \rightarrow \neg \bigcirc green)$
- Eventually the traffic light will become yellow
  $\Diamond yellow$
- Once the traffic light becomes yellow, it will eventually become green

Examples

- Whenever the traffic light is red, it cannot become green immediately:
  $\Box(\mathit{red} \rightarrow \neg \bigcirc \mathit{green})$
- Eventually the traffic light will become yellow
  $\Diamond \mathit{yellow}$
- Once the traffic light becomes yellow, it will eventually become green
  $\Box(\mathit{yellow} \rightarrow \Diamond \mathit{green})$

Semantics over Infinite Words

Given LTL formula φ over AP,

$$L(\varphi) = \{\sigma \in (2^{AP})^\omega \mid \sigma \models \varphi\}$$

Let $\sigma = A_0 A_1 A_2 \ldots$, with $A_i \subseteq AP$.

Semantics over Infinite Words

Given LTL formula φ over AP,

$$L(\varphi) = \{\sigma \in (2^{AP})^\omega \mid \sigma \models \varphi\}$$

Let $\sigma = A_0 A_1 A_2 \ldots$, with $A_i \subseteq AP$.

- $\sigma \models a$ iff $a \in A_0$
- $\sigma \models \varphi_1 \wedge \varphi_2$ iff $\sigma \models \varphi_1$ and $\sigma \models \varphi_2$
- $\sigma \models \neg\varphi$ iff $\sigma \not\models \varphi$
- $\sigma \models \bigcirc\varphi$ iff $A_1 A_2 \ldots \models \varphi$
- $\sigma \models \varphi \,\mathsf{U} \psi$ iff
  $\exists j \geqslant 0$ such that $A_j A_{j+1} \ldots \models \psi \wedge \forall 0 \leqslant i < j, A_i A_{i+1} \ldots \models \varphi$

Semantics over Infinite Words

Given LTL formula φ over AP,

$$L(\varphi) = \{\sigma \in (2^{AP})^\omega \mid \sigma \models \varphi\}$$

Let $\sigma = A_0 A_1 A_2 \ldots$, with $A_i \subseteq AP$.

- $\sigma \models \Diamond\varphi$ iff $\exists j \geqslant 0,\ A_j A_{j+1} \ldots \models \varphi$
- $\sigma \models \Box\varphi$ iff $\forall j \geqslant 0,\ A_j A_{j+1} \ldots \models \varphi$
- $\sigma \models \Box\Diamond\varphi$ iff $\forall j \geqslant 0,\ \exists i \geqslant j,\ A_i A_{i+1} \ldots \models \varphi$
- $\sigma \models \Diamond\Box\varphi$ iff $\exists j \geqslant 0,\ \forall i \geqslant j,\ A_i A_{i+1} \ldots \models \varphi$

If $\sigma = A_0 A_1 A_2 \ldots$, $\sigma \models \varphi$ is also written as $\sigma, 0 \models \varphi$. This simply means $A_0 A_1 A_2 \ldots \models \varphi$. One can also define $\sigma, i \models \varphi$ to mean $A_i A_{i+1} A_{i+2} \ldots \models \varphi$ to talk about a suffix of the word σ satisfying a property.

Transition System Semantics $TS \models \varphi$

Let $TS = (S, S_0, \rightarrow, AP, L)$ be a transition system, and φ an LTL formula over AP

- For an infinite path fragment π of TS,

$$\pi \models \varphi \text{ iff } \mathit{trace}(\pi) \models \varphi$$

Transition System Semantics $TS \models \varphi$

Let $TS = (S, S_0, \rightarrow, AP, L)$ be a transition system, and φ an LTL formula over AP

- For an infinite path fragment π of TS,

$$\pi \models \varphi \text{ iff } \mathit{trace}(\pi) \models \varphi$$

- For $s \in S$,

$$s \models \varphi \text{ iff } \forall \pi \in \mathit{Paths}(s), \pi \models \varphi$$

Transition System Semantics $TS \models \varphi$

Let $TS = (S, S_0, \rightarrow, AP, L)$ be a transition system, and φ an LTL formula over AP

- For an infinite path fragment π of TS,

$$\pi \models \varphi \text{ iff } \mathit{trace}(\pi) \models \varphi$$

- For $s \in S$,

$$s \models \varphi \text{ iff } \forall \pi \in \mathit{Paths}(s), \pi \models \varphi$$

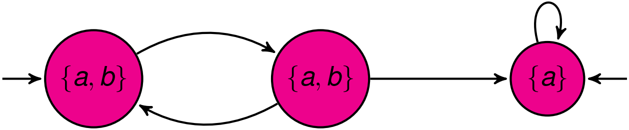
- $TS \models \varphi$ iff $\mathit{Traces}(TS) \subseteq L(\varphi)$

Transition System Semantics $TS \models \varphi$

Assume all states in TS are reachable from S_0.

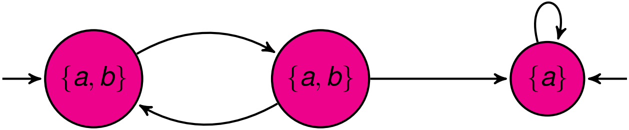
- $TS \models \varphi$ iff $TS \models L(\varphi)$ iff $Traces(TS) \subseteq L(\varphi)$
- $TS \models L(\varphi)$ iff $\pi \models \varphi \; \forall \pi \in Paths(TS)$
- $\pi \models \varphi \; \forall \pi \in Paths(TS)$ iff $s_0 \models \varphi \; \forall s_0 \in S_0$

Example



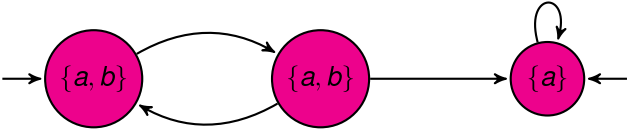
- $TS \models \Box a$,

Example



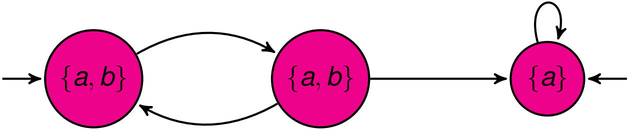
- $TS \models \Box a$,
- $TS \not\models \bigcirc(a \wedge b)$

Example



- $TS \models \Box a$,
- $TS \not\models \bigcirc(a \wedge b)$
- $TS \not\models (b \, \mathsf{U} (a \wedge \neg b))$

Example



- $TS \models \Box a$,
- $TS \not\models \bigcirc(a \wedge b)$
- $TS \not\models (b \,\mathsf{U} (a \wedge \neg b))$
- $TS \models \Box(\neg b \rightarrow \Box(a \wedge \neg b))$

More Semantics

- For paths π, $\pi \models \varphi$ iff $\pi \not\models \neg\varphi$

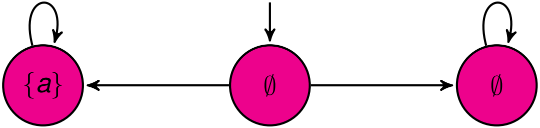
More Semantics

- For paths π, $\pi \models \varphi$ iff $\pi \not\models \neg\varphi$
  $\mathit{trace}(\pi) \in L(\varphi)$ iff $\mathit{trace}(\pi) \notin L(\neg\varphi) = \overline{L(\varphi)}$
- $TS \not\models \varphi$ iff $TS \models \neg\varphi$?

More Semantics

- For paths π, $\pi \models \varphi$ iff $\pi \not\models \neg\varphi$
  trace$(\pi) \in L(\varphi)$ iff *trace*$(\pi) \notin L(\neg\varphi) = \overline{L(\varphi)}$
- $TS \not\models \varphi$ iff $TS \models \neg\varphi$?
  - $TS \models \neg\varphi \rightarrow \forall$ paths π of TS, $\pi \models \neg\varphi$
  - Thus, $\forall\pi$, $\pi \not\models \varphi$. Hence, $TS \not\models \varphi$
  - Now assume $TS \not\models \varphi$
  - Then $\exists$ some path π in TS such that $\pi \models \neg\varphi$
  - However, there could be another path π' such that $\pi' \models \varphi$
  - Then $TS \not\models \neg\varphi$ as well
- Thus, $TS \not\models \varphi \not\equiv TS \models \neg\varphi$.

An Example



$TS \not\models \Diamond a$ and $TS \not\models \Box\neg a$

Equivalence of LTL Formulae

Equivalence

φ and ψ are equivalent ($\varphi \equiv \psi$) iff $L(\varphi) = L(\psi)$.

Expansion Laws

- $\varphi \, \mathsf{U} \psi \equiv \psi \vee (\varphi \wedge \bigcirc(\varphi \, \mathsf{U} \psi))$
- $\Diamond\varphi \equiv \varphi \vee \bigcirc\Diamond\varphi$
- $\Box\varphi \equiv \varphi \wedge \bigcirc\Box\varphi$

Equivalence of LTL Formulae

φ and ψ are equivalent iff $L(\varphi) = L(\psi)$.

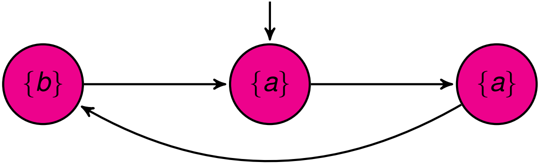
Equivalence of LTL Formulae

φ and ψ are equivalent iff $L(\varphi) = L(\psi)$.

Distribution

$\bigcirc(\varphi \vee \psi) \equiv \bigcirc\varphi \vee \bigcirc\psi$,
$\bigcirc(\varphi \wedge \psi) \equiv \bigcirc\varphi \wedge \bigcirc\psi$,
$\bigcirc(\varphi \,\mathsf{U}\, \psi) \equiv (\bigcirc\varphi) \,\mathsf{U}\, (\bigcirc\psi)$,
$\Diamond(\varphi \vee \psi) \equiv \Diamond\varphi \vee \Diamond\psi$,
$\Box(\varphi \wedge \psi) \equiv \Box\varphi \wedge \Box\psi$

Equivalence of LTL Formulae



$$TS \models \Diamond a \wedge \Diamond b,\ TS \not\models \Diamond(a \wedge b)$$

$$TS \models \Box(a \vee b),\ TS \not\models \Box a \vee \Box b$$

Satisfiability, Model Checking of LTL

Two Questions

Given transition system *TS*, and an LTL formula φ. Does $TS \models \varphi$?
Given an LTL formula φ, is $L(\varphi) = \emptyset$?

ω-automata

An ω-automaton is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, Acc)$ where

- Q is a finite set of states
- Σ is a finite alphabet
- $\delta : Q \times \Sigma \to 2^Q$ is a state transition function (if non-deterministic, otherwise, $\delta : Q \times \Sigma \to Q$)
- $q_0 \in Q$ is an initial state and Acc is an acceptance condition

ω-automata

An ω-automaton is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, Acc)$ where

- Q is a finite set of states
- Σ is a finite alphabet
- $\delta : Q \times \Sigma \to 2^Q$ is a state transition function (if non-deterministic, otherwise, $\delta : Q \times \Sigma \to Q$)
- $q_0 \in Q$ is an initial state and Acc is an acceptance condition

Run

A run ρ of $\mathcal{A}$ on an ω-word $\alpha = a_1 a_2 \cdots \in \Sigma^\omega$ is an infinite state sequence $\rho(0)\rho(1)\rho(2)\ldots$ such that

- $\rho(0) = q_0$,
- $\rho(i) = \delta(\rho(i-1), a_i)$ if $\mathcal{A}$ is deterministic,
- $\rho(i) \in \delta(\rho(i-1), a_i)$ if $\mathcal{A}$ is non-deterministic,

ω-automata

An ω-automaton is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, Acc)$ where

- Q is a finite set of states
- Σ is a finite alphabet
- $\delta : Q \times \Sigma \to 2^Q$ is a state transition function (if non-deterministic, otherwise, $\delta : Q \times \Sigma \to Q$)
- $q_0 \in Q$ is an initial state and Acc is an acceptance condition

Run

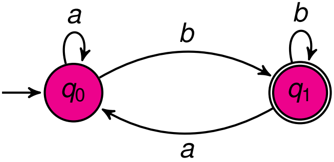
A run ρ of $\mathcal{A}$ on an ω-word $\alpha = a_1 a_2 \cdots \in \Sigma^\omega$ is an infinite state sequence $\rho(0)\rho(1)\rho(2)\ldots$ such that

- $\rho(0) = q_0$,
- $\rho(i) = \delta(\rho(i-1), a_i)$ if $\mathcal{A}$ is deterministic,
- $\rho(i) \in \delta(\rho(i-1), a_i)$ if $\mathcal{A}$ is non-deterministic,

Büchi Acceptance

For Büchi Acceptance, Acc is specified as a set of states, $G \subseteq Q$. The ω-word α is accepted if there is a run ρ of α such that $Inf(\rho) \cap G \neq \emptyset$.

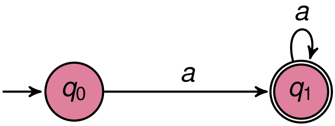
ω-Automata with Büchi Acceptance

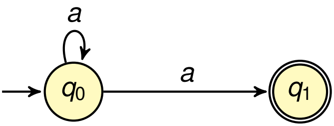


$L(\mathcal{A}) = \{\alpha \in \Sigma^\omega \mid \alpha$ has a run ρ such that $Inf(\rho) \cap G \neq \emptyset$

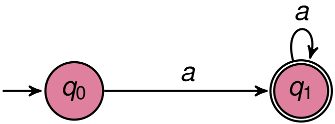
Language accepted=Infinitely many b's.

