

Introduction

Modelling parallel systems

Linear Time Properties

Regular Properties

regular safety properties

-regular properties

model checking with Buchi automata

Linear Temporal Logic

Computation-Tree Logic

Equivalences and Abstraction

Go

idea: define **regular LT properties** to be those languages of **finite words** over the alphabet Σ^* that have a representation by a **finite automata**

idea: define **regular LT properties** to be those languages of **finite words** over the alphabet 2^{AP} that have a representation by a **finite automata**

regular safety properties:

NFA-representation for the **bad prefixes**

idea: define **regular LT properties** to be those languages of **finite words** over the alphabet 2^{AP} that have a representation by a **finite automata**

regular safety properties:

NFA-representation for the **bad prefixes**

representation other regular LT properties by

-automata, i.e., acceptors for **finite words**

idea: define **regular LT properties** to be those languages of **finite words** over the alphabet 2^{AP} that have a representation by a **finite automata**

regular safety properties:

NFA-representation for the **bad prefixes**

representation other regular LT properties by

-automata, i.e., acceptors for **finite words**

-regular expressions

remind: syntax and semantics of regular expressions
over some alphabet $= \{A, B, \dots\}$

$::=$ A $1 + 2$ $1 \cdot 2$

$::=$ A $1 + 2$ $1 \cdot 2$

where A

$::=$	A	$1 + 2$	$1 \cdot 2$
-------	-----	---------	-------------

where A

semantics: $L()$ language of the words

$$::= \quad A \quad 1 + 2 \quad 1 \cdot 2$$

where A

semantics: $L()$ language of the words

$L() =$	$L() = \{ \}$	$L(A) = \{A\}$
$L(1 + 2) =$	$L(1) \cup L(2)$	union
$L(1 \cdot 2) =$	$L(1) L(2)$	concatenation
$L()^*$	$L()^*$	Kleene closure

regular expressions:

$::= \quad / \quad / A \quad / \quad 1 + 2 \quad / \quad 1 \cdot 2 \quad /$

-regular expressions:

regular expressions + -operator

regular expressions:

$::= \quad / \quad / A \quad / \quad 1 + 2 \quad / \quad 1 \cdot 2 \quad /$

-regular expressions:

regular expressions + -operator

Kleene star: ~~finite repetition~~ $\%$

-operator: ~~finite repetition~~ $\%$

regular expressions:

$::= \quad / \quad / A \quad / \quad 1^+ \quad 2 \quad / \quad 1 \cdot 2 \quad /$

-regular expressions:

regular expressions + -operator

Kleene star: ~~finite repetition~~

-operator: ~~finite repetition~~

for L :

$L \stackrel{\text{def}}{=} w_1 w_2 w_3 \dots : w_i \in L \text{ for all } i \geq 1$

regular expressions:

$::= \quad / \quad / A \quad / \quad 1 + 2 \quad / \quad 1 \cdot 2 \quad /$

-regular expressions:

regular expressions + -operator

Kleene star: ~~finite repetition~~ $\%$

-operator: ~~finite repetition~~ $\%$

for L :

$L \stackrel{\text{def}}{=} w_1 w_2 w_3 \dots : w_i \in L \text{ for all } i \geq 1$

note: L if $/ L$

syntax of ϵ -regular expressions over alphabet Σ :

$$= \epsilon \cdot r_1 + \dots + \epsilon \cdot r_n \text{ where}$$

r_i, r_i are regular expressions over Σ s.t. $L(r_i)$

syntax of ϵ -regular expressions over alphabet Σ :

$$= \epsilon \cdot e_1 + \dots + \epsilon \cdot e_n \text{ where}$$

e_i, e_i are regular expressions over Σ s.t. $L(e_i)$

semantics: the language generated by e is:

$$L(e) \stackrel{\text{def}}{=} \bigcup_{i=1}^n L(e_i)$$

syntax of ϵ -regular expressions over alphabet Σ :

$$= \epsilon \cdot e_1 + \dots + \epsilon \cdot e_n \text{ where}$$

e_i, e_i are regular expressions over Σ s.t. $L(e_i)$

semantics: the language generated by e is:

$$L(e) \stackrel{\text{def}}{=} \bigcup_{i=1}^n L(e_i)$$

syntax of ϵ -regular expressions over alphabet Σ :

$$= \epsilon \mid \epsilon_1 + \dots + \epsilon_n \text{ where}$$

ϵ_i, ϵ_j are regular expressions over Σ s.t. $L(\epsilon_i)$

semantics: the language generated by ϵ is:

$$L(\epsilon) \stackrel{\text{def}}{=} \bigcup_{i=1}^n L(\epsilon_i)$$

language of $(A \cdot B)$

syntax of \cdot -regular expressions over alphabet Σ :

$$= \epsilon + r_1 \cdot r_2 + \dots + r_n \cdot r_{n+1} \text{ where}$$

r_i, r_{i+1} are regular expressions over Σ s.t. $L(r_i) \neq \emptyset$

semantics: the language generated by r is:

$$L(r) \stackrel{\text{def}}{=} \begin{cases} \{\epsilon\} & \text{if } r = \epsilon \\ L(r_1)L(r_2) & \text{if } r = r_1 \cdot r_2 \end{cases}$$

language of $(A \cdot B)$ = set of all infinite words over Σ
 = $\{A, B\}$ containing infinitely many B

syntax of ϵ -regular expressions over alphabet Σ :

$$= \epsilon + r_1 \cdot r_2 + \dots + r_n \cdot r_{n+1} \text{ where}$$

r_i, r_{i+1} are regular expressions over Σ s.t. $L(r_i) \cap L(r_{i+1}) = \emptyset$

semantics: the language generated by r is:

$$L(r) \stackrel{\text{def}}{=} \bigcup_{i=1}^n L(r_i) \cdot L(r_{i+1})$$

language of $(A \cdot B)$ = set of all infinite words over Σ containing infinitely many B

language of $(A \cdot B) + (B \cdot A)$

syntax of ϵ -regular expressions over alphabet Σ :

$$= \epsilon \cdot \epsilon_1 + \dots + \epsilon_n \cdot \epsilon_n \text{ where}$$

ϵ_i, ϵ_i are regular expressions over Σ s.t. $L(\epsilon_i)$

semantics: the language generated by ϵ is:

$$L(\epsilon) \stackrel{\text{def}}{=} \bigcup_{i=1}^n L(\epsilon_i) L(\epsilon_i)$$

language of $(A \cdot B)$ = set of all infinite words over Σ containing infinitely many B

language of $(A \cdot B) + (B \cdot A)$ = set of all infinite words over Σ with infinitely many A or B

syntax of ϵ -regular expressions over alphabet Σ :

$$= \epsilon \mid \epsilon_1 + \dots + \epsilon_n \text{ where}$$

ϵ_i, ϵ_j are regular expressions over Σ s.t. $L(\epsilon_i)$

semantics: the language generated by ϵ is:

$$L(\epsilon) \stackrel{\text{def}}{=} \bigcup_{i=1}^n L(\epsilon_i)$$

language of $(A \cdot B)$ = set of all infinite words over Σ containing infinitely many B

language of $(A \cdot B) + (B \cdot A)$ = set of all infinite words over Σ with infinitely many A or B

syntax of ϵ -regular expressions over alphabet Σ :

$$= \epsilon \cdot \epsilon_1 + \dots + \epsilon_n \cdot \epsilon_n \quad \text{where}$$

ϵ_i, ϵ_i are regular expressions over Σ s.t. $L(\epsilon_i)$

semantics: the language generated by ϵ is:

$$L(\epsilon) \stackrel{\text{def}}{=} \bigcup_{i=1}^n L(\epsilon_i) L(\epsilon_i)$$

A language L is called ϵ -regular i
there exists an ϵ -regular expression s.t.

$$L = L(\epsilon)$$

Provide an ϵ -regular expression for ...

It1mc3.2-25a

alphabet = $\{A, B\}$

set of all infinite words over alphabet containing only
finitely many A 's

Provide an ϵ -regular expression for ...

It1mc3.2-25a

alphabet = $\{A, B\}$

set of all infinite words over $\{A, B\}$ containing only
finitely many A 's

$(A + B)^* . B$

Provide an ϵ -regular expression for ...

It1mc3.2-25a

alphabet = $\{A, B\}$

set of all infinite words over containing only
finitely many A 's

$(A + B)^* . B$

set of all infinite words where each A is followed
immediately by letter B

Provide an ϵ -regular expression for ...

It1mc3.2-25a

alphabet = $\{A, B\}$

~~1/2~~ set of all infinite words over $\{A, B\}$ containing only
~~1/2~~ finitely many A 's

$$(A + B)^* . B$$

~~1/2~~ set of all infinite words where each A is followed
immediately by letter B

$$(B . A . B)^* . B + (B . A . B)^*$$

Provide an ϵ -regular expression for ...

It1mc3.2-25a

alphabet = $\{A, B\}$

~~1/0~~ set of all infinite words over $\{A, B\}$ containing only finitely many A 's

$$(A + B)^* . B$$

~~1/0~~ set of all infinite words where each A is followed immediately by letter B

$$(B . A . B)^* . B + (B . A . B)^*$$

~~1/0~~ set of all infinite words where each A is followed eventually by letter B

Provide an ϵ -regular expression for ...

It1mc3.2-25a

alphabet = $\{A, B\}$

~~1/5~~ set of all infinite words over alphabet containing only
~~1/5~~ finitely many A 's

$$(A + B)^{\omega} . B$$

~~1/5~~ set of all infinite words where each A is followed
immediately by letter B

$$(B . A . B)^{\omega} . B + (B . A . B)^{\omega}$$

~~1/5~~ set of all infinite words where each A is followed
eventually by letter B

$$(B . A^{+} . B)^{\omega} . B + (B . A^{+} . B)^{\omega}$$

where $+ \stackrel{\text{def}}{=} . .$

Provide an ϵ -regular expression for ...