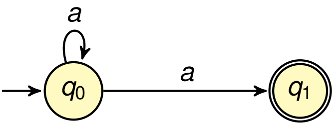
Comparing NFA and NBA

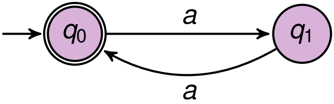


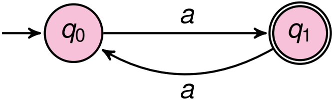


Comparing NFA and NBA

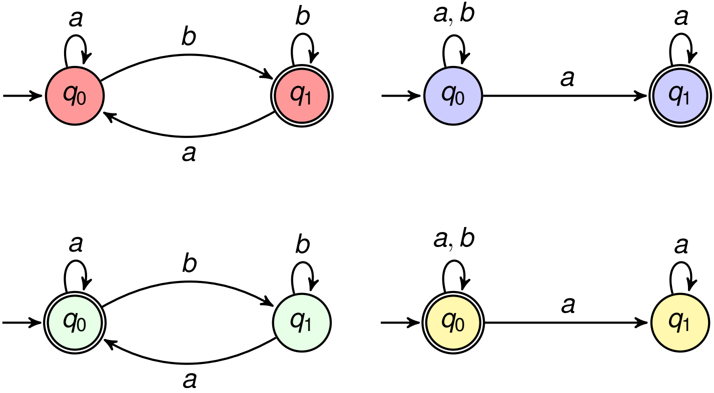








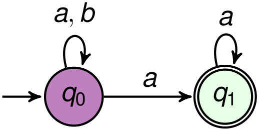
ω-Automata with Büchi Acceptance



- Left (T-B): Inf many *b*'s, Inf many *a*'s
- Right (T-B): Finitely many *b*'s, $(a + b)^\omega$

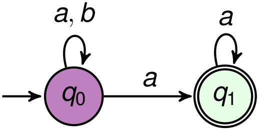
NBA and DBA

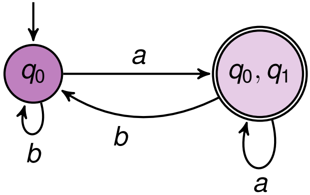
- Is every DBA as expressible as a NBA, like in the case of DFA and NFA?
- Can we do subset construction on NBA and obtain DBA?



NBA and DBA

- Is every DBA as expressible as a NBA, like in the case of DFA and NFA?
- Can we do subset construction on NBA and obtain DBA?



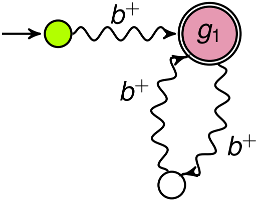


NBA and DBA

There does not exist a deterministic Büchi automata capturing the language finitely many *a*'s.

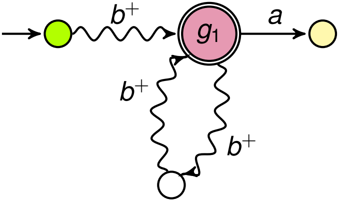
NBA and DBA

There does not exist a deterministic Büchi automata capturing the language finitely many *a*'s.



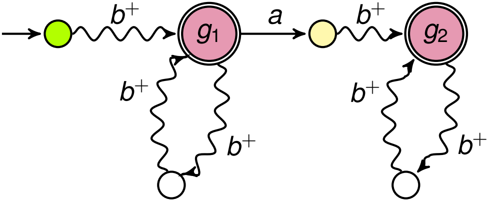
NBA and DBA

There does not exist a deterministic Büchi automata capturing the language finitely many a's.



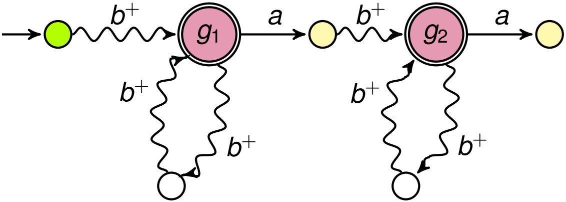
NBA and DBA

There does not exist a deterministic Büchi automata capturing the language finitely many a's.



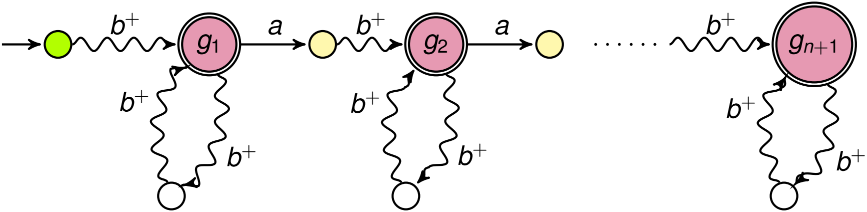
NBA and DBA

There does not exist a deterministic Büchi automata capturing the language finitely many *a*'s.



NBA and DBA

There does not exist a deterministic Büchi automata capturing the language finitely many a's.



CS 228 : Logic in Computer Science

Krishna. S

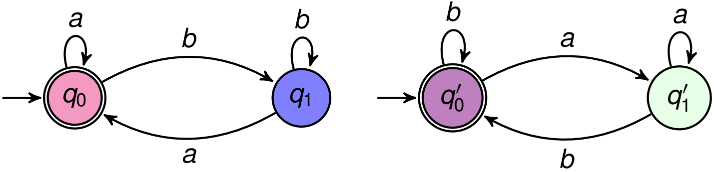
So Far

- ω-automata with Büchi acceptance, also called Büchi automata
- Non-determinism versus determinism

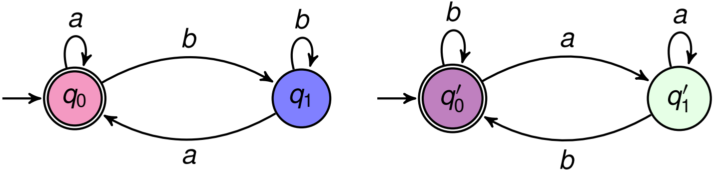
Büchi Acceptance

A language $L \subseteq \Sigma^\omega$ is called ω-regular if there exists a NBA $\mathcal{A}$ such that $L = L(\mathcal{A})$.

Union and Intersection of NBA

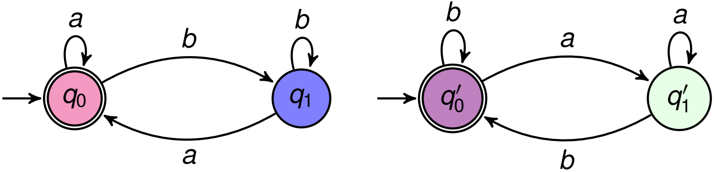


Union and Intersection of NBA



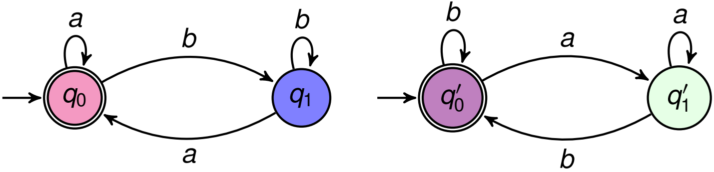
- States as $Q_1 \times Q_2 \times \{1, 2\}$, start state $(q_0, q_0', 1)$

Union and Intersection of NBA



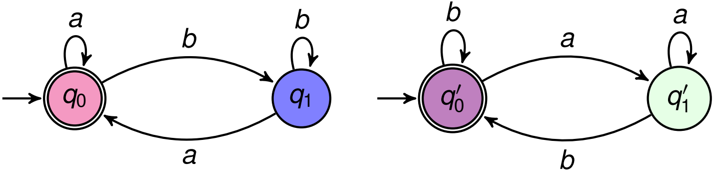
- States as $Q_1 \times Q_2 \times \{1, 2\}$, start state $(q_0, q'_0, 1)$
- $(q_1, q_2, 1) \xrightarrow{a} (q'_1, q'_2, 1)$ if $q_1 \xrightarrow{a} q'_1$ and $q_2 \xrightarrow{a} q'_2$ and $q_1 \notin G_1$
- $(q_1, q_2, 1) \xrightarrow{a} (q'_1, q'_2, 2)$ if $q_1 \xrightarrow{a} q'_1$ and $q_2 \xrightarrow{a} q'_2$ and $q_1 \in G_1$

Union and Intersection of NBA



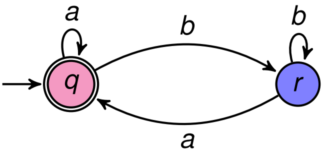
- States as $Q_1 \times Q_2 \times \{1, 2\}$, start state $(q_0, q'_0, 1)$
- $(q_1, q_2, 1) \xrightarrow{a} (q'_1, q'_2, 1)$ if $q_1 \xrightarrow{a} q'_1$ and $q_2 \xrightarrow{a} q'_2$ and $q_1 \notin G_1$
- $(q_1, q_2, 1) \xrightarrow{a} (q'_1, q'_2, 2)$ if $q_1 \xrightarrow{a} q'_1$ and $q_2 \xrightarrow{a} q'_2$ and $q_1 \in G_1$
- $(q_1, q_2, 2) \xrightarrow{a} (q'_1, q'_2, 2)$ if $q_1 \xrightarrow{a} q'_1$ and $q_2 \xrightarrow{a} q'_2$ and $q_2 \notin G_2$
- $(q_1, q_2, 2) \xrightarrow{a} (q'_1, q'_2, 1)$ if $q_1 \xrightarrow{a} q'_1$ and $q_2 \xrightarrow{a} q'_2$ and $q_2 \in G_2$

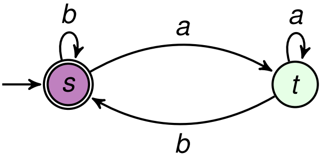
Union and Intersection of NBA



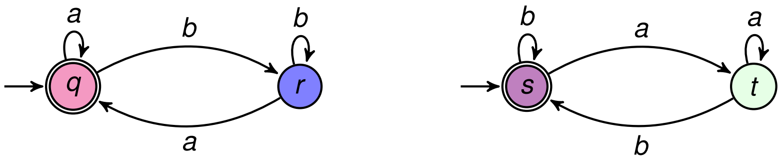
- States as $Q_1 \times Q_2 \times \{1, 2\}$, start state $(q_0, q'_0, 1)$
- $(q_1, q_2, 1) \xrightarrow{a} (q'_1, q'_2, 1)$ if $q_1 \xrightarrow{a} q'_1$ and $q_2 \xrightarrow{a} q'_2$ and $q_1 \notin G_1$
- $(q_1, q_2, 1) \xrightarrow{a} (q'_1, q'_2, 2)$ if $q_1 \xrightarrow{a} q'_1$ and $q_2 \xrightarrow{a} q'_2$ and $q_1 \in G_1$
- $(q_1, q_2, 2) \xrightarrow{a} (q'_1, q'_2, 2)$ if $q_1 \xrightarrow{a} q'_1$ and $q_2 \xrightarrow{a} q'_2$ and $q_2 \notin G_2$
- $(q_1, q_2, 2) \xrightarrow{a} (q'_1, q'_2, 1)$ if $q_1 \xrightarrow{a} q'_1$ and $q_2 \xrightarrow{a} q'_2$ and $q_2 \in G_2$
- Good states=$Q_1 \times G_2 \times \{2\}$ or $G_1 \times Q_2 \times \{1\}$

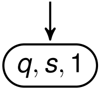
Union and Intersection of NBA



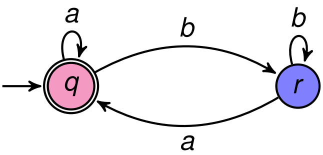


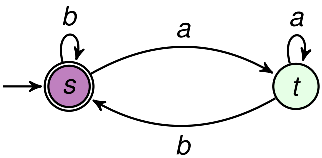
Union and Intersection of NBA

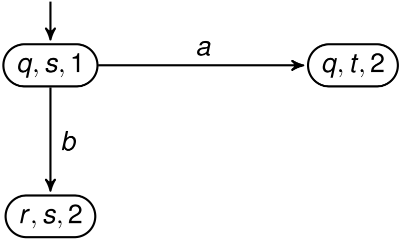




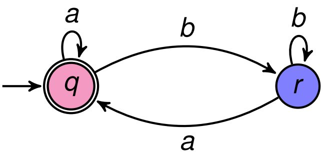
Union and Intersection of NBA

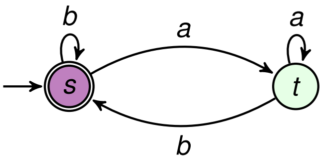


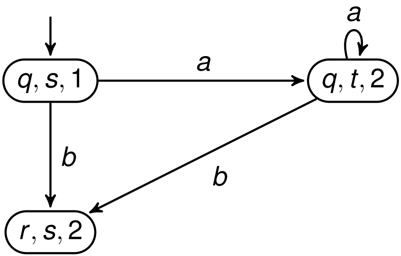




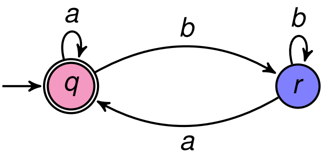
Union and Intersection of NBA

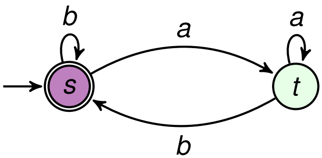


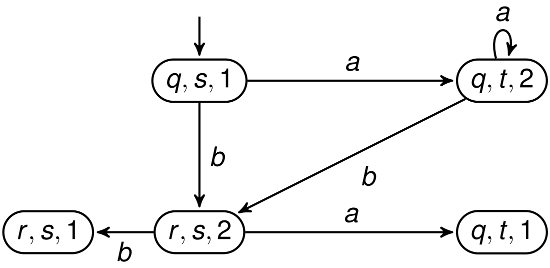




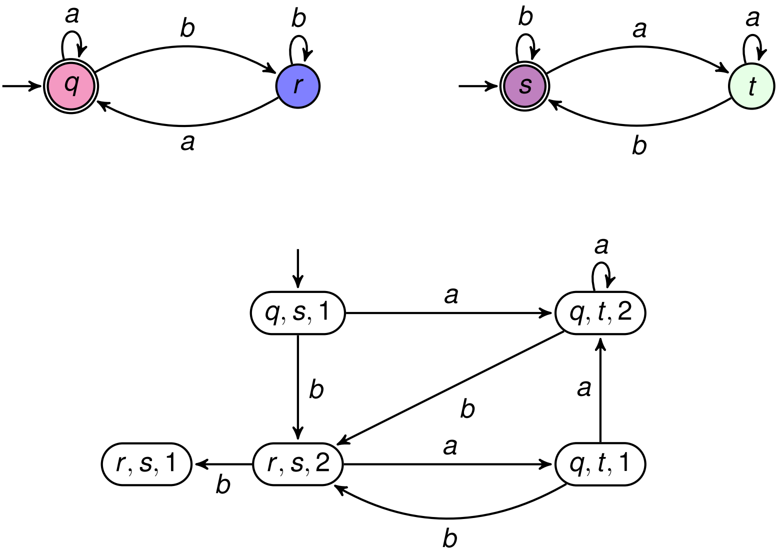
Union and Intersection of NBA



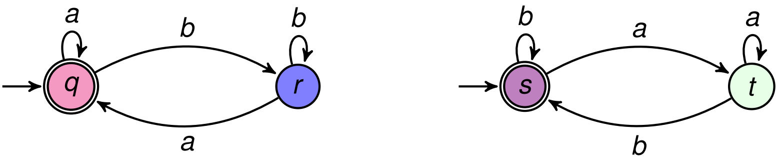


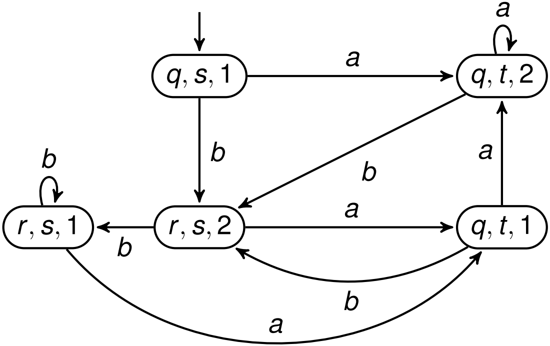


Union and Intersection of NBA

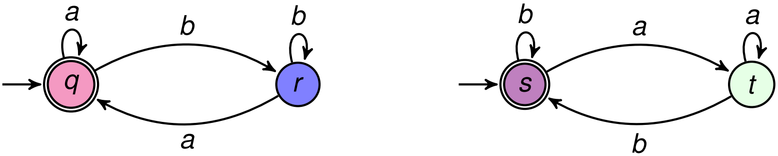


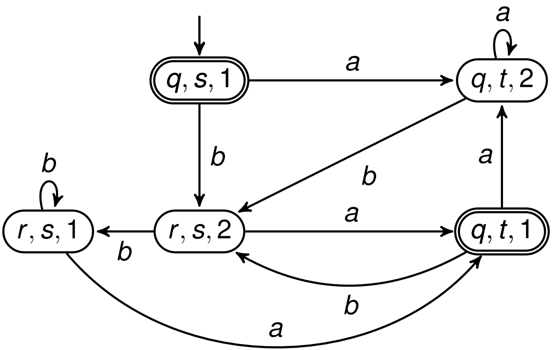
Union and Intersection of NBA





Union and Intersection of NBA



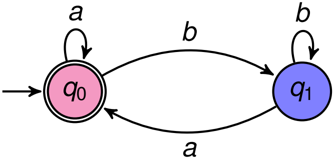


Emptiness

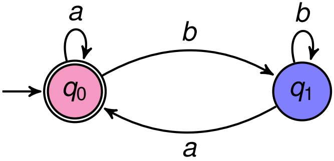
Given an NBA/DBA $\mathcal{A}$, how do you check if $L(\mathcal{A}) = \emptyset$?

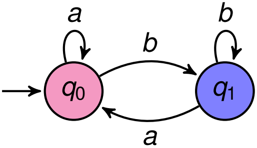
- Enumerate SCCs
- Check if there is an SCC containing a good state

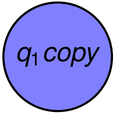
Complementation of DBA



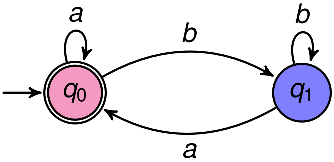
Complementation of DBA

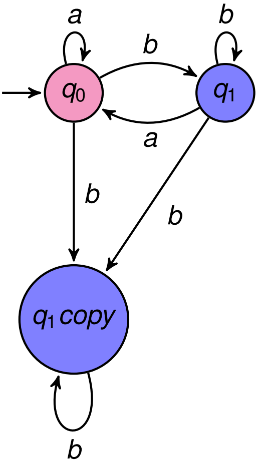




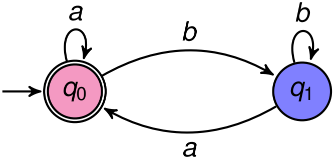


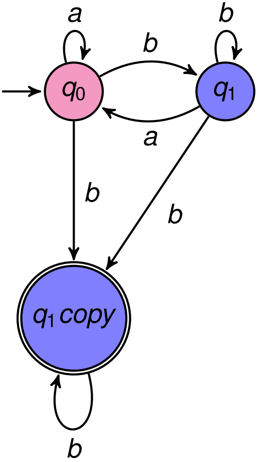
Complementation of DBA





Complementation of DBA





Complementation of DBA

- Given $\mathcal{A}$ is a DBA, and $w \notin L(\mathcal{A})$, then after some finite prefix, the unique run of w settles in bad states.
- Idea for complement: “copy” states of $Q - G$, once you enter this block, you stay there.
- View this as the set of good states, any word w that was rejected by $\mathcal{A}$ has two possible runs in this automaton: the original run, and one another, that will settle in the $Q - G$ copy, and will be accepted.
- What we get now is an NBA for $\overline{L(\mathcal{A})}$, not a DBA.

Complementing NBA non-trivial, can be done.

Normal Form for ω-regular languages

An ω-regular language $L \subseteq \Sigma^\omega$ can be written as $L = \bigcup_{i=1}^{n} U_i V_i^\omega$, where U_i, V_i are regular languages.

One direction : Assume L is accepted by an NBA/DBA.

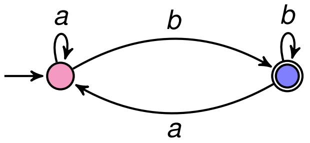
- Define $U_g = \{w \in \Sigma^* \mid q_0 \xrightarrow{w} g\}$
- Define $V_g = \{w \in \Sigma^* \mid g \xrightarrow{w} g\}$
- Then $L = \bigcup_{g \in G} U_g V_g^\omega$, where U_g, V_g are regular
- Show that U_g, V_g are regular.

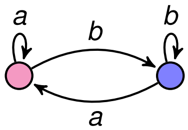
Normal Form for ω-regular languages

An ω-regular language $L \subseteq \Sigma^\omega$ can be written as $L = \bigcup_{i=1}^{n} U_i V_i^\omega$, where U_i, V_i are regular languages.

Other direction : Assume $L = \bigcup_{i=1}^{n} U_i V_i^\omega$. Show that L is accepted by an NBA/DBA.

1. If V is regular, V^ω is ω-regular



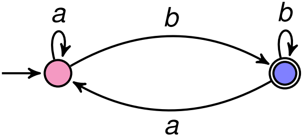


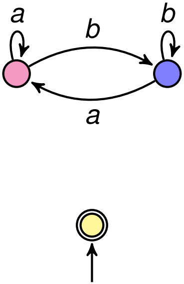
Normal Form for ω-regular languages

An ω-regular language $L \subseteq \Sigma^\omega$ can be written as $L = \bigcup_{i=1}^{n} U_i V_i^\omega$, where U_i, V_i are regular languages.

Other direction : Assume $L = \bigcup_{i=1}^{n} U_i V_i^\omega$. Show that L is accepted by an NBA/DBA.

1. If V is regular, V^ω is ω-regular



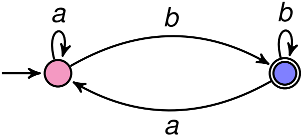


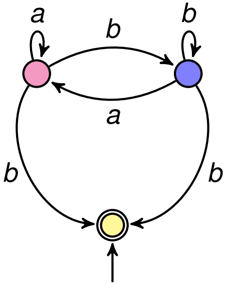
Normal Form for ω-regular languages

An ω-regular language $L \subseteq \Sigma^\omega$ can be written as $L = \bigcup_{i=1}^{n} U_i V_i^\omega$, where U_i, V_i are regular languages.

Other direction : Assume $L = \bigcup_{i=1}^{n} U_i V_i^\omega$. Show that L is accepted by an NBA/DBA.

1. If V is regular, V^ω is ω-regular



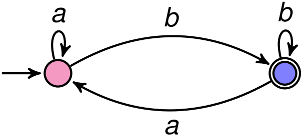


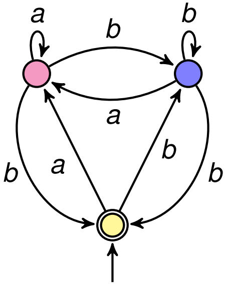
Normal Form for ω-regular languages

An ω-regular language $L \subseteq \Sigma^\omega$ can be written as $L = \bigcup_{i=1}^{n} U_i V_i^\omega$, where U_i, V_i are regular languages.

Other direction : Assume $L = \bigcup_{i=1}^{n} U_i V_i^\omega$. Show that L is accepted by an NBA/DBA.

1. If V is regular, V^ω is ω-regular





Normal Form for ω-regular languages

1. If V is regular, V^ω is ω-regular
   - Let $D = (Q, \Sigma, q_0, \delta, F)$ be a DFA accepting V
   - Construct NBA $E = (Q \cup \{p_0\}, \Sigma, p_0, \Delta, G)$ such that $G = \{p_0\}$,
   - $\Delta = \delta \cup \{p_0 \in \Delta(q, a) \mid \delta(q, a) \in F\} \cup \{\Delta(p_0, a) = s \mid \delta(q_0, a) = s\}$
2. Show that if U is regular and V^ω is ω-regular, then UV^ω is ω-regular
   - $D = (Q_1, \Sigma, q_0, \delta_1, F)$ be a DFA, $L(D) = U$ and $E = (Q_2, \Sigma, q_0', \delta_2, G)$ be an NBA, $L(E) = V^\omega$.
   - $A = (Q_1 \cup Q_2, \Sigma, q_0, \delta', G)$ NBA such that $\delta' = \delta_1 \cup \delta_2 \cup \{(q, a, q_0') \mid \delta_1(q, a) \in F\}$

CS 228 : Logic in Computer Science

Krishna. S

Union of NBA/DBA

Normal Form for ω-regular languages

An ω-regular language $L \subseteq \Sigma^\omega$ can be written as $L = \bigcup_{i=1}^{n} U_i V_i^\omega$, where U_i, V_i are regular languages.

One direction : Assume L is accepted by an NBA/DBA.

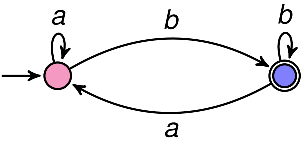
- Define $U_g = \{w \in \Sigma^* \mid q_0 \xrightarrow{w} g\}$
- Define $V_g = \{w \in \Sigma^* \mid g \xrightarrow{w} g\}$
- Then $L = \bigcup_{g \in G} U_g V_g^\omega$, where U_g, V_g are regular
- Show that U_g, V_g are regular.

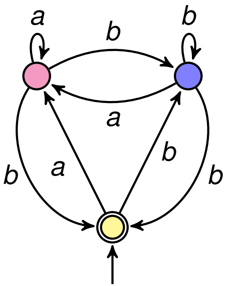
Normal Form for ω-regular languages

An ω-regular language $L \subseteq \Sigma^\omega$ can be written as $L = \bigcup_{i=1}^{n} U_i V_i^\omega$, where U_i, V_i are regular languages.

Other direction : Assume $L = \bigcup_{i=1}^{n} U_i V_i^\omega$. Show that L is accepted by an NBA/DBA.