It1mc3.2-25a

alphabet = $\{A, B\}$

set of all infinite words over alphabet containing only finitely many A 's

$$(A + B)^{\omega} . B$$

set of all infinite words where each A is followed immediately by letter B

$$(B . A . B)^{\omega} . B + (B . A . B)^{\omega}$$

set of all infinite words where each A is followed eventually by letter B

$$(B . A^{+} . B)^{\omega} . B + (B . A^{+} . B)^{\omega} (A . B)^{\omega}$$

where $+ \stackrel{\text{def}}{=} .$

Let E be an LT-property over AP , i.e., $E \subseteq 2^{AP}$

Let E be an LT-property over AP , i.e., $E \subseteq 2^{AP}$

E is called an **-regular property** if there exists an **-regular expression** over 2^{AP} s.t. $E = L()$

Let E be an LT-property over AP , i.e., $E \subseteq 2^{AP}$

E is called an **-regular property** if there exists an **-regular expression** over 2^{AP} s.t. $E = L(\)$

Examples for $AP = \{a, b\}$

~~Not~~invariant with invariant condition $a \neq b$

Let E be an LT-property over AP , i.e., $E \subseteq 2^{AP}$

E is called an **-regular property** if there exists an **-regular expression** over 2^{AP} s.t. $E = L(\)$

Examples for $AP = \{a, b\}$

~~Not~~invariant with invariant condition $a \neq b$

$(\{a\} + \{a, b\})$

Let E be an LT-property over AP , i.e., $E \subseteq 2^{AP}$

E is called an **-regular property** if there exists an **-regular expression** over 2^{AP} s.t. $E = L(\)$

Examples for $AP = \{a, b\}$

~~Not~~invariant with invariant condition ~~$a \neq b$~~

$(\) + \{a\} + \{a, b\}$

Each invariant is **-regular**

Let E be an LT-property over AP , i.e., $E \subseteq 2^{AP}$

E is called an **-regular property** if there exists an **-regular expression** over 2^{AP} s.t. $E = L(\)$

Examples for $AP = \{a, b\}$

~~Not~~invariant with invariant condition $a \neq b$

$(\ + \{a\} + \{a, b\})$

Each invariant is **-regular**

Let ϕ be an invariant condition and let

$\{A \subseteq AP : A \models \phi\} = \{A_1, \dots, A_k\}$

Then: invariant ~~ways~~ $\phi \models (A_1 + \dots + A_k)$

Let E be an LT-property over AP , i.e., $E \subseteq 2^{AP}$.

E is called an **-regular property** if there exists an ω -regular expression over 2^{AP} s.t. $E = L(\omega)$

Examples for $AP = \{a, b\}$:

~~Always~~ a (or any other invariant)

~~Infinitely often~~ a

~~Eventually~~ a

Let E be an LT-property over AP , i.e., $E \subseteq 2^{AP}$.

E is called an **-regular property** if there exists an **-regular expression** over 2^{AP} s.t. $E = L(\)$

Examples for $AP = \{a, b\}$:

~~Never~~ always a (or any other invariant)

~~Never~~ infinitely often a

~~Never~~ eventually a

$$(2^{AP}) . (\{a\} + \{a, b\}) . (2^{AP})$$

$$\text{where } 2^{AP} = \{ \} + \{a\} + \{b\} + \{a, b\}$$

Let E be an LT-property over AP , i.e., $E \subseteq 2^{AP}$.

E is called an **-regular property** if there exists an **-regular expression** over 2^{AP} s.t. $E = L(\)$

Examples for $AP = \{a, b\}$:

~~Never~~ always a (or any other invariant)

~~Never~~ infinitely often a

~~Never~~ eventually a

$(2^{AP}) \cdot (\{a\} + \{a, b\}) \cdot (2^{AP})$

~~Never~~ from some moment on a

Let E be an LT-property over AP , i.e., $E \subseteq 2^{AP}$.

E is called an **-regular property** if there exists an **-regular expression** over 2^{AP} s.t. $E = L(\)$

Examples for $AP = \{a, b\}$:

~~Never~~ always a (or any other invariant)

~~Never~~ infinitely often a

~~Never~~ eventually a

$(2^{AP}) . (\{a\} + \{a, b\}) . (2^{AP})$

~~Never~~ from some moment on a

$(2^{AP}) . (\{a\} + \{a, b\})$

symbolic notation for ω -regular properties

... using **formulas** instead of **sums**

Examples for $AP = \{a, b\}$

~~%~~invariant with invariant condition ~~a~~ ~~$i \leq 0$~~

(+ $\{a\}$ + $\{a, b\}$)

Examples for $AP = \{a, b\}$

\neg invariant with invariant condition $a \wedge b$

$$(a \wedge b) = (\text{ } + \{a\} + \{a, b\})$$

~~Not~~invariant with invariant condition **a** ~~is~~ **no**

~~Very~~ often ~~a~~

47 / 233

~~Not~~invariant with invariant condition **a** ~~is~~ **no**

~~Very~~ often ~~a~~

~~From~~ from some moment on ~~a~~

Examples for $AP = \{a, b\}$

$\mathcal{L} \models$ invariant with invariant condition $a \wedge \neg b$

$$(a \wedge \neg b) = (\text{true} + \{a\} + \{\neg b\})$$

$\mathcal{L} \models$ eventually often a

$$(\neg a) \cdot a = (\text{true} + \{\neg a\}) \cdot (\{a\} + \{a, b\})$$

$\mathcal{L} \models$ from some moment on a

$$\text{true} \cdot a$$

Examples for $AP = \{a, b\}$

φ invariant with invariant condition $a \wedge b$

$$(\varphi \wedge a \wedge b) = (\varphi \wedge (a + \{a\} + \{a, b\}))$$

φ infinitely often a

$$(\varphi \wedge a) \wedge b = (\varphi \wedge (a + \{b\})) \wedge (\{a\} + \{a, b\})$$

φ from some moment on a

$$\text{true} \wedge a$$

φ whenever a then b will hold somewhen later

Examples for $AP = \{a, b\}$

$\mathcal{L} \models \text{invariant with invariant condition } a \text{ } i_{\mathcal{L}}^a$

$$(a \text{ } i_{\mathcal{L}}^a) = (\text{ } + \{a\} + \{a, b\})$$

$\mathcal{L} \models \text{Eventually often } a \text{ } e_{\mathcal{L}}^a$

$$(e_{\mathcal{L}}^a) . a = (\text{ } + \{b\}) . (\{a\} + \{a, b\})$$

$\mathcal{L} \models \text{From some moment on } a \text{ } f_{\mathcal{L}}^a$

$$\text{true} . a$$

$\mathcal{L} \models \text{Whenever } a \text{ then } b \text{ will hold somewhen later } \mathcal{L} \models$

$$(f_{\mathcal{L}}^a) . a . \text{true} . b . (f_{\mathcal{L}}^a) + (f_{\mathcal{L}}^a) . a . \text{true} . b$$

Nondeterministic Buchi automata (NBA)

ltlmc3.2-21a

syntax as for **NFA**

nondeterministic Buchi automata

Nondeterministic Buchi automata (NBA)

lt1mc3.2-21a

syntax as for **NFA**

nondeterministic finite automata

semantics: language of infinite words

Nondeterministic Buchi automata (NBA)

lt1mc3.2-21a

NBA $A = (Q, \delta, Q_0, F)$

NBA $A = (Q, \delta, Q_0, F)$

Q is the set of states

NBA $A = (Q, \delta, Q_0, F)$

Q finite set of states

Σ alphabet

NBA $A = (Q, \Sigma, Q_0, F)$

Q finite set of states

Σ alphabet

$\delta : Q \times \Sigma \rightarrow 2^Q$ transition relation

NBA $A = (Q, \Sigma, Q_0, F)$

Q finite set of states

Σ alphabet

$\delta : Q \times \Sigma \rightarrow 2^Q$ transition relation

$Q_0 \subseteq Q$ set of initial states

NBA $A = (Q, \Sigma, Q_0, F)$

Q finite set of states

Σ alphabet

$\delta : Q \times \Sigma \rightarrow 2^Q$ transition relation

$Q_0 \subseteq Q$ set of initial states

$F \subseteq Q$ set of ~~bad~~ states, also called accept states

NBA $A = (Q, \Sigma, Q_0, F)$

Q finite set of states

Σ alphabet

$\delta : Q \times \Sigma \rightarrow 2^Q$ transition relation

$Q_0 \subseteq Q$ set of initial states

$F \subseteq Q$ set of ~~bad~~ states, also called **accept states**

run for a word $A_0 A_1 A_2 \dots$:

state sequence $= q_0 q_1 q_2 \dots$ where $q_0 \in Q_0$

and $q_{i+1} \in \delta(q_i, A_i)$ for $i \geq 0$

Nondeterministic Buchi automata (NBA)

lt1mc3.2-21a

NBA $A = (Q, \Sigma, Q_0, F)$

Q finite set of states

Σ alphabet

$\delta : Q \times \Sigma \rightarrow 2^Q$ transition relation

$Q_0 \subseteq Q$ set of initial states

$F \subseteq Q$ set of ~~final~~ states, also called **accept states**

run for a word $A_0 A_1 A_2 \dots$:

state sequence $= q_0 q_1 q_2 \dots$ where $q_0 \in Q_0$

and $q_{i+1} \in \delta(q_i, A_i)$ for $i \geq 0$

run is **accepting** if $\exists i \in \mathbb{N}. q_i \in F$

NBA $A = (Q, \Sigma, Q_0, F)$

Q finite set of states

Σ alphabet

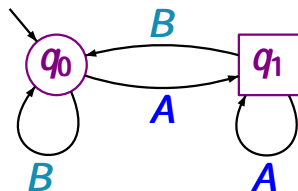
$\delta : Q \times \Sigma \rightarrow 2^Q$ transition relation