1. If V is regular, V^ω is ω-regular





Normal Form for ω-regular languages

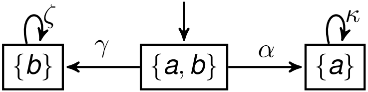
1. If V is regular, V^ω is ω-regular
   - Let $D = (Q, \Sigma, q_0, \delta, F)$ be a DFA accepting V
   - Construct NBA $E = (Q \cup \{p_0\}, \Sigma, p_0, \Delta, G)$ such that $G = \{p_0\}$,
   - $\Delta = \delta \cup \{p_0 \in \Delta(q, a) \mid \delta(q, a) \in F\} \cup \{\Delta(p_0, a) = s \mid \delta(q_0, a) = s\}$
2. Show that if U is regular and V^ω is ω-regular, then UV^ω is ω-regular
   - $D = (Q_1, \Sigma, q_0, \delta_1, F)$ be a DFA, $L(D) = U$ and $E = (Q_2, \Sigma, q_0', \delta_2, G)$ be an NBA, $L(E) = V^\omega$.
   - $A = (Q_1 \cup Q_2, \Sigma, q_0, \delta', G)$ NBA such that $\delta' = \delta_1 \cup \delta_2 \cup \{(q, a, q_0') \mid \delta_1(q, a) \in F\}$

LTL ModelChecking

- Given transition system TS, and LTL formula φ, does $TS \models \varphi$?
- $Tr(TS) \subseteq L(\varphi)$ iff $Tr(TS) \cap \overline{L(\varphi)} = \emptyset$
- First construct NBA $A_{\neg\varphi}$ for $\neg\varphi$.
- Construct product of TS and $A_{\neg\varphi}$, obtaining a new TS, say TS'.
- Check some very simple property on TS', to check $TS \models \varphi$.

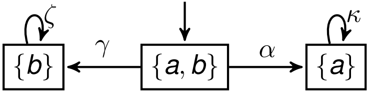
An Example $TS \models \varphi$

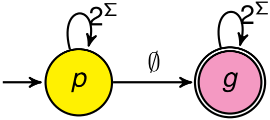
- Let $\varphi = \Box(a \vee b), \neg\varphi = \Diamond(\neg a \wedge \neg b)$
- Take TS and $A_{\neg\varphi}$, and check the intersection.



An Example $TS \models \varphi$

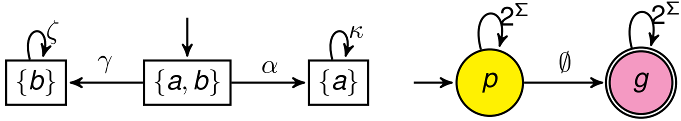
- Let $\varphi = \square(a \vee b), \neg\varphi = \Diamond(\neg a \wedge \neg b)$
- Take TS and $A_{\neg\varphi}$, and check the intersection.

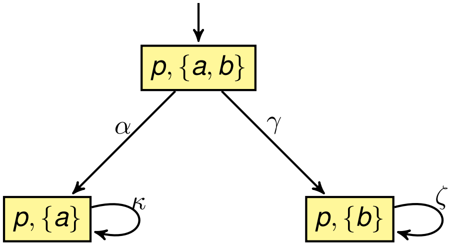




An Example $TS \models \varphi$

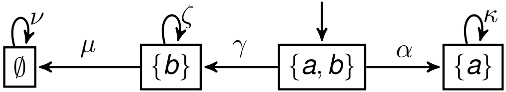
- Let $\varphi = \Box(a \vee b)$, $\neg\varphi = \Diamond(\neg a \wedge \neg b)$
- Take TS and $A_{\neg\varphi}$, and check the intersection.





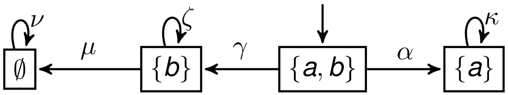
An Example : $TS \not\models \varphi$

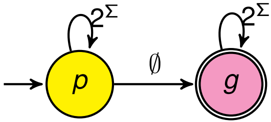
- Let $\varphi = \Box(a \vee b), \neg\varphi = \Diamond(\neg a \wedge \neg b)$
- Take TS and $A_{\neg\varphi}$, and check the intersection.



An Example : $TS \not\models \varphi$

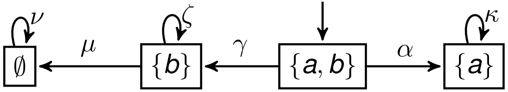
- Let $\varphi = \Box(a \vee b), \neg\varphi = \Diamond(\neg a \wedge \neg b)$
- Take TS and $A_{\neg\varphi}$, and check the intersection.

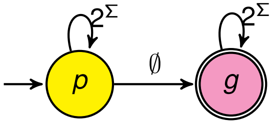


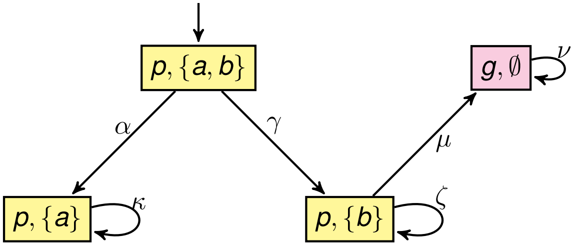


An Example : $TS \not\models \varphi$

- Let $\varphi = \Box(a \vee b), \neg\varphi = \Diamond(\neg a \wedge \neg b)$
- Take TS and $A_{\neg\varphi}$, and check the intersection.







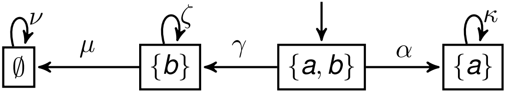
Product of TS and NBA

Given $TS = (S, Act, I, AP, L)$ and $\mathcal{A} = (Q, 2^{AP}, \delta, Q_0, G)$ NBA.
Define $TS \otimes \mathcal{A} = (S \times Q, Act, I', AP', L')$ such that

- $I' = \{(s_0, q) \mid s_0 \in I \text{ and } \exists q_0 \in Q_0, q_0 \stackrel{L(s_0)}{\rightarrow} q\}$
- $AP' = Q$, $L' : S \times Q \rightarrow 2^Q$ such that $L'((s, q)) = \{q\}$
- If $s \stackrel{\alpha}{\rightarrow} t$ and $q \stackrel{L(t)}{\rightarrow} p$, then $(s, q) \stackrel{\alpha}{\rightarrow} (t, p)$

Persistence Properties

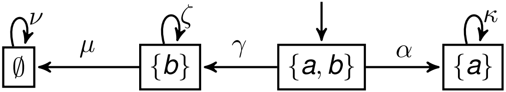
Let η be a propositional logic formula over AP. A persistence property P_{pers} has the form $\Diamond\Box\eta$. How will you check a persistence property on a TS?



- For example, $TS \not\models \Diamond\Box(a \vee b)$
- For example, $TS \models \Diamond\Box(a \vee (a \rightarrow b))$

Persistence Properties

Let η be a propositional logic formula over *AP*. A persistence property P_{pers} has the form $\Diamond\Box\eta$. How will you check a persistence property on a TS?



- For example, $TS \not\models \Diamond\Box(a \vee b)$
- For example, $TS \models \Diamond\Box(a \vee (a \rightarrow b))$
- $TS \not\models P_{pers}$ iff there is a reachable cycle in the TS containing a state with a label which satisfies $\neg\eta$.

LTL ModelChecking

- Given TS and LTL formula φ. Does $TS \models \varphi$?
- Construct $A_{\neg\varphi}$, and let $g_1, \ldots, g_n$ be the good states in $A_{\neg\varphi}$.
- Build $TS' = TS \otimes A_{\neg\varphi}$.
- The labels of TS' are the state names of $A_{\neg\varphi}$.
- Check if $TS' \models \Diamond\Box(\neg g_1 \wedge \ldots \neg g_n)$.

LTL ModelChecking

- Given TS and LTL formula φ. Does $TS \models \varphi$?
- Construct $A_{\neg\varphi}$, and let $g_1, \ldots, g_n$ be the good states in $A_{\neg\varphi}$.
- Build $TS' = TS \otimes A_{\neg\varphi}$.
- The labels of TS' are the state names of $A_{\neg\varphi}$.
- Check if $TS' \models \Diamond\Box(\neg g_1 \wedge \ldots \neg g_n)$.

ModelChecking LTL in TS = Check Persistence in TS'

The following are equivalent.

- $TS \models \varphi$
- $Tr(TS) \cap L(A_{\neg\varphi}) = \emptyset$
- $TS' \models \Diamond\Box(\neg g_1 \wedge \ldots \neg g_n)$.

CS 228 : Logic in Computer Science

Krishna. S

GNBA

- Generalized NBA, a variant of NBA
- Only difference is in acceptance condition
- Acceptance condition in GNBA is a set $\mathcal{F} = \{F_1, \ldots, F_k\}$, each $F_i \subseteq Q$
- An infinite run ρ is accepting in a GNBA iff

$$\forall F_i \in \mathcal{F}, \mathit{Inf}(\rho) \cap F_i \neq \emptyset$$

- Note that when $\mathcal{F} = \emptyset$, all infinite runs are accepting
- GNBA and NBA are equivalent in expressive power.

Word View (On the board)

- $w = \{a\}\{a, b\}\{\} \ldots,$
- $\varphi = a \, \mathsf{U}(\neg a \wedge b)$
- Subformulae of $\varphi = \{a, \neg a, b, \neg a \wedge b, \varphi\}$
- Parse trees to compute all subformulae

Closure of φ, $cl(\varphi)$

- $cl(\varphi)$=all subformulae of φ and their negations, identifying $\neg\neg\psi$ to be ψ.
- Example for $\varphi = a\,\mathrm{U}(\neg a \wedge b)$
- $cl(\varphi) = \{a, \neg a, b, \neg b, \neg a \wedge b, \neg(\neg a \wedge b), \varphi, \neg\varphi\}$

Elementary Sets

Let φ be an LTL formula. Then $B \subseteq cl(\varphi)$ is elementary provided:

- B is propositionally and maximally consistent : for all $\varphi_1 \wedge \varphi_2, \psi \in cl(\varphi)$,
  - $\varphi_1 \wedge \varphi_2 \in B \Leftrightarrow \varphi_1 \in B \wedge \varphi_2 \in B$
  - $\psi \in B \Leftrightarrow \neg\psi \notin B$
  - $true \in cl(\varphi) \Rightarrow true \in B$
- B is locally consistent wrt U. That is, for all $\varphi_1 \mathsf{U} \varphi_2 \in cl(\varphi)$,
  - $\varphi_2 \in B \Rightarrow \varphi_1 \mathsf{U} \varphi_2 \in B$
  - $\varphi_1 \mathsf{U} \varphi_2 \in B, \varphi_2 \notin B \Rightarrow \varphi_1 \in B$
- B is elementary : B is propositionally, maximally and locally consistent
- Given a $B \subseteq cl(\varphi)$, how can you check if B is elementary?

Check Elementary

Let $\varphi = a \,\mathsf{U}(\neg a \wedge b)$

- $B_1 = \{a, b, \neg a \wedge b, \varphi\}$

Check Elementary

Let $\varphi = a \, \mathrm{U} (\neg a \wedge b)$

- $B_1 = \{a, b, \neg a \wedge b, \varphi\}$ No, propositionally inconsistent
- $B_2 = \{\neg a, b, \varphi\}$

Check Elementary

Let $\varphi = a \, \mathrm{U} (\neg a \wedge b)$

- $B_1 = \{a, b, \neg a \wedge b, \varphi\}$ No, propositionally inconsistent
- $B_2 = \{\neg a, b, \varphi\}$ No, not maximal as $\neg a \wedge b \notin B_2$, $\neg(\neg a \wedge b) \notin B_2$
- $B_3 = \{\neg a, b, \neg a \wedge b, \neg\varphi\}$

Check Elementary

Let $\varphi = a\,\mathrm{U}(\neg a \wedge b)$

- $B_1 = \{a, b, \neg a \wedge b, \varphi\}$ No, propositionally inconsistent
- $B_2 = \{\neg a, b, \varphi\}$ No, not maximal as $\neg a \wedge b \notin B_2$, $\neg(\neg a \wedge b) \notin B_2$
- $B_3 = \{\neg a, b, \neg a \wedge b, \neg\varphi\}$ No, not locally consistent for U
- $B_4 = \{\neg a, \neg b, \neg(\neg a \wedge b), \neg\varphi\}$

Check Elementary

Let $\varphi = a\,\mathsf{U}(\neg a \wedge b)$

- $B_1 = \{a, b, \neg a \wedge b, \varphi\}$ No, propositionally inconsistent
- $B_2 = \{\neg a, b, \varphi\}$ No, not maximal as $\neg a \wedge b \notin B_2$, $\neg(\neg a \wedge b) \notin B_2$
- $B_3 = \{\neg a, b, \neg a \wedge b, \neg\varphi\}$ No, not locally consistent for U
- $B_4 = \{\neg a, \neg b, \neg(\neg a \wedge b), \neg\varphi\}$ Yes, elementary

LTL φ to GNBA G_φ

- States of G_φ are elementary sets B_i
- For a word $w = A_0 A_1 A_2 \ldots$ the sequence of states $\sigma = B_0 B_1 B_2 \ldots$ will be a run for w
- σ will be accepting iff $w \models \varphi$ iff $\varphi \in B_0$
- In general, a run $B_i B_{i+1} \ldots$ for $A_i A_{i+1} \ldots$ is accepting iff $A_i A_{i+1} \ldots \models \psi$ for all $\psi \in B_i$.