$Q_0 \subseteq Q$ set of initial states

$F \subseteq Q$ set of ~~final~~ states, also called **accept states**

accepted language $L(A)$ is given by:

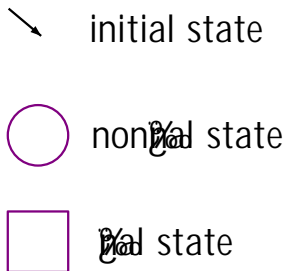
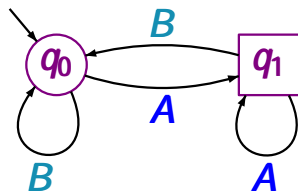
$L(A) \stackrel{\text{def}}{=} \{ w \mid w \text{ is a word over } \Sigma \text{ that has an accepting run in } A \}$



↘ initial state

○ non-final state

□ final state

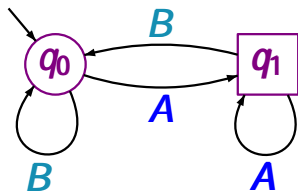


NBA with state space $\{q_0, q_1\}$

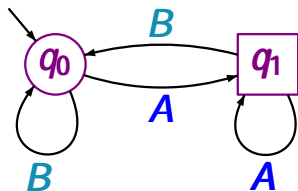
q_0 initial state

q_1 accept state

alphabet = $\{A, B\}$

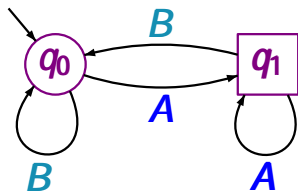


accepted language: ?



accepted language:

set of all infinite words that
contain infinitely many A 's

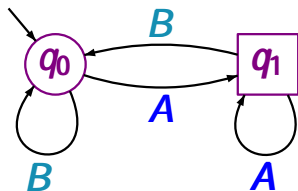


accepted language:

set of all infinite words that
contain infinitely many A 's
($B \cdot A$)

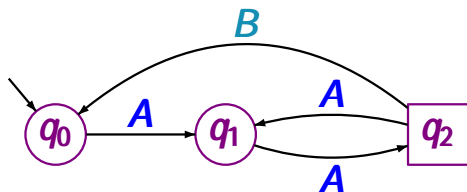
Examples for NBA over $\Sigma = \{A, B\}$

ltlmc3.2-22



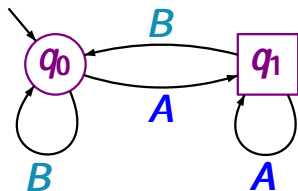
accepted language:

set of all infinite words that
contain infinitely many A 's
($B.A$)^{*}



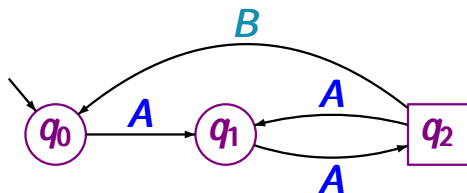
Examples for NBA over $\Sigma = \{A, B\}$

lt1mc3.2-22



accepted language:

set of all infinite words that
contain infinitely many A 's
($B \cdot A$)

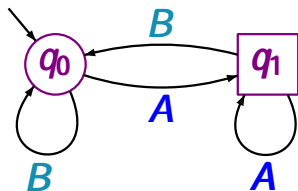


$AABABABAB \dots$
 $AAAAAAAAAA \dots$

accepted words

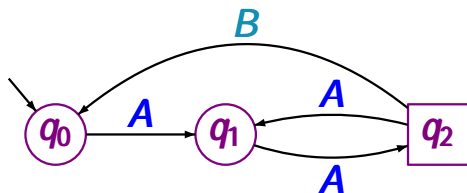
Examples for NBA over $\Sigma = \{A, B\}$

lt1mc3.2-22



accepted language:

set of all infinite words that
contain infinitely many A 's
($B \cdot A$)

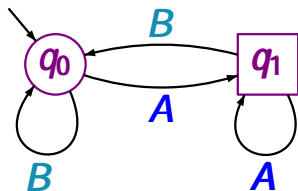


accepted language:

Every B is preceded
by a positive even
number of A 's

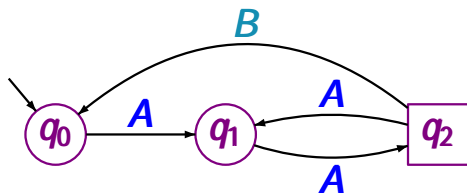
$AABAAABAAB \dots$
 $AAAAAAAAAA \dots$

accepted words



accepted language:

set of all infinite words that
contain infinitely many A 's
($B.A$)^{*}



accepted language:

Every B is preceded
by a positive even
number of A 's

$$((A.A)^+.B) + ((A.A)^+.B).A$$

NBA $A = (Q, \Sigma, Q_0, F)$

Q finite set of states

Σ alphabet

$\delta : Q \times \Sigma \rightarrow 2^Q$ transition relation

Q_0 Q set of initial states

F Q set of ~~final~~ states, also called accept states

NBA $A = (Q, \Sigma, Q_0, F)$

Q finite set of states

Σ alphabet here: $\Sigma = 2^{AP}$

$\delta : Q \times \Sigma \rightarrow 2^Q$ transition relation

Q_0 Q set of initial states

F Q set of final states, also called accept states

NBA $A = (Q, \Sigma, Q_0, F)$

Q finite set of states

Σ alphabet here: $\Sigma = 2^{AP}$

$\delta : Q \times \Sigma \rightarrow 2^Q$ transition relation

Q_0 Q set of initial states

F Q set of final states, also called accept states

accepted language $L(A)$ is an LT-property:

$L(A) =$ set of infinite words over 2^{AP} that have an accepting run in A



$$L(A) = ?$$

set of atomic propositions $AP = \{a, b\}$

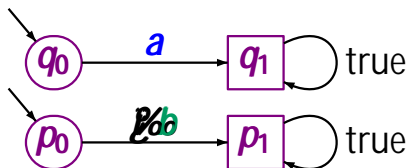


$$L(A) = \text{true. } \cancel{P/a}. \text{true}$$

set of atomic propositions $AP = \{a, b\}$



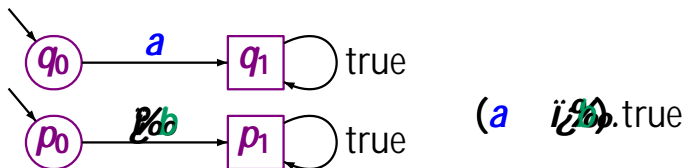
$$L(A) = \text{true. } \cancel{P/a}. \text{true}$$



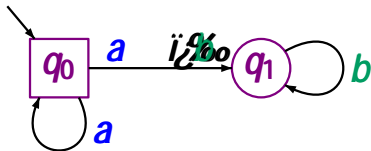
set of atomic propositions $AP = \{a, b\}$



$$L(A) = \text{true. } a. \text{true}$$

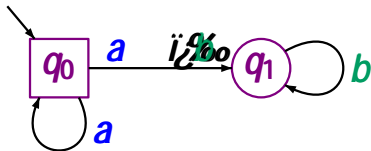


set of atomic propositions $AP = \{a, b\}$

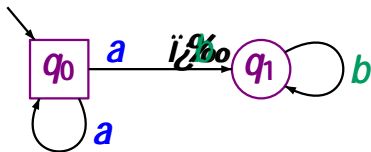


NBA for LT properties

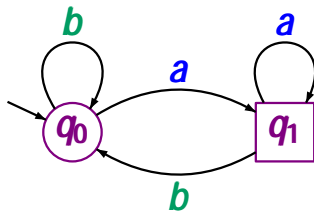
lt1mc3.2-NBA-2-omega-reg

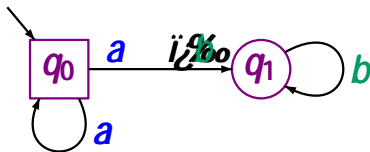


Always $a \neq a$

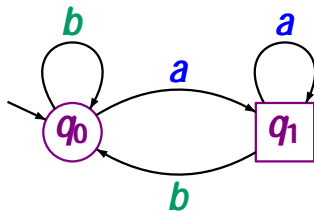


Always $a \neq a$

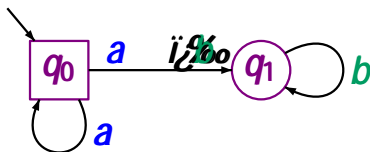




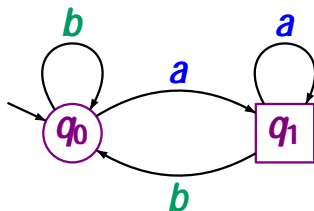
always a ω



infinitely often a and ... ω



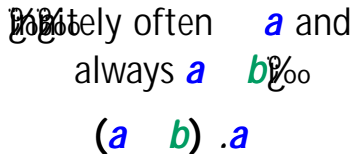
Always $a \vee b$



Exactly often a and always $a \wedge b$

$$= (a \wedge b) \cdot a$$

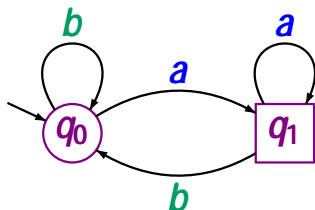
ltlmc3.2-NBA-2-omega-reg



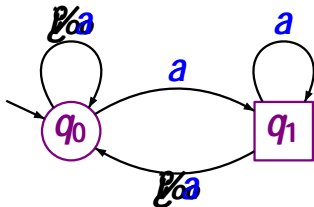
86 / 233

NBA for LT properties

lt1mc3.2-NBA-2-omega-reg



Does not satisfy often a and
 always a b^{ω}
 $(a \ b)^{\omega} . a$



Does not satisfy often a^{ω}
 $(a^{\omega}) . a$

From NBA to ω -regular expressions

ltlmc3.2-NBA-to-omega

For each NBA A there is an ω -regular expression
with $L(A) = L(\omega)$

For each NBA A there is an ω -regular expression
with $L(A) = L(\)$

Proof. Let A be an NBA (Q, δ, Q_0, F)

For each NBA A there is an ω -regular expression
with $L(A) = L(R)$

Proof. Let A be an NBA (Q, δ, Q_0, F) and $q, p \in Q$.
Let $A_{q,p}$ be the NFA $(Q, \delta, q, \{p\})$.