LTL to GNBA

- Let $\varphi = \bigcirc a$.
- Subformulae of φ : $\{a, \bigcirc a\}$. Let $A = \{a, \bigcirc a, \neg a, \neg \bigcirc a\}$.
- Possibilities at each state
  - $\{a, \bigcirc a\}$
  - $\{\neg a, \bigcirc a\}$
  - $\{a, \neg \bigcirc a\}$
  - $\{\neg a, \neg \bigcirc a\}$
- Our initial state(s) must guarantee truth of $\bigcirc a$. Thus, initial states: $\{a, \bigcirc a\}$ and $\{\neg a, \bigcirc a\}$

LTL to GNBA

$\{a, \bigcirc a\}$

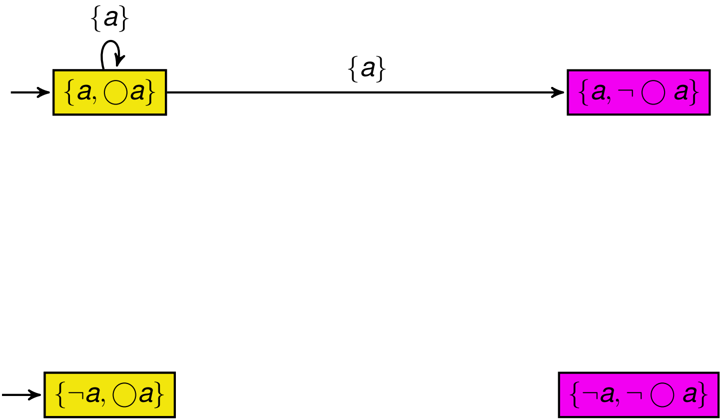
$\{a, \neg \bigcirc a\}$

$\{\neg a, \bigcirc a\}$

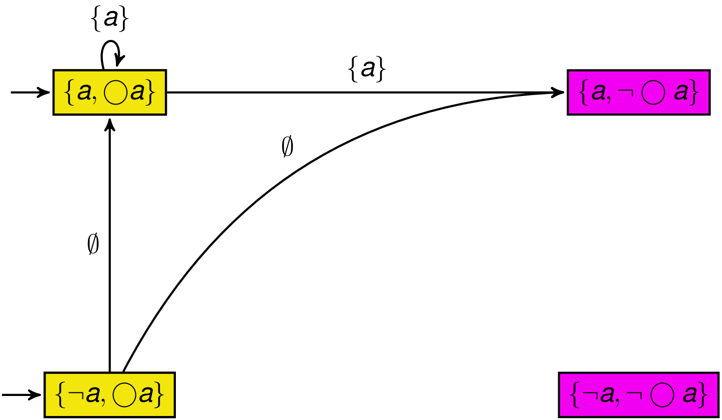
$\{\neg a, \neg \bigcirc a\}$

LTL to GNBA

LTL to GNBA

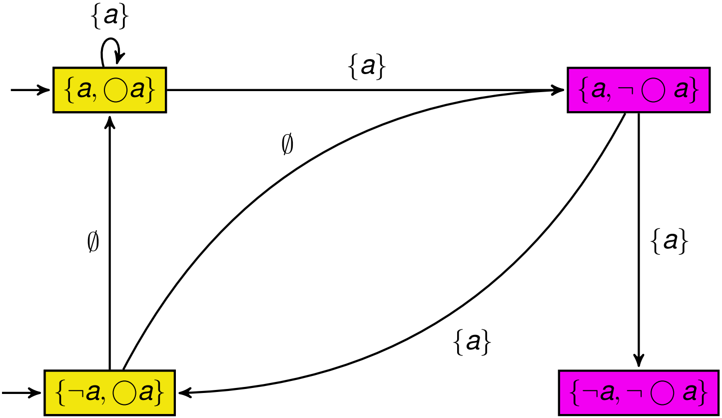

$\{a\}$
$\{a\}$
$\{a, \bigcirc a\}$
$\{a, \neg \bigcirc a\}$
$\{\neg a, \bigcirc a\}$
$\{\neg a, \neg \bigcirc a\}$

LTL to GNBA



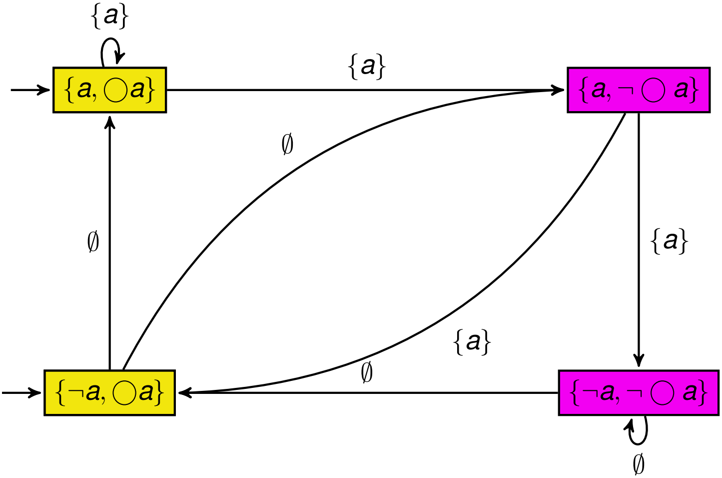
$\{a\}$
$\{a\}$
$\{a, \bigcirc a\}$
$\{a, \neg \bigcirc a\}$
$\emptyset$
$\emptyset$
$\{\neg a, \bigcirc a\}$
$\{\neg a, \neg \bigcirc a\}$

LTL to GNBA



$\{a\}$
$\{a, \bigcirc a\}$
$\{a\}$
$\{a, \neg \bigcirc a\}$
$\emptyset$
$\emptyset$
$\{a\}$
$\{a\}$
$\{\neg a, \bigcirc a\}$
$\{\neg a, \neg \bigcirc a\}$

LTL to GNBA



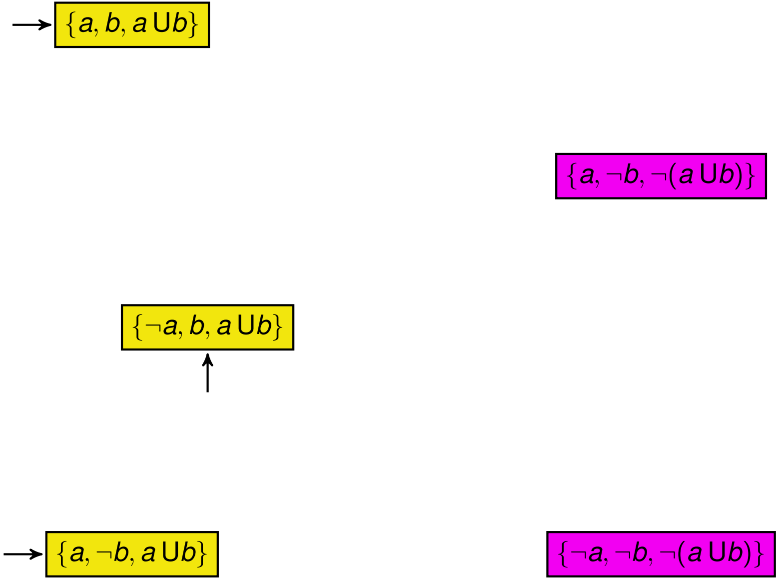
LTL to GNBA

- Claim : Runs from a state labelled set B indeed satisfy B
- No good states. All words having a run from a start state are accepted.
- Automaton for $\neg \bigcirc a$ same, except for the start states.

LTL to GNBA

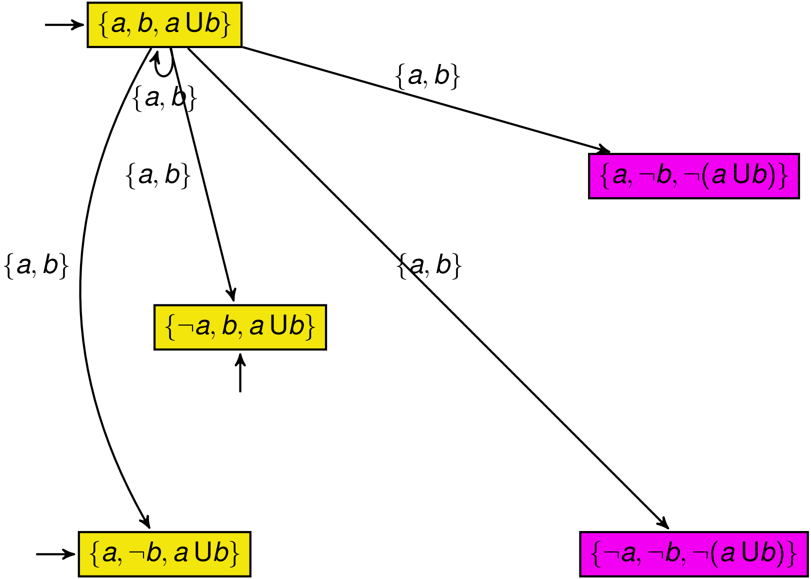
- Let $\varphi = a \,\mathsf{U} b$.
- Subformulae of φ : $\{a, b, a \,\mathsf{U} b\}$. Let $B = \{a, \neg a, b, \neg b, a \,\mathsf{U} b, \neg(a \,\mathsf{U} b)\}$.
- Possibilities at each state
  - $\{a, \neg b, a \,\mathsf{U} b\}$
  - $\{\neg a, b, a \,\mathsf{U} b\}$
  - $\{a, b, a \,\mathsf{U} b\}$
  - $\{a, \neg b, \neg(a \,\mathsf{U} b)\}$
  - $\{\neg a, \neg b, \neg(a \,\mathsf{U} b)\}$
- Our initial state(s) must guarantee truth of $a \,\mathsf{U} b$. Thus, initial states: $\{a, b, a \,\mathsf{U} b\}$ and $\{\neg a, b, a \,\mathsf{U} b\}$ and $\{a, \neg b, a \,\mathsf{U} b\}$.

LTL to GNBA

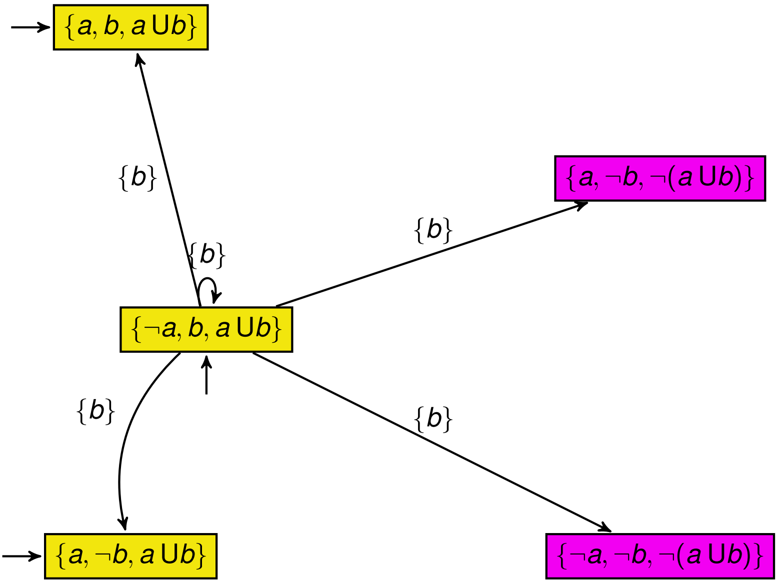


$\{a, b, a \,\mathsf{U}\, b\}$
$\{a, \neg b, \neg(a \,\mathsf{U}\, b)\}$
$\{\neg a, b, a \,\mathsf{U}\, b\}$
$\{a, \neg b, a \,\mathsf{U}\, b\}$
$\{\neg a, \neg b, \neg(a \,\mathsf{U}\, b)\}$