For each NBA A there is an ω -regular expression with $L(A) = L(\text{expression})$

Proof. Let A be an NBA (Q, Σ, Q_0, F) and $q, p \in Q$. Let $A_{q,p}$ be the NFA $(Q, \Sigma, q, \{p\})$. Then:

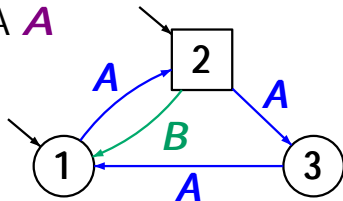
$$L(A) = \bigcup_{q \in Q_0} \bigcap_{p \in F} L(A_{q,p}) \setminus \{\epsilon\}$$

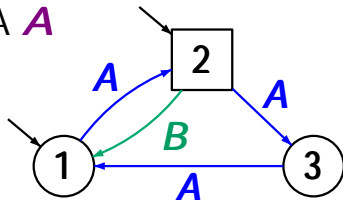
For each NBA A there is an ω -regular expression with $L(A) = L(\text{expression})$

Proof. Let A be an NBA (Q, Σ, Q_0, F) and $q, p \in Q$. Let $A_{q,p}$ be the NFA $(Q, \Sigma, q, \{p\})$. Then:

$$L(A) = \bigcup_{q \in Q_0} \bigcap_{p \in F} L(A_{q,p}) \setminus \{\epsilon\}$$

is ω -regular as $L(A_{q,p})$ and $L(A_{p,p}) \setminus \{\epsilon\}$ are regular

NBA *A*

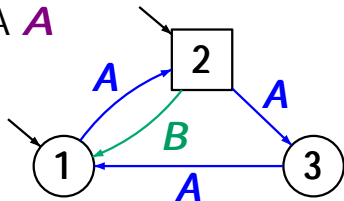
NBA A 

$$L(A) = L_{12}(L_{22}) \quad L_{22}(L_{22})$$

$$L_{12} = L(A_{12})$$

$$L_{22} = L(A_{22})$$

$$L_{22} = L_{22} \setminus \{ \}$$

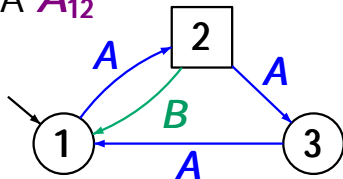
NBA A 

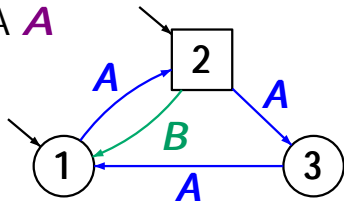
$$L(A) = L_{12}(L_{22}) \quad L_{22}(L_{22})$$

$$L_{12} = L(A_{12})$$

$$L_{22} = L(A_{22})$$

$$L_{22} = L_{22} \setminus \{ \}$$

NFA A_{12} 

NBA A 

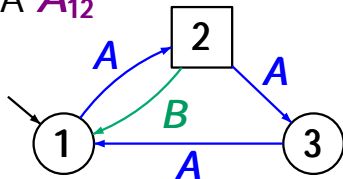
$$L(A) = L_{12}(L_{22}) \quad L_{22}(L_{22})$$

$$L_{12} = L(A_{12})$$

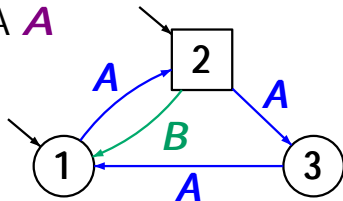
$$L_{22} = L(A_{22})$$

$$L_{22} = L_{22} \setminus \{ \}$$

$$L_{12} = A.(B.A + A.A.A)$$

NFA A_{12} 

NBA A



$$L(A) = L_{12}(L_{22}) \quad L_{22}(L_{22})$$

$$L_{12} = L(A_{12})$$

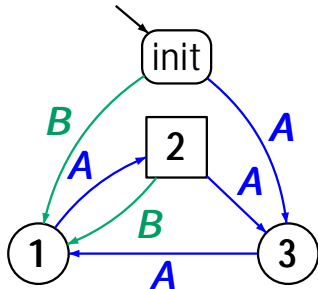
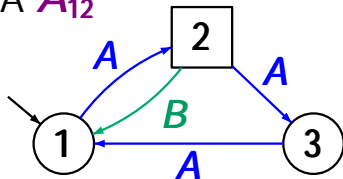
$$L_{22} = L(A_{22})$$

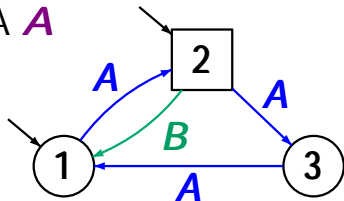
$$L_{22} = L_{22} \setminus \{ \}$$

$$L_{12} = A.(B.A + A.A.A)$$

$$L_{22} = (B.A + A.A.A)^+$$

NFA A_{12}

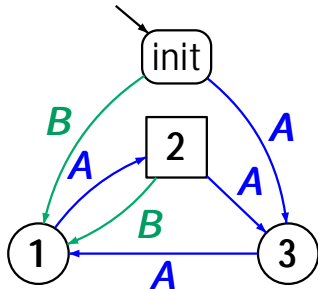
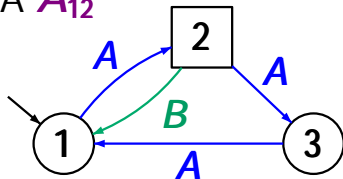


NBA A language of A :

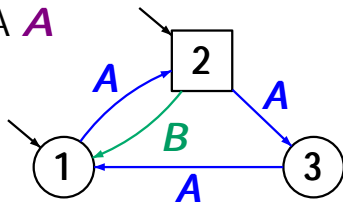
$$A.(B.A + A.A.A) + (B.A + A.A.A)$$

$$L_{12} = A.(B.A + A.A.A)$$

$$L_{22} = (B.A + A.A.A)^+$$

NFA A_{12} 

NBA A



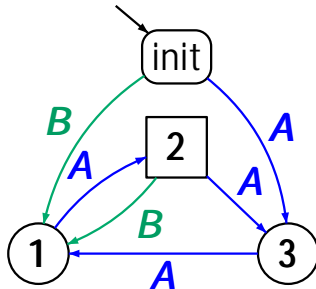
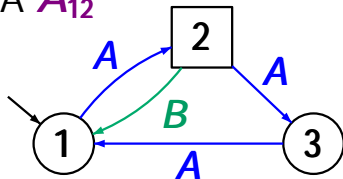
language of A :

$$A.(B.A + A.A.A) + (B.A + A.A.A)(A +).(B.A + A.A.A)$$

$$L_{12} = A.(B.A + A.A.A)$$

$$L_{22} = (B.A + A.A.A)^+$$

NFA A_{12}



For each ϵ -regular expression

$$= 1 \cdot 1 + \dots + n \cdot n$$

there exists an NBA A with $L(A) = L(\epsilon)$.

For each ϵ -regular expression

$$= 1 \cdot 1 + \dots + n \cdot n$$

there exists an NBA A with $L(A) = L(\epsilon)$.

Proof.

For each ϵ -regular expression

$$= r_1 \cdot r_1 + \dots + r_n \cdot r_n$$

there exists an NBA A with $L(A) = L(r)$.

Proof. consider NFA A_i for r_i and B_i for r_i

For each ϵ -regular expression

$$= r_1 \cdot r_1 + \dots + r_n \cdot r_n$$

there exists an NBA A with $L(A) = L(r)$.

Proof. consider NFA A_i for r_i and B_i for r_i

we construct NBA B_i for r_i

For each ϵ -regular expression

$$= r_1 \cdot r_2 + \dots + r_n \cdot r_n$$

there exists an NBA A with $L(A) = L(r)$.

Proof. consider NFA A_i for r_i and B_i for r_i

Also construct NBA B_i for r_i

Also construct NBA $C_i = A_i B_i$ for $r_i \cdot r_i$

For each ϵ -regular expression

$$= r_1 \cdot r_1 + \dots + r_n \cdot r_n$$

there exists an NBA A with $L(A) = L(r)$.

Proof. consider NFA A_i for r_i and B_i for r_i

do construct NBA B_i for r_i

do construct NBA $C_i = A_i B_i$ for $r_i \cdot r_i$

do construct NBA for $\bigcup_{i=1}^n L(C_i)$

For each ϵ -regular expression

$$= r_1 \cdot r_1 + \dots + r_n \cdot r_n$$

there exists an NBA A with $L(A) = L(r)$.

Proof. consider NFA A_i for r_i and B_i for r_i

\hookrightarrow construct NBA B_i for r_i

\hookrightarrow construct NBA $C_i = A_i B_i$ for $r_i \cdot r_i$

\hookrightarrow construct NBA for $\bigcup_{i=1}^n L(C_i)$

\hookrightarrow

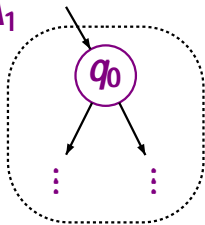
NBA are closed under union

ltlmc3.2-28

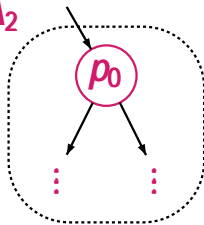
NBA are closed under union

ltlmc3.2-28

NBA A_1



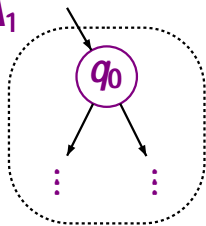
NBA A_2



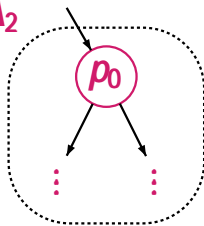
NBA are closed under union

ltlmc3.2-28

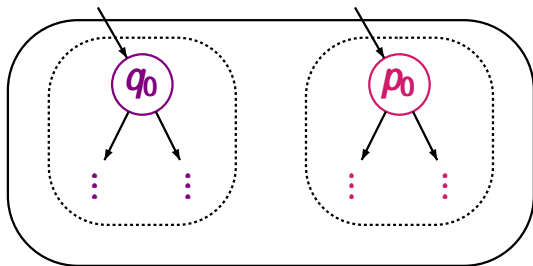
NBA A_1



NBA A_2



NBA for $L(A_1)$ $L(A_2)$



For each ϵ -regular expression

$$= r_1 \cdot r_1 + \dots + r_n \cdot r_n$$

there exists an NBA A with $L(A) = L(r)$.

Proof. consider NFA A_i for r_i and B_i for r_i

$\%o$ construct NBA B_i for r_i

$\%o$ construct NBA $C_i = A_i B_i$ for $r_i \cdot r_i$

$\%o$

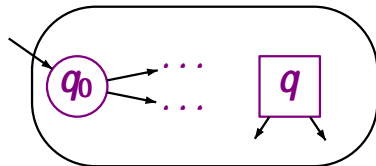
$\%o$ construct NBA for $L(C_i)$

$1 \leq i \leq n$

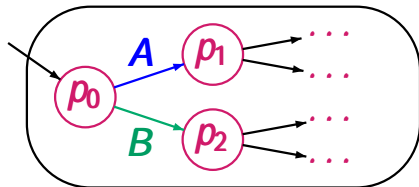
Concatenation of an NFA and an NBA

ltlmc3.2-29

NFA A_1



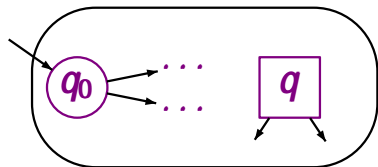
NBA A_2



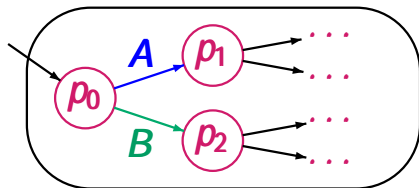
Concatenation of an NFA and an NBA

ltlmc3.2-29

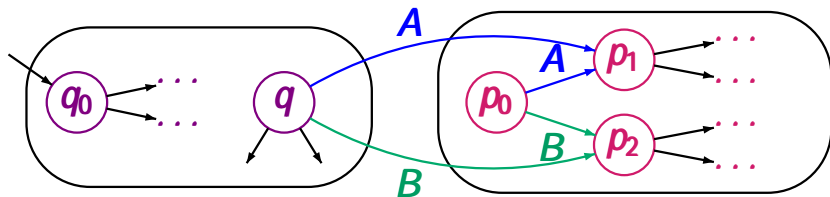
NFA A_1



NBA A_2



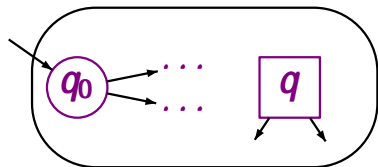
NBA for $L(A_1).L(A_2)$:



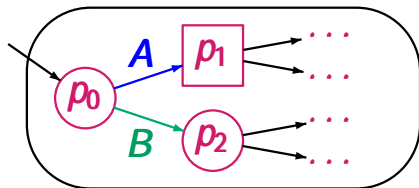
Concatenation of an NFA and an NBA

ltlmc3.2-29

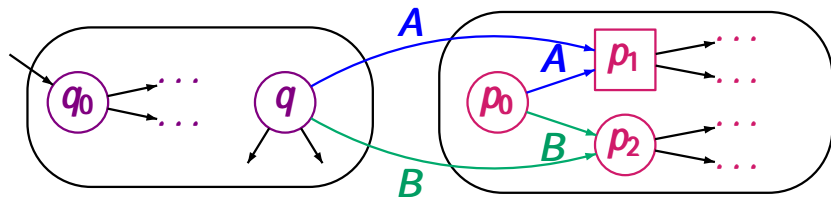
NFA A_1



NBA A_2



NBA for $L(A_1).L(A_2)$:

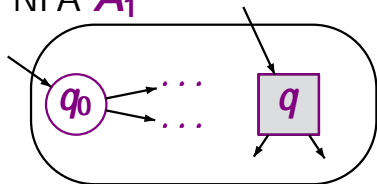


accept states as in A_2

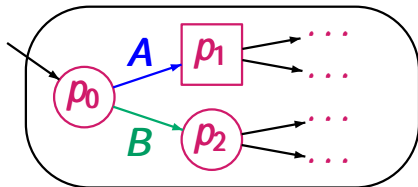
Concatenation of an NFA and an NBA

ltlmc3.2-29

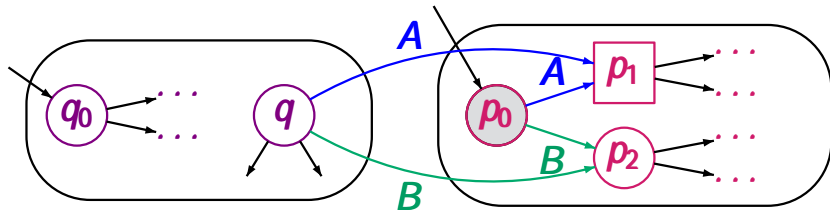
NFA A_1



NBA A_2



NBA for $L(A_1).L(A_2)$:



accept states as in A_2

For each ϵ -regular expression

$$= r_1 \cdot r_1 + \dots + r_n \cdot r_n$$

there exists an NBA A with $L(A) = L(r)$.

Proof. consider NFA A_i for r_i and B_i for r_i

Do construct NBA B_i for r_i

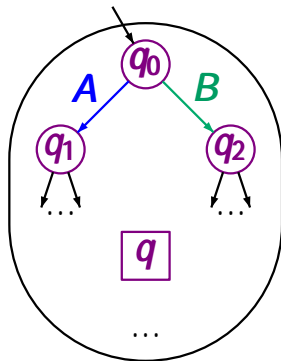
Do

Do construct NBA $C_i = A_i B_i$ for $r_i \cdot r_i$

Do construct NBA for $L(C_i)$

1 i n

NFA A for language
 L +

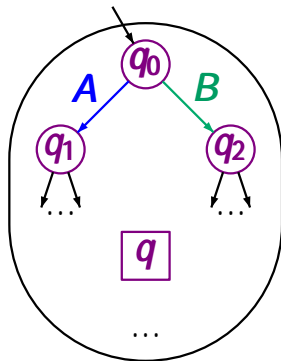


NBA A for language
 L

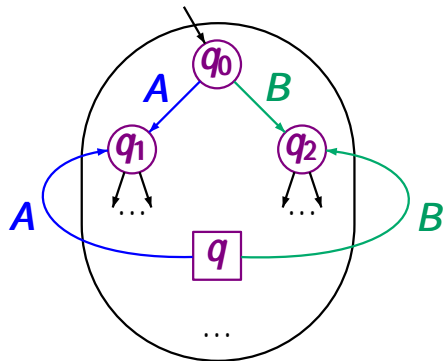
-operator for NFA

l1mc3.2-30

NFA A for language L +



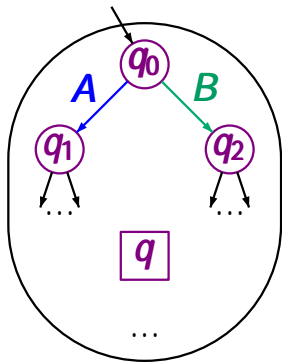
NBA A for language L



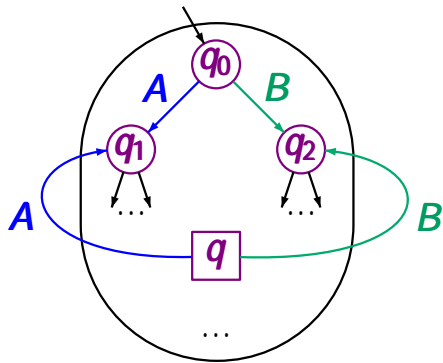
-operator for NFA

ltlmc3.2-30

NFA A for language
 L +



NBA A for language
 L

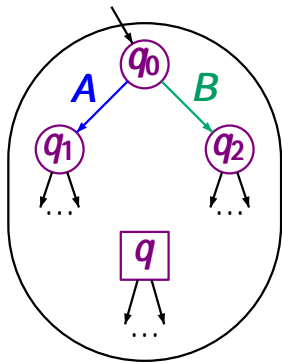


wrong !