LTL to GNBA

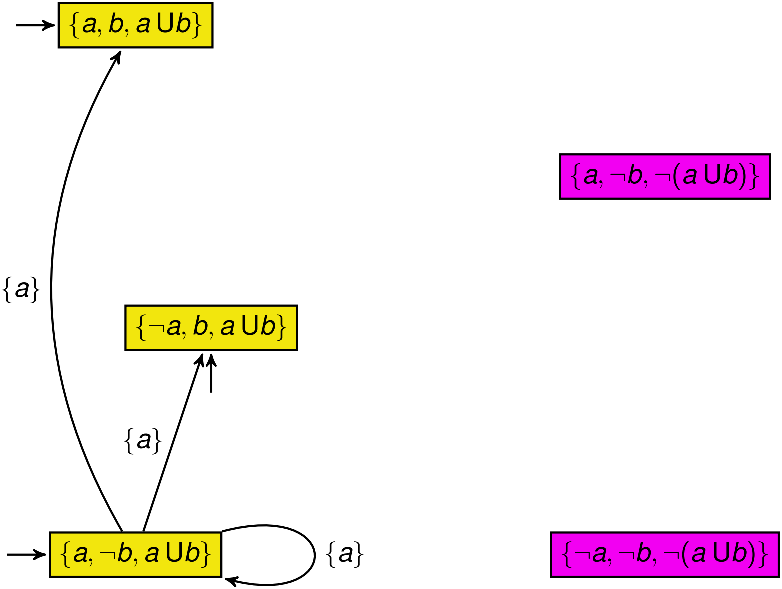


LTL to GNBA



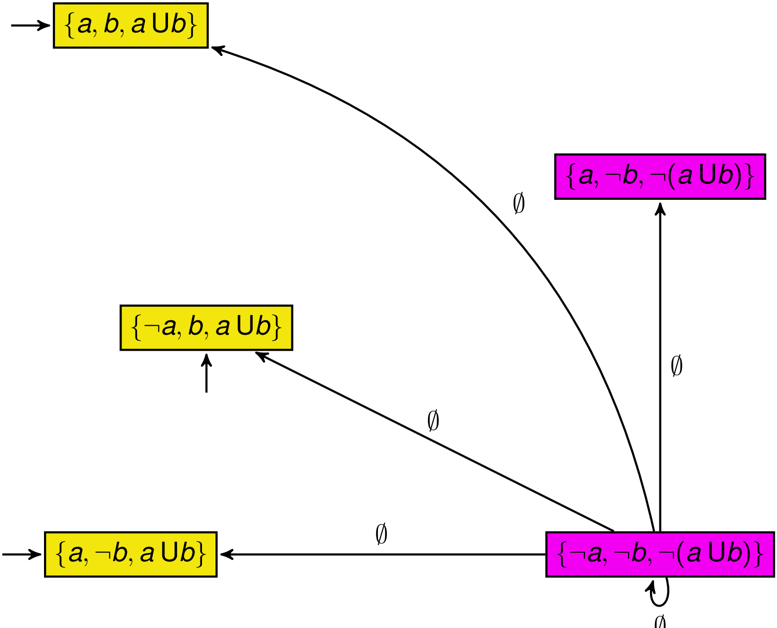
$\{a, b, a\,\mathsf{U}\,b\}$
$\{b\}$
$\{a, \neg b, \neg(a\,\mathsf{U}\,b)\}$
$\{b\}$
$\{b\}$
$\{\neg a, b, a\,\mathsf{U}\,b\}$
$\{b\}$
$\{b\}$
$\{a, \neg b, a\,\mathsf{U}\,b\}$
$\{\neg a, \neg b, \neg(a\,\mathsf{U}\,b)\}$

LTL to GNBA



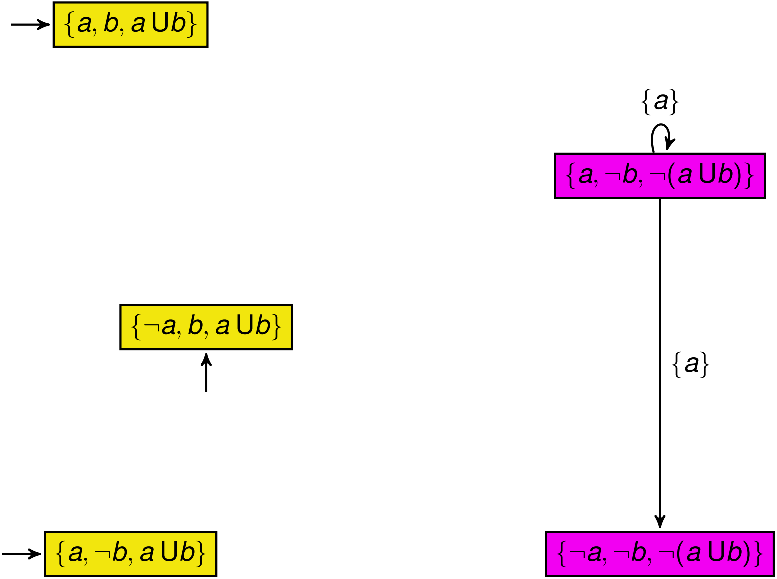
$\{a, b, a\,\mathsf{U}\,b\}$
$\{a, \neg b, \neg(a\,\mathsf{U}\,b)\}$
$\{a\}$
$\{\neg a, b, a\,\mathsf{U}\,b\}$
$\{a\}$
$\{a, \neg b, a\,\mathsf{U}\,b\}$
$\{a\}$
$\{\neg a, \neg b, \neg(a\,\mathsf{U}\,b)\}$

LTL to GNBA



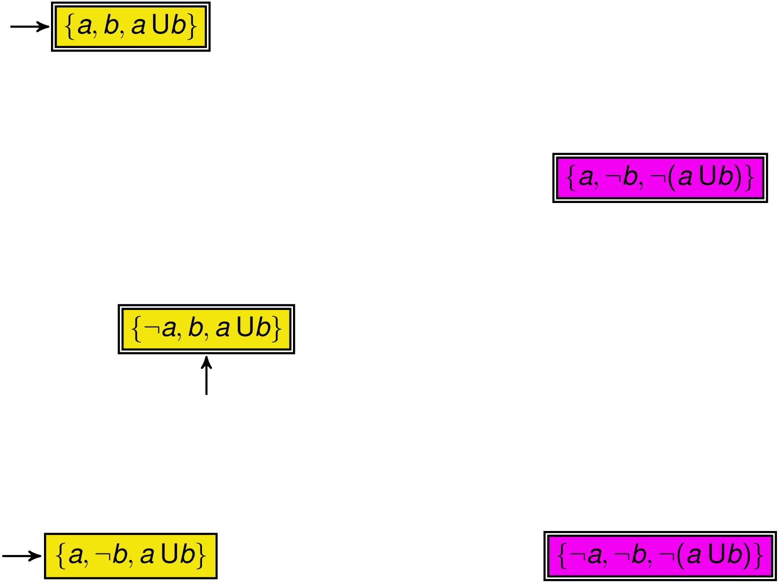
$\{a, b, a\,\mathsf{U}\,b\}$
$\{a, \neg b, \neg(a\,\mathsf{U}\,b)\}$
$\emptyset$
$\emptyset$
$\{\neg a, b, a\,\mathsf{U}\,b\}$
$\emptyset$
$\emptyset$
$\{a, \neg b, a\,\mathsf{U}\,b\}$
$\{\neg a, \neg b, \neg(a\,\mathsf{U}\,b)\}$
$\emptyset$

LTL to GNBA



$\{a, b, a \mathsf{U} b\}$
$\{a\}$
$\{a, \neg b, \neg(a \mathsf{U} b)\}$
$\{\neg a, b, a \mathsf{U} b\}$
$\{a\}$
$\{a, \neg b, a \mathsf{U} b\}$
$\{\neg a, \neg b, \neg(a \mathsf{U} b)\}$

LTL to GNBA : Accepting States



LTL to GNBA

Construct GNBA for $\neg(a \, \mathrm{U} b)$.

LTL to GNBA

- Let $\varphi = a \,\mathsf{U} (\neg a \,\mathsf{U} c)$. Let $\psi = \neg a \,\mathsf{U} c$
- Subformulae of φ : $\{a, \neg a, c, \psi, \varphi\}$. Let $B = \{a, \neg a, c, \neg c, \psi, \neg\psi, \varphi, \neg\varphi\}$.
- Possibilities at each state
  - $\{a, c, \psi, \varphi\}$
  - $\{\neg a, c, \psi, \varphi\}$
  - $\{a, \neg c, \neg\psi, \varphi\}$
  - $\{a, \neg c, \neg\psi, \neg\varphi\}$
  - $\{\neg a, \neg c, \psi, \varphi\}$
  - $\{\neg a, \neg c, \neg\psi, \neg\varphi\}$

LTL to GNBA

$\rightarrow \{a, c, \psi, \varphi\}$

$\{\neg a, \neg c, \psi, \varphi\} \leftarrow$

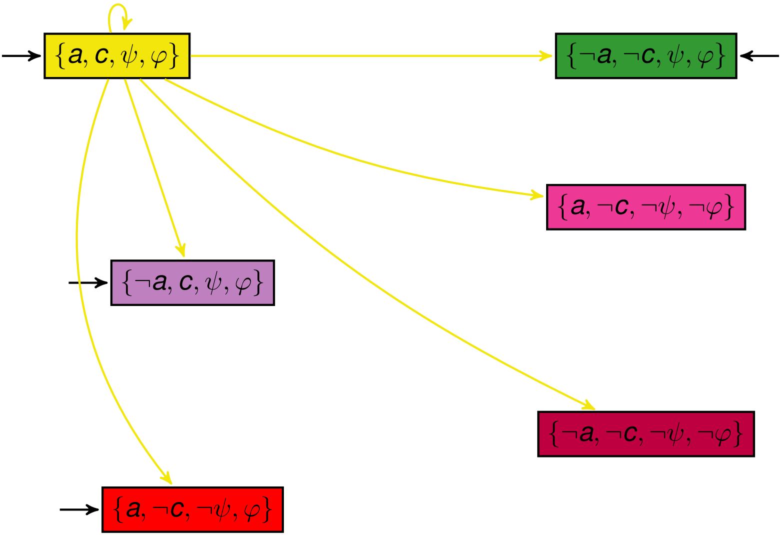
$\{a, \neg c, \neg\psi, \neg\varphi\}$

$\rightarrow \{\neg a, c, \psi, \varphi\}$

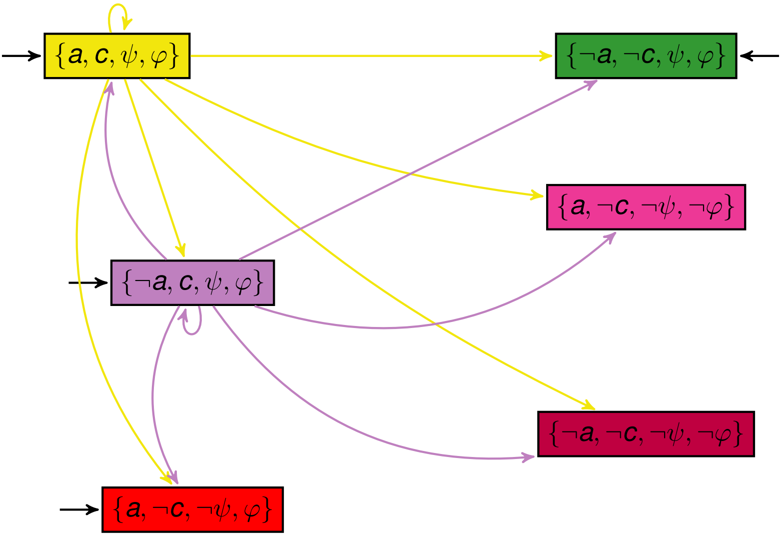
$\{\neg a, \neg c, \neg\psi, \neg\varphi\}$

$\rightarrow \{a, \neg c, \neg\psi, \varphi\}$

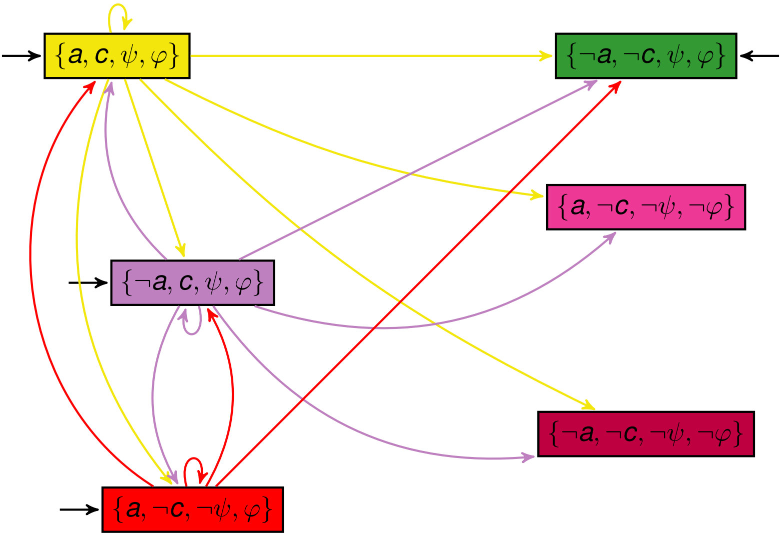
LTL to GNBA


$\{a, c, \psi, \varphi\}$
$\{\neg a, \neg c, \psi, \varphi\}$
$\{a, \neg c, \neg\psi, \neg\varphi\}$
$\{\neg a, c, \psi, \varphi\}$
$\{\neg a, \neg c, \neg\psi, \neg\varphi\}$
$\{a, \neg c, \neg\psi, \varphi\}$

LTL to GNBA

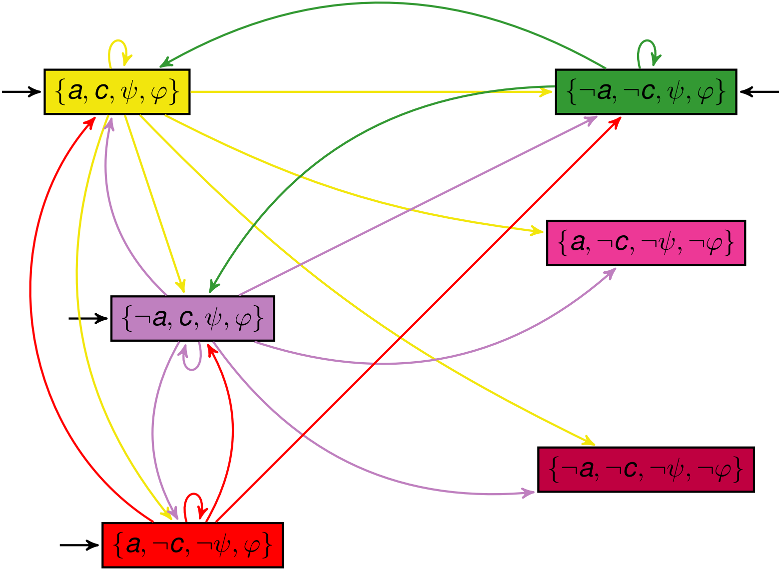

$\{a, c, \psi, \varphi\}$
$\{\neg a, \neg c, \psi, \varphi\}$
$\{a, \neg c, \neg\psi, \neg\varphi\}$
$\{\neg a, c, \psi, \varphi\}$
$\{\neg a, \neg c, \neg\psi, \neg\varphi\}$
$\{a, \neg c, \neg\psi, \varphi\}$