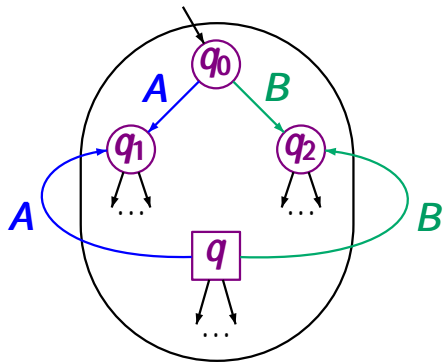
-operator for NFA

ltlmc3.2-30

NFA A for language
 L +



NBA A for language
 L



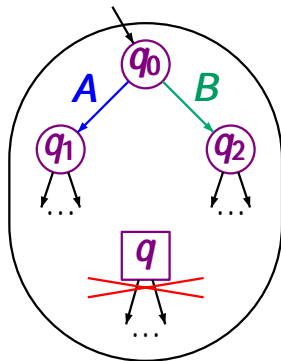
wrong !

-operator for NFA

ltlmc3.2-30

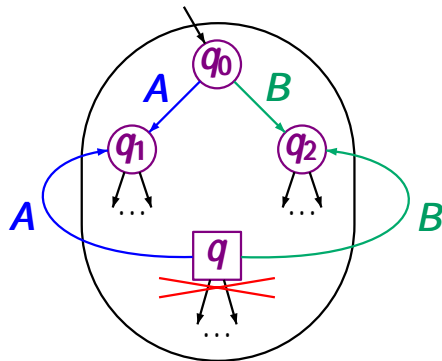
NFA A for language

L +



NBA A for language

L

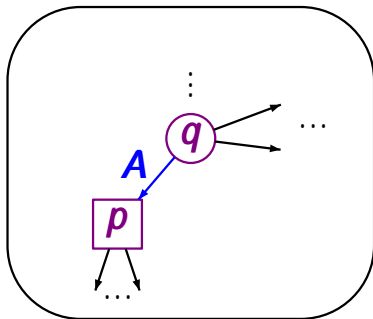


wrong !

... correct, if $(q, x) = q \ F \ x$

NFA A for language L $+$ =

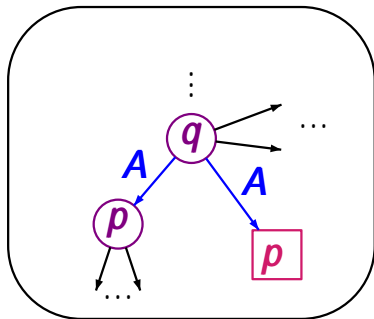
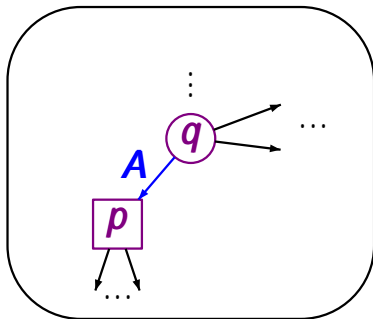
NFA B for L s.t. all states are terminal



-operator for NFA

ltlmc3.2-31

NFA A for language L $+$ = NFA B for L s.t. all ϵ states are terminal



... add a new ϵ state p ...

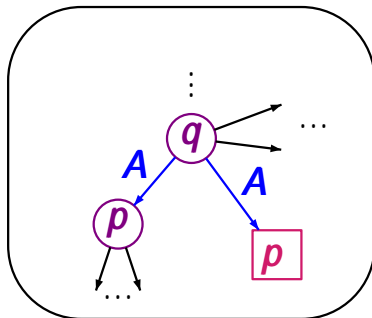
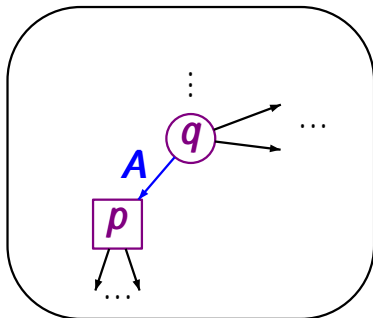
-operator for NFA

ltlmc3.2-31

NFA A for language L $+$ $=$

NFA B for L s.t. all $\forall q$ states are terminal

NBA B



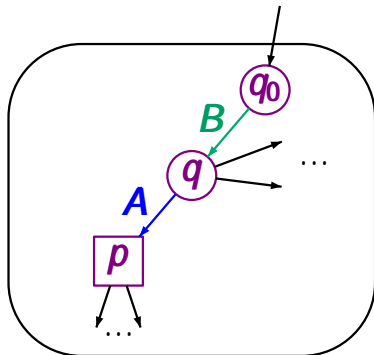
... add a new $\forall q$ state p ...

-operator for NFA

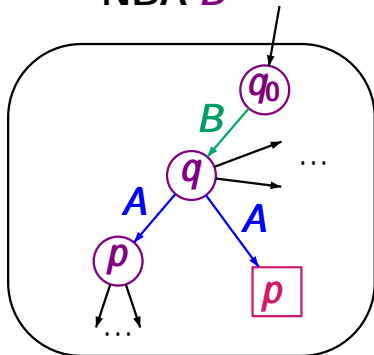
ltlmc3.2-31

NFA A for language L $+$ =

NFA B for L s.t. all $\forall q$ states are terminal



NFA B



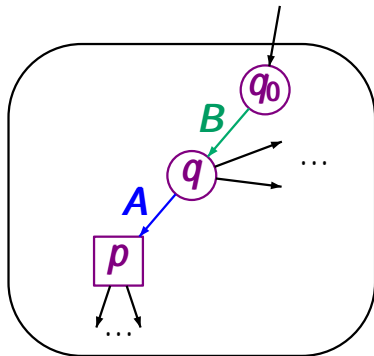
... add a new $\forall q$ state p ...

-operator for NFA

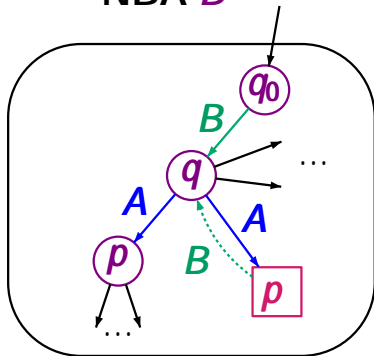
ltlmc3.2-31

NFA A for language L $+$ =

NFA B for L s.t. all \forall states are terminal



NFA B



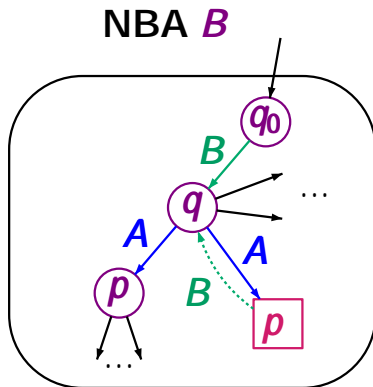
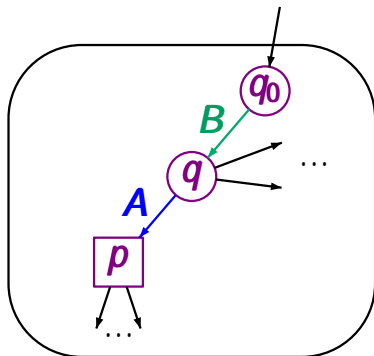
... add a new \forall state p ...

-operator for NFA

ltlmc3.2-31

NFA **A** for language L $+$ =

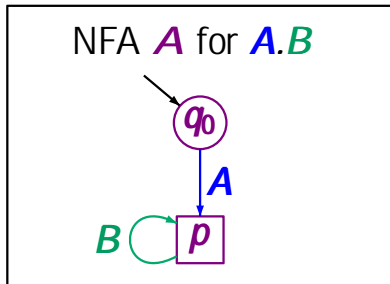
NFA **B** for L s.t. all \forall states are terminal



$$L(A) = L(B)$$

Example: -operator for NFA

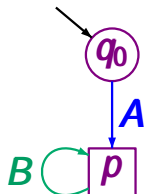
ltlmc3.2-32



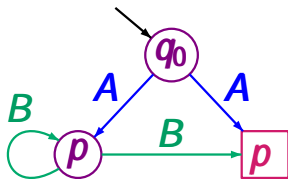
Example: -operator for NFA

ltlmc3.2-32

NFA A for $A.B$



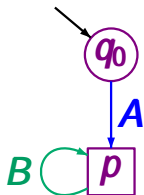
NFA B for $A.B$



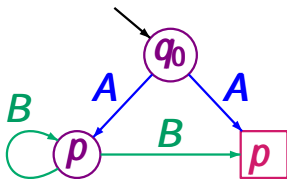
Example: -operator for NFA

ltlmc3.2-32

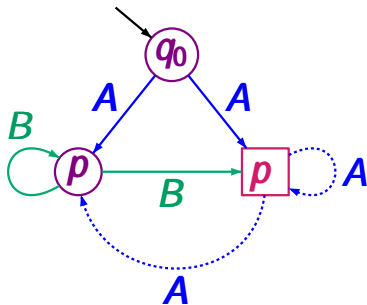
NFA A for $A.B$



NFA B for $A.B$



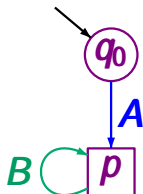
NBA B



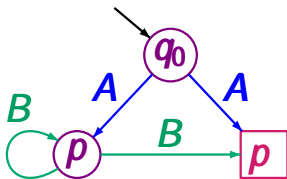
Example: -operator for NFA

ltlmc3.2-32

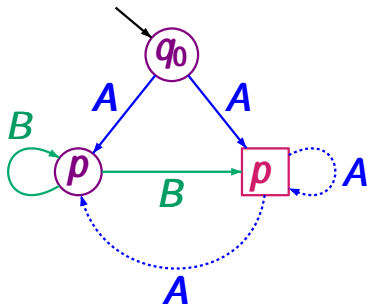
NFA **A** for **A.B**



NFA **B** for **A.B**



NBA **B** for (**A.B**)



- (1) For each NBA A there exists an ϵ -regular expression with $L(A) = L(\epsilon)$
- (2) For each ϵ -regular expression there exists an NBA A with $L(A) = L(\epsilon)$

- (1) For each NBA A there exists an λ -regular expression with $L(A) = L(E)$
- (2) For each λ -regular expression E there exists an NBA A with $L(A) = L(E)$

Corollary:

If E be an LT property then:

E is λ -regular i $E = L(A)$ for some NBA A

- (1) For each NBA A there exists an ϵ -regular expression with $L(A) = L(\epsilon)$
- (2) For each ϵ -regular expression there exists an NBA A with $L(A) = L(\epsilon)$

Corollary:

If E be an LT property, i.e., $E \in 2^{AP}$, then:

E is ϵ -regular i $E = L(A)$ for some NBA A over the alphabet 2^{AP}

Closure properties of Σ^* -regular properties

lt1mc3.2-32b

remind: Kleene's theorem for regular languages:

The class of **regular languages** is closed under

\cup union, intersection, complementation

\cdot concatenation and Kleene star

remind: Kleene's theorem for regular languages:

The class of **regular languages** is closed under
 ϵ **union, intersection, complementation**
 ϵ concatenation and Kleene star

The class of **ϵ -regular languages** is closed under
union, intersection and **complementation**.

The class of ϵ -regular languages is closed under union, intersection and complementation.

ϵ -union:

ϵ -intersection:

ϵ -complementation:

The class of ϵ -regular languages is closed under union, intersection and complementation.

~~%~~union:

obvious from definition of ϵ -regular expressions

~~%~~intersection:

~~%~~complementation:

The class of ϵ -regular languages is closed under union, intersection and complementation.

~~Proof~~ *union:*

obvious from definition of ϵ -regular expressions

~~Proof~~ *intersection:*

will be discussed later

relies on a certain product construction for NBA

~~Proof~~ *complementation:*

The class of ϵ -regular languages is closed under union, intersection and complementation.

ϵ union:

obvious from definition of ϵ -regular expressions

ϵ intersection:

will be discussed later

relies on a certain product construction for NBA

ϵ complementation:

much more difficult than for NFA,
via other types of ϵ -automata

Nonemptiness for NBA

It1mc3.2-NBA-emptiness

given: NBA $A = (Q, \delta, Q_0, F)$

question: does $L(A) \neq \emptyset$ hold?

Nonemptiness for NBA

It1mc3.2-NBA-emptiness

Let $A = (Q, \delta, q_0, F)$ be an NBA. Then:

$$L(A) = \{ x \in \Sigma^+ \mid \exists p \in Q, y \in \Sigma^+ \text{ such that } (q_0, x) \xrightarrow{p} (p, y) \text{ and } p \in F \}$$

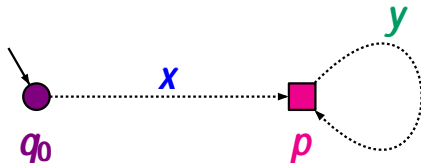
Nonemptiness for NBA

It1mc3.2-NBA-emptiness

Let $A = (Q, \Sigma, q_0, F)$ be an NBA. Then:

$$L(A) = \{ x \in \Sigma^+ \mid \text{there exists a path } q_0 \xrightarrow{x} p \text{ where } p \in F \text{ and } p \text{ is on a cycle} \}$$

there exists a reachable accept state $p \in F$
that belongs to a cycle

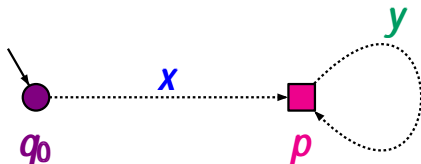


Nonemptiness for NBA

It1mc3.2-NBA-emptiness

Let $A = (Q, \Sigma, Q_0, F)$ be an NBA. Then:

$$L(A) = \{ x \in \Sigma^+ \mid \text{there exist words } y \text{ s.t. } xy \in L(A) \}$$



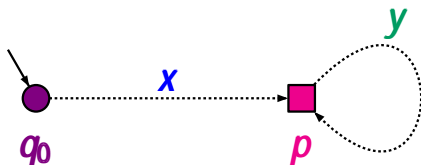
Nonemptiness for NBA

It1mc3.2-NBA-emptiness

Let $A = (Q, \Sigma, q_0, F)$ be an NBA. Then:

$$L(A) = \{ x \mid \text{there exist words } y \text{ s.t. } xy \in L(A) \}$$

Alternatively periodic words



Nonemptiness for NBA

It1mc3.2-NBA-emptiness

Let $A = (Q, \delta, q_0, F)$ be an NBA. Then:

$$L(A) = \{ x y \mid \begin{array}{l} \text{there exist words } x, y \\ \text{s.t. } y = \text{ and } xy \in L(A) \end{array} \}$$

The emptiness problem for NBA is solvable by means of graph algorithms in time $O(\text{poly}(A))$

A DBA is an NBA $A = (Q, \delta, Q_0, F)$ such that $\mathcal{L}_{\text{DBA}} A$ has a unique initial state,

$$\mathcal{L}_{\text{DBA}} (q, A) = 1 \text{ for all } q \in Q \text{ and } A$$

A DBA is an NBA $A = (Q, \delta, Q_0, F)$ such that

$\forall A$ has a unique initial state,
i.e., Q_0 is a singleton

$\forall (q, A) \quad 1$ for all $q \in Q$ and A

A DBA is an NBA $A = (Q, \delta, Q_0, F)$ such that

δ has a unique initial state,
i.e., Q_0 is a singleton

$\delta(q, a) \neq \emptyset$ for all $q \in Q$ and $a \in A$

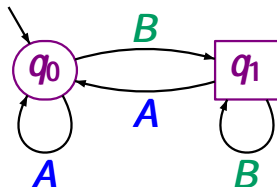
notation: $A = (Q, \delta, q_0, F)$ if $Q_0 = \{q_0\}$

A DBA is an NBA $\mathcal{A} = (Q, \rightarrow, q_0, F)$ such that

q_0 has a unique initial state,
i.e., q_0 is a singleton

$\forall (q, A) \in \rightarrow \quad 1$ for all $q \in Q$ and A

notation: $\mathcal{A} = (Q, \rightarrow, q_0, F)$ if $q_0 = \{q_0\}$



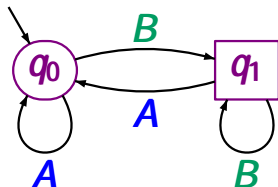
alphabet = $\{A, B\}$

A DBA is an NBA $A = (Q, \delta, q_0, F)$ such that

q_0 has a unique initial state,
i.e., Q_0 is a singleton

$\forall (q, A) \in Q \times \Sigma$ for all $q \in Q$ and $A \in \Sigma$

notation: $A = (Q, \delta, q_0, F)$ if $Q_0 = \{q_0\}$



DBA for \forall often Σ

alphabet $= \{A, B\}$

well-known:

the powerset construction for the
determinization (and complementation) of
NFA (NFA)

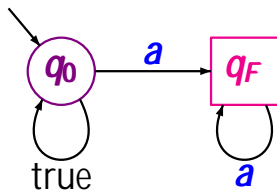
well-known:

the powerset construction for the
determinization (and complementation) of
NFA (NFA)

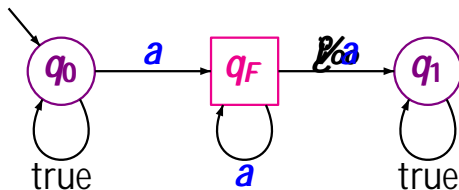
question:

does the powerset construction also work for
BFA (BFA) ?