LTL to GNBA



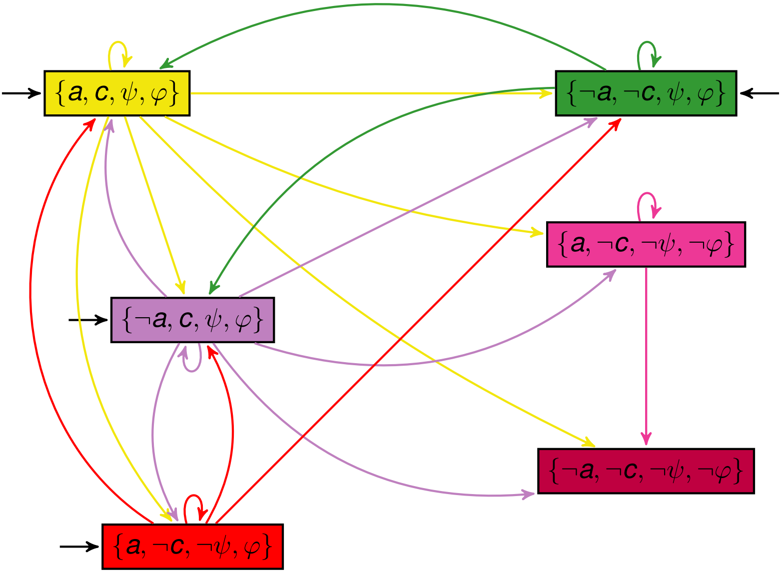
$\{a, c, \psi, \varphi\}$
$\{\neg a, \neg c, \psi, \varphi\}$
$\{a, \neg c, \neg\psi, \neg\varphi\}$
$\{\neg a, c, \psi, \varphi\}$
$\{\neg a, \neg c, \neg\psi, \neg\varphi\}$
$\{a, \neg c, \neg\psi, \varphi\}$

LTL to GNBA

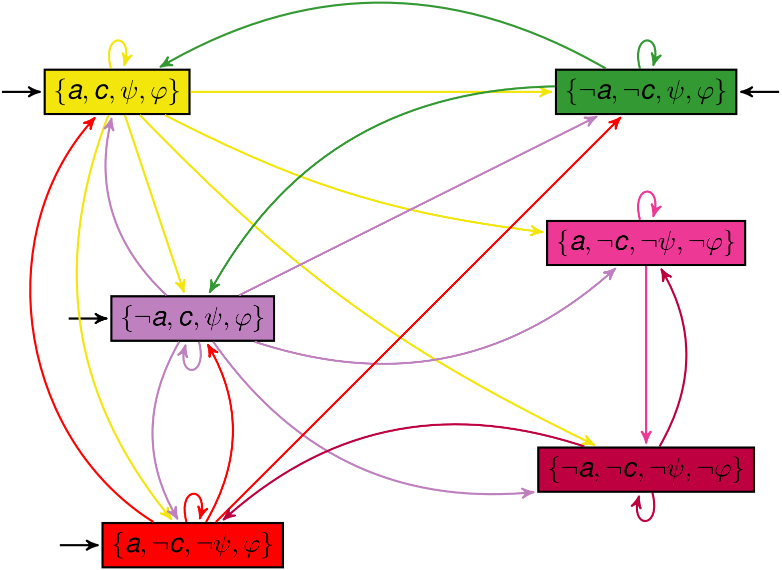


$\{a, c, \psi, \varphi\}$
$\{\neg a, \neg c, \psi, \varphi\}$
$\{a, \neg c, \neg\psi, \neg\varphi\}$
$\{\neg a, c, \psi, \varphi\}$
$\{\neg a, \neg c, \neg\psi, \neg\varphi\}$
$\{a, \neg c, \neg\psi, \varphi\}$

LTL to GNBA


$\{a, c, \psi, \varphi\}$
$\{\neg a, \neg c, \psi, \varphi\}$
$\{a, \neg c, \neg\psi, \neg\varphi\}$
$\{\neg a, c, \psi, \varphi\}$
$\{\neg a, \neg c, \neg\psi, \neg\varphi\}$
$\{a, \neg c, \neg\psi, \varphi\}$

LTL to GNBA



$\{a, c, \psi, \varphi\}$
$\{\neg a, \neg c, \psi, \varphi\}$
$\{a, \neg c, \neg\psi, \neg\varphi\}$
$\{\neg a, c, \psi, \varphi\}$
$\{\neg a, \neg c, \neg\psi, \neg\varphi\}$
$\{a, \neg c, \neg\psi, \varphi\}$

GNBA Acceptance Condition

- $\psi = \neg a \, \mathsf{U} c$
- $\varphi = a \, \mathsf{U} \psi$
- $F_1 = \{B \mid \psi \in B \rightarrow c \in B\}$
- $F_2 = \{B \mid \varphi \in B \rightarrow \psi \in B\}$
- $\mathcal{F} = \{F_1, F_2\}$

Final States

$\rightarrow$ $\{a, c, \psi, \varphi\} \in F_1, F_2$

$\{\neg a, \neg c, \psi, \varphi\} \in F_2$ $\leftarrow$

$\{a, \neg c, \neg\psi, \neg\varphi\} \in F_1, F_2$

$\rightarrow$ $\{\neg a, c, \psi, \varphi\} \in F_1, F_2$

$\{\neg a, \neg c, \neg\psi, \neg\varphi\} \in F_1, F_2$

$\rightarrow$ $\{a, \neg c, \neg\psi, \varphi\} \in F_1$

Putting Together

- Given φ, build $Cl(\varphi)$, the set of all subformulae of φ and their negations

Putting Together

- Given φ, build $Cl(\varphi)$, the set of all subformulae of φ and their negations
- Consider those $B \subseteq Cl(\varphi)$ which are consistent
  - $\varphi_1 \wedge \varphi_2 \in B \leftrightarrow \varphi_1 \in B$ and $\varphi_2 \in B$

Putting Together

- Given φ, build $Cl(\varphi)$, the set of all subformulae of φ and their negations
- Consider those $B \subseteq Cl(\varphi)$ which are consistent
  - $\varphi_1 \wedge \varphi_2 \in B \leftrightarrow \varphi_1 \in B$ and $\varphi_2 \in B$
  - $\psi \in B \rightarrow \neg\psi \notin B$ and $\psi \notin B \rightarrow \neg\psi \in B$

Putting Together

- Given φ, build $Cl(\varphi)$, the set of all subformulae of φ and their negations
- Consider those $B \subseteq Cl(\varphi)$ which are consistent
  - $\varphi_1 \wedge \varphi_2 \in B \leftrightarrow \varphi_1 \in B$ and $\varphi_2 \in B$
  - $\psi \in B \rightarrow \neg\psi \notin B$ and $\psi \notin B \rightarrow \neg\psi \in B$
  - Whenever $\psi_1 \, \mathsf{U} \psi_2 \in Cl(\varphi)$,
    - $\psi_2 \in B \rightarrow \psi_1 \, \mathsf{U} \psi_2 \in B$
    - $\psi_1 \, \mathsf{U} \psi_2 \in B$ and $\psi_2 \notin B \rightarrow \psi_1 \in B$

Putting Together

Given φ over AP, construct $A_\varphi = (Q, 2^{AP}, \delta, Q_0, \mathcal{F})$,

- $Q = \{B \mid B \subseteq Cl(\varphi) \text{ is consistent }\}$
- $Q_0 = \{B \mid \varphi \in B\}$
- $\delta : Q \times 2^{AP} \to 2^Q$ is such that

Putting Together

Given φ over AP, construct $A_\varphi = (Q, 2^{AP}, \delta, Q_0, \mathcal{F})$,

- $Q = \{B \mid B \subseteq Cl(\varphi) \text{ is consistent }\}$
- $Q_0 = \{B \mid \varphi \in B\}$
- $\delta : Q \times 2^{AP} \to 2^Q$ is such that
  - For $C = B \cap AP$, $\delta(B, C)$ is enabled and is defined as :
  - If $\bigcirc\psi \in Cl(\varphi)$, $\bigcirc\psi \in B$ iff $\psi \in \delta(B, C)$
  - If $\varphi_1 \,\mathsf{U} \varphi_2 \in Cl(\varphi)$,
    $\varphi_1 \,\mathsf{U} \varphi_2 \in B$ iff $(\varphi_2 \in B \vee (\varphi_1 \in B \wedge \varphi_1 \,\mathsf{U} \varphi_2 \in \delta(B, C)))$

Putting Together

Given φ over AP, construct $A_\varphi = (Q, 2^{AP}, \delta, Q_0, \mathcal{F})$,

- $Q = \{B \mid B \subseteq Cl(\varphi) \text{ is consistent }\}$
- $Q_0 = \{B \mid \varphi \in B\}$
- $\delta : Q \times 2^{AP} \rightarrow 2^Q$ is such that
  - For $C = B \cap AP$, $\delta(B, C)$ is enabled and is defined as :
  - If $\bigcirc\psi \in Cl(\varphi)$, $\bigcirc\psi \in B$ iff $\psi \in \delta(B, C)$
  - If $\varphi_1 \mathsf{U} \varphi_2 \in Cl(\varphi)$,
    $\varphi_1 \mathsf{U} \varphi_2 \in B$ iff $(\varphi_2 \in B \vee (\varphi_1 \in B \wedge \varphi_1 \mathsf{U} \varphi_2 \in \delta(B, C)))$
- $\mathcal{F} = \{F_{\varphi_1 \mathsf{U} \varphi_2} \mid \varphi_1 \mathsf{U} \varphi_2 \in Cl(\varphi)\}$, with
  $F_{\varphi_1 \mathsf{U} \varphi_2} = \{B \in Q \mid \varphi_1 \mathsf{U} \varphi_2 \in B \rightarrow \varphi_2 \in B\}$

Putting Together

Given φ over AP, construct $A_\varphi = (Q, 2^{AP}, \delta, Q_0, \mathcal{F})$,

- $Q = \{B \mid B \subseteq Cl(\varphi) \text{ is consistent }\}$
- $Q_0 = \{B \mid \varphi \in B\}$
- $\delta : Q \times 2^{AP} \to 2^Q$ is such that
  - For $C = B \cap AP$, $\delta(B, C)$ is enabled and is defined as :
  - If $\bigcirc\psi \in Cl(\varphi)$, $\bigcirc\psi \in B$ iff $\psi \in \delta(B, C)$
  - If $\varphi_1 \mathsf{U} \varphi_2 \in Cl(\varphi)$,
    $\varphi_1 \mathsf{U} \varphi_2 \in B$ iff $(\varphi_2 \in B \vee (\varphi_1 \in B \wedge \varphi_1 \mathsf{U} \varphi_2 \in \delta(B, C)))$
- $\mathcal{F} = \{F_{\varphi_1 \mathsf{U} \varphi_2} \mid \varphi_1 \mathsf{U} \varphi_2 \in Cl(\varphi)\}$, with
  $F_{\varphi_1 \mathsf{U} \varphi_2} = \{B \in Q \mid \varphi_1 \mathsf{U} \varphi_2 \in B \to \varphi_2 \in B\}$
- Prove that $L(\varphi) = L(A_\varphi)$

$L(\varphi) \subseteq L(A_\varphi)$

Let $\sigma = A_0 A_1 A_2 \cdots \in L(\varphi)$. Show that there is an accepting run $B_0 A_0 B_1 A_1 B_2 A_2 \ldots$ in A_φ for σ, B_i are the states, such that $B_i = \{\psi \mid A_i A_{i+1} \ldots \models \psi\}$.

Structural induction on φ

- $\varphi = a$. All starting states contain a, and can go to all successor states with all combinations of propositions.
- If $a \in B_i$, every run starting at B_i starts with a. Hence, $A_i A_{i+1} \ldots \models a$

$L(\varphi) \subseteq L(A_\varphi)$

Let $\sigma = A_0 A_1 A_2 \cdots \in L(\varphi)$. Show that there is an accepting run $B_0 A_0 B_1 A_1 B_2 A_2 \ldots$ in A_φ for σ, B_i are the states, such that $B_i = \{\psi \mid A_i A_{i+1} \ldots \models \psi\}$.

- $\varphi = \bigcirc a$, then all initial states contain $\bigcirc a$, and all successor states contain a by construction.
- If $\bigcirc a \in B_i$, then by construction, $B_{i+1} \in \delta(B_i, B_i \cap AP)$ iff $a \in B_{i+1}$, for every successor B_{i+1}.
  Then $A_{i+1} \ldots \models a$, and hence $A_i A_{i+1} \ldots \models \bigcirc a$.

$L(\varphi) \subseteq L(A_\varphi)$

Let $\sigma = A_0 A_1 A_2 \cdots \in L(\varphi)$. Show that there is an accepting run $B_0 A_0 B_1 A_1 B_2 A_2 \ldots$ in A_φ for σ, B_i are the states, such that $B_i = \{\psi \mid A_i A_{i+1} \ldots \models \psi\}$.

- If $\varphi_1 \mathrm{U} \varphi_2 \in B_i$, then by construction, either $\varphi_2 \in B_i$ or $\varphi_1, \varphi_1 \mathrm{U} \varphi_2 \in B_i$. If $\varphi_2 \in B_i$ then $A_i A_{i+1} \ldots \models \varphi_2$ by induction hypothesis, and hence, $A_i A_{i+1} \ldots \models \varphi_1 \mathrm{U} \varphi_2$.

$L(\varphi) \subseteq L(A_\varphi)$

Let $\sigma = A_0 A_1 A_2 \cdots \in L(\varphi)$. Show that there is an accepting run $B_0 A_0 B_1 A_1 B_2 A_2 \ldots$ in A_φ for σ, B_i are the states, such that $B_i = \{\psi \mid A_i A_{i+1} \ldots \models \psi\}$.

- If $\varphi_1 \mathrm{U} \varphi_2 \in B_i$, then by construction, either $\varphi_2 \in B_i$ or $\varphi_1, \varphi_1 \mathrm{U} \varphi_2 \in B_i$. If $\varphi_2 \in B_i$ then $A_i A_{i+1} \ldots \models \varphi_2$ by induction hypothesis, and hence, $A_i A_{i+1} \ldots \models \varphi_1 \mathrm{U} \varphi_2$.
- If $\varphi_1, \varphi_1 \mathrm{U} \varphi_2 \in B_i$, then by construction, $B_{i+1} \in \delta(B_i, B_i \cap AP)$ iff $\varphi_2 \in B_{i+1}$ or $\varphi_1, \varphi_1 \mathrm{U} \varphi_2 \in B_{i+1}$. How long can we go like this?

$L(\varphi) \subseteq L(A_\varphi)$

Let $\sigma = A_0 A_1 A_2 \cdots \in L(\varphi)$. Show that there is an accepting run $B_0 A_0 B_1 A_1 B_2 A_2 \ldots$ in A_φ for σ, B_i are the states, such that $B_i = \{\psi \mid A_i A_{i+1} \ldots \models \psi\}$.

- If $\varphi_1 \, \mathsf{U} \varphi_2 \in B_i$, then by construction, either $\varphi_2 \in B_i$ or $\varphi_1, \varphi_1 \, \mathsf{U} \varphi_2 \in B_i$. If $\varphi_2 \in B_i$ then $A_i A_{i+1} \ldots \models \varphi_2$ by induction hypothesis, and hence, $A_i A_{i+1} \ldots \models \varphi_1 \, \mathsf{U} \varphi_2$.
- If $\varphi_1, \varphi_1 \, \mathsf{U} \varphi_2 \in B_i$, then by construction, $B_{i+1} \in \delta(B_i, B_i \cap AP)$ iff $\varphi_2 \in B_{i+1}$ or $\varphi_1, \varphi_1 \, \mathsf{U} \varphi_2 \in B_{i+1}$. How long can we go like this?
- If $\varphi_2 \in B_k$ for $k > i$, and $\varphi_1, \varphi_1 \, \mathsf{U} \varphi_2 \in B_{i+1}, \ldots, B_{k-1}$ then $A_i A_{i+1} \ldots \models \varphi_1 \, \mathsf{U} \varphi_2$.
- When is $B_i B_{i+1} B_{i+2} \ldots$ an accepting run?
- $B_j \in F_{\varphi_1 \, \mathsf{U} \varphi_2}$ for infinitely many $j \geqslant i$.

$L(\varphi) \subseteq L(A_\varphi)$

Let $\sigma = A_0A_1A_2\cdots \in L(\varphi)$. Show that there is an accepting run $B_0A_0B_1A_1B_2A_2\ldots$ in A_φ for σ, B_i are the states, such that $B_i = \{\psi \mid A_iA_{i+1}\ldots \models \psi\}$.

- If $\varphi_1 \mathsf{U} \varphi_2 \in B_i$, then by construction, either $\varphi_2 \in B_i$ or $\varphi_1, \varphi_1 \mathsf{U} \varphi_2 \in B_i$. If $\varphi_2 \in B_i$ then $A_iA_{i+1}\ldots \models \varphi_2$ by induction hypothesis, and hence, $A_iA_{i+1}\ldots \models \varphi_1 \mathsf{U} \varphi_2$.
- If $\varphi_1, \varphi_1 \mathsf{U} \varphi_2 \in B_i$, then by construction, $B_{i+1} \in \delta(B_i, B_i \cap AP)$ iff $\varphi_2 \in B_{i+1}$ or $\varphi_1, \varphi_1 \mathsf{U} \varphi_2 \in B_{i+1}$. How long can we go like this?
- If $\varphi_2 \in B_k$ for $k > i$, and $\varphi_1, \varphi_1 \mathsf{U} \varphi_2 \in B_{i+1}, \ldots, B_{k-1}$ then $A_iA_{i+1}\ldots \models \varphi_1 \mathsf{U} \varphi_2$.
- When is $B_iB_{i+1}B_{i+2}\ldots$ an accepting run?
- $B_j \in F_{\varphi_1 \mathsf{U} \varphi_2}$ for infinitely many $j \geqslant i$.
- $\varphi_2 \notin B_j$ or $\varphi_2, \varphi_1 \mathsf{U} \varphi_2 \in B_j$ for infinitely many $j \geqslant i$.
- By construction, there is an accepting run where $\varphi_2 \in B_k$ for some $k \geqslant i$. Hence, $A_iA_{i+1}\ldots \models \varphi_1 \mathsf{U} \varphi_2$.

$L(\varphi) \subseteq L(A_\varphi)$

Let $\sigma = A_0 A_1 A_2 \cdots \in L(\varphi)$. Show that there is an accepting run $B_0 A_0 B_1 A_1 B_2 A_2 \ldots$ in A_φ for σ, B_i are the states, such that $B_i = \{\psi \mid A_i A_{i+1} \ldots \models \psi\}$.

- If $\neg(\varphi_1 \, \mathsf{U} \varphi_2) \in B_i$, then either $\neg\varphi_1, \neg\varphi_2 \in B_i$ or $\varphi_1, \neg\varphi_2 \in B_i$. If $\neg\varphi_1, \neg\varphi_2 \in B_i$ then $A_i A_{i+1} \ldots \models \neg(\varphi_1 \, \mathsf{U} \varphi_2)$.

$L(\varphi) \subseteq L(A_\varphi)$

Let $\sigma = A_0 A_1 A_2 \cdots \in L(\varphi)$. Show that there is an accepting run $B_0 A_0 B_1 A_1 B_2 A_2 \ldots$ in A_φ for σ, B_i are the states, such that $B_i = \{\psi \mid A_i A_{i+1} \ldots \models \psi\}$.

- If $\neg(\varphi_1 \mathsf{U} \varphi_2) \in B_i$, then either $\neg\varphi_1, \neg\varphi_2 \in B_i$ or $\varphi_1, \neg\varphi_2 \in B_i$. If $\neg\varphi_1, \neg\varphi_2 \in B_i$ then $A_i A_{i+1} \ldots \models \neg(\varphi_1 \mathsf{U} \varphi_2)$.
- If $\varphi_1, \neg\varphi_2 \in B_i$, then by construction, $B_{i+1} \in \delta(B_i, B_i \cap AP)$ iff $\varphi_1, \neg\varphi_2 \in B_{i+1}$ or $\neg\varphi_1, \neg\varphi_2 \in B_{i+1}$.

$L(\varphi) \subseteq L(A_\varphi)$

Let $\sigma = A_0 A_1 A_2 \cdots \in L(\varphi)$. Show that there is an accepting run $B_0 A_0 B_1 A_1 B_2 A_2 \ldots$ in A_φ for σ, B_i are the states, such that $B_i = \{\psi \mid A_i A_{i+1} \ldots \models \psi\}$.

- If $\neg(\varphi_1 \mathsf{U} \varphi_2) \in B_i$, then either $\neg\varphi_1, \neg\varphi_2 \in B_i$ or $\varphi_1, \neg\varphi_2 \in B_i$. If $\neg\varphi_1, \neg\varphi_2 \in B_i$ then $A_i A_{i+1} \ldots \models \neg(\varphi_1 \mathsf{U} \varphi_2)$.
- If $\varphi_1, \neg\varphi_2 \in B_i$, then by construction, $B_{i+1} \in \delta(B_i, B_i \cap AP)$ iff $\varphi_1, \neg\varphi_2 \in B_{i+1}$ or $\neg\varphi_1, \neg\varphi_2 \in B_{i+1}$.
- Either case, $A_i A_{i+1} \ldots \models \neg(\varphi_1 \mathsf{U} \varphi_2)$

$L(A_\varphi) \subseteq L(\varphi)$

For a sequence $B_0 B_1 B_2 \ldots$ of states satisfying

- $B_{i+1} \in \delta(B_i, A_i)$,
- $\forall F \in \mathcal{F}$, $B_j \in F$ for infinitely many j,

we have $\psi \in B_0 \leftrightarrow A_0 A_1 \ldots \models \psi$

- Structural Induction on ψ. Interesting case : $\psi = \varphi_1 \,\mathsf{U} \varphi_2$
- Assume $A_0 A_1 \ldots \models \varphi_1 \,\mathsf{U} \varphi_2$. Then $\exists j \geqslant 0$, $A_j A_{j+1} \ldots \models \varphi_2$ and $A_i A_{i+1} \ldots \models \varphi_1, \varphi_1 \,\mathsf{U} \varphi_2$ for all $i \leqslant j$.
- By induction hypothesis (applied to φ_1, φ_2), we obtain $\varphi_2 \in B_j$ and $\varphi_1 \in B_i$ for all $i \leqslant j$
- By induction on j, $\varphi_1 \,\mathsf{U} \varphi_2 \in B_j, \ldots, B_0$.

$L(A_\varphi) \subseteq L(\varphi)$

For a sequence $B_0 B_1 B_2 \ldots$ of states satisfying

(a) $B_{i+1} \in \delta(B_i, A_i)$,

(b) $\forall F \in \mathcal{F}$, $B_j \in F$ for infinitely many j,

we have $\psi \in B_0 \leftrightarrow A_0 A_1 \ldots \models \psi$

- Conversely, assume $\varphi_1 \,\mathsf{U} \varphi_2 \in B_0$. Then $\varphi_2 \in B_0$ or $\varphi_1, \varphi_1 \,\mathsf{U} \varphi_2 \in B_0$.
- If $\varphi_2 \in B_0$, by induction hypothesis, $A_0 A_1 \ldots \models \varphi_2$, and hence $A_0 A_1 \ldots \models \varphi_1 \,\mathsf{U} \varphi_2$
- If $\varphi_1, \varphi_1 \,\mathsf{U} \varphi_2 \in B_0$. Assume $\varphi_2 \notin B_j$ for all $j \geqslant 0$. Then $\varphi_1, \varphi_1 \,\mathsf{U} \varphi_2 \in B_j$ for all $j \geqslant 0$.
- Since $B_0 B_1 \ldots$ satisfies (b), $B_j \in F_{\varphi_1 \,\mathsf{U} \varphi_2}$ for infinitely many $j \geqslant 0$, we obtain a contradiction.
- Thus, $\exists$ a smallest k s.t. $\varphi_2 \in B_k$. Then by induction hypothesis, $A_i A_{i+1} \ldots \models \varphi_1$ and $A_k A_{k+1} \models \varphi_2$ for all $i < k$
- Hence, $A_0 A_1 \ldots \models \varphi_1 \,\mathsf{U} \varphi_2$.

GNBA Size

- States of A_φ are subsets of $Cl(\varphi)$

GNBA Size

- States of A_φ are subsets of $Cl(\varphi)$
- Maximum number of states $\leqslant 2^{|\varphi|}$

GNBA Size

- States of A_φ are subsets of *Cl*(φ)
- Maximum number of states $\leqslant 2^{|\varphi|}$
- Number of sets in $\mathcal{F} = |\varphi|$

GNBA Size

- States of A_φ are subsets of $Cl(\varphi)$
- Maximum number of states $\leqslant 2^{|\varphi|}$
- Number of sets in $\mathcal{F} = |\varphi|$
- LTL $\varphi \rightsquigarrow$ NBA A_φ : Number of states in $A_\varphi \leqslant |\varphi|.2^{|\varphi|}$
- There is no LTL formula φ for the language

$$L = \{A_0 A_1 A_2 \cdots \mid a \in A_{2i}, i \geqslant 0\}$$

Complexity of LTL Modelchecking

- Given φ, $A_{\neg\varphi}$ has $\leqslant 2^{|\varphi|}$ states
  - $|\varphi|$=size/length of φ, the number of operators in φ
- $TS \otimes A_{\neg\varphi}$ has $\leqslant |TS|.2^{|\varphi|}$ states
- Persistence checking : Checking $\Box\Diamond\eta$ on $TS \otimes A_{\neg\varphi}$ takes time linear in $\eta.|TS \otimes A_{\neg\varphi}|$

∃∀ Automata and the LTL connection

- For finite words