NBA for φ eventually forever a φ_{∞}



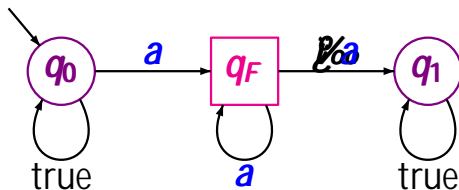
NBA for φ eventually forever a^{ω}



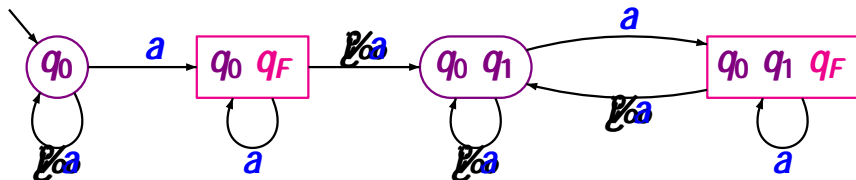
Determinization by powerset construction

ltlmc3.2-82

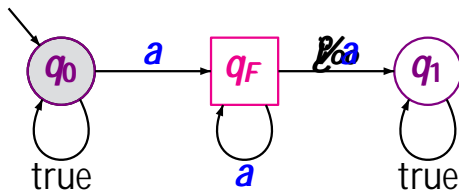
NBA for $\Box \text{ eventually forever } a$



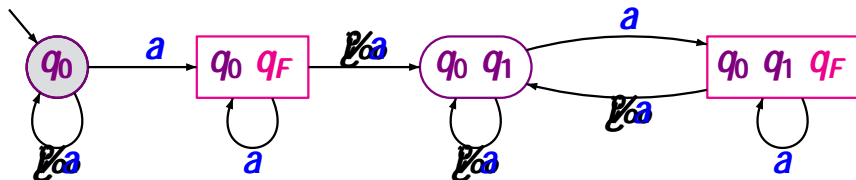
powerset construction



NBA for $\text{Eventually forever } a$

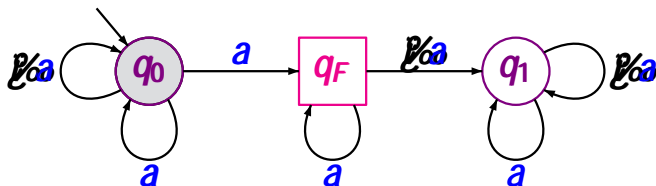


powerset construction

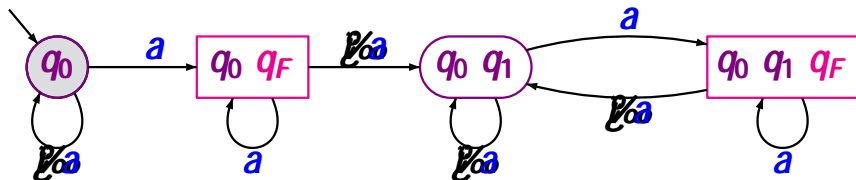


e.g., $(q_0, a) = \{q_0, q_F\}$ and $(q_0, \neg a) = \{q_0\}$

NBA for $\text{Eventually forever } a$

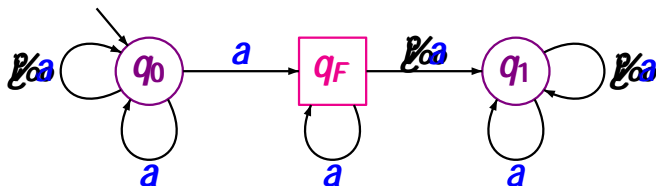


powerset construction

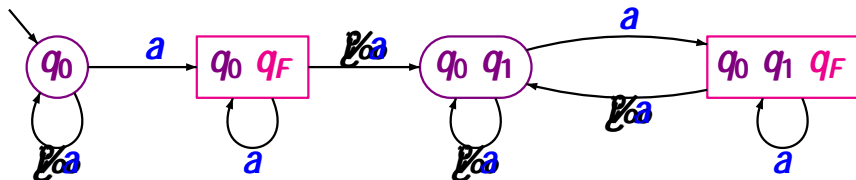


e.g., $(q_0, a) = \{q_0, q_F\}$ and $(q_0, \neg a) = \{q_0\}$

NBA for φ eventually forever a^{ω}

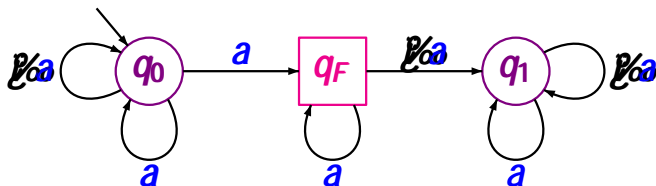


powerset construction

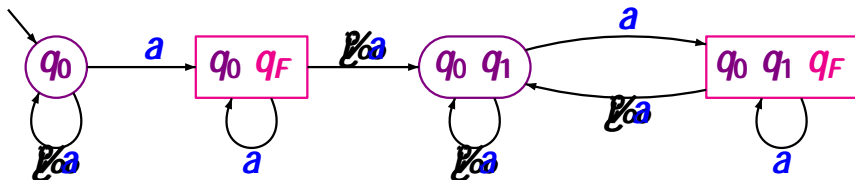


DBA for φ infinitely often a^{ω}

NBA for $\varphi = \text{Eventually forever } a \text{ } \varphi_{\infty}$



powerset construction



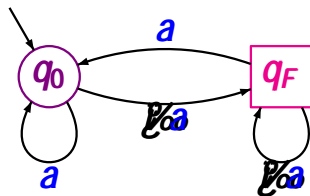
DBA for $\varphi = \text{Infinitely often } a \text{ } \varphi_{\infty}$

well-known:

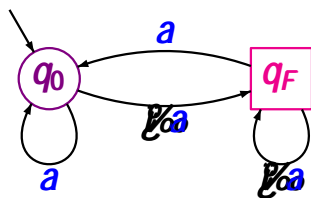
DFA can be complemented by
complementation of the acceptance set

question:

does this also work for **DBA** ?

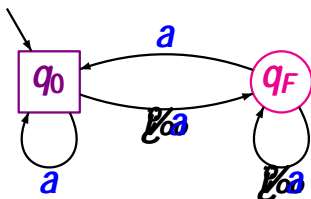


DBA for
 Not often



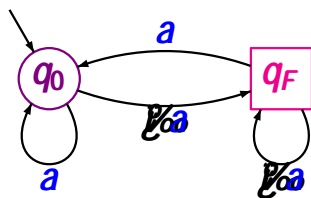
DBA for
 not often ~~not~~

complement automaton



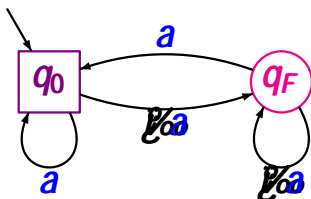
Complementation of DBA

ltlmc3.2-83

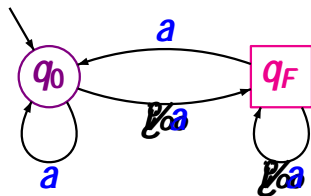


DBA for
eventually often a

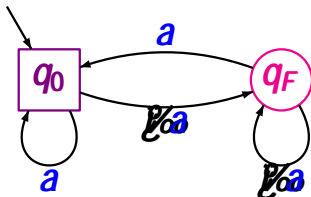
complement automaton



DBA for
eventually often a



complement automaton

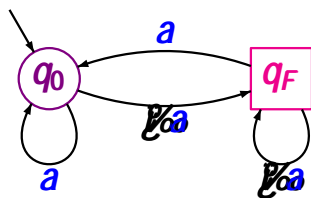


DBA for

 ~~ω~~
 ω
 frequently often ~~ω~~
 ω

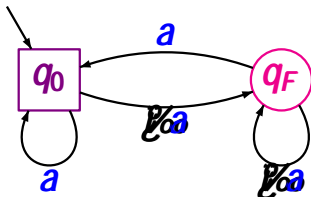
DBA for

 ~~ω~~
 ω
 frequently often a
 ~~ω~~
 ω



complement automaton

DBA for
eventually often ~~ω~~ a ~~ω~~



DBA for
eventually often a ~~ω~~

There is **no DBA** for the LT-property
eventually forever a ~~ω~~

There is no DBA A over the alphabet $= \{A, B\}$
such that $L(A) = L(A + B) \cdot A$

There is no DBA A over the alphabet $\Sigma = \{A, B\}$
 such that $L(A) = L(A + B)^*A$

Hence: there is no DBA for the LT-property
 eventually forever $\forall \infty$

There is no DBA A over the alphabet $\Sigma = \{a, b\}$
 such that $L(A) = L(A + B) \cdot A$

Hence: there is no DBA for the LT-property

~~We eventually forever~~

Proof: apply the above theorem for $A = \{a\}$, $B =$

There is no DBA A over the alphabet $\Sigma = \{A, B\}$
 such that $L(A) = L(A + B)^*$.

Hence: there is no DBA for the LT-property

Eventually forever a^*b^*

Proof: apply the above theorem for $A = \{a\}$, $B = \{b\}$

The class of **DBA-recognizable languages** is a
 proper subclass of the class of **-regular languages**

There is no DBA A over the alphabet $\Sigma = \{A, B\}$ such that $L(A) = L(A + B)^*$.

Hence: there is no DBA for the LT-property
Eventually forever ω

Proof: apply the above theorem for $A = \{a\}$, $B =$

The class of **DBA-recognizable languages** is a proper subclass of the class of **-regular languages** and is not closed under complementation.

There is no DBA A over the alphabet $\Sigma = \{A, B\}$ such that $L(A) = L(A + B) \cdot A$

The class of **DBA-recognizable languages** is a proper subclass of the class of **-regular languages** and is not closed under complementation.

$(A \cdot B)$ infinitely many B DBA-recognizable
 $(A + B) \cdot A$ only finitely many B not DBA-recognizable

A generalized nondeterministic Buchi automaton is a tuple

$$G = (Q, \delta, Q_0, F)$$

where Q, δ, Q_0 are as in NBA, but F is a set of **accept sets**, i.e., $F \subseteq 2^Q$.

A generalized nondeterministic Buchi automaton is a tuple

$$G = (Q, \delta, Q_0, F)$$

where Q, δ, Q_0 are as in NBA, but F is a set of **accept sets**, i.e., $F \subseteq 2^Q$.

A run $q_0 q_1 q_2 \dots$ for some infinite word is called **accepting** if **each accept set** is visited infinitely often

A generalized nondeterministic Buchi automaton is a tuple

$$G = (Q, \delta, Q_0, F)$$

where Q, δ, Q_0 are as in NBA, but F is a set of **accept sets**, i.e., $F \subseteq 2^Q$.

A run $q_0 q_1 q_2 \dots$ for some infinite word is called **accepting** if **each accept set** is visited infinitely often, i.e.,

$$\forall F \in F \quad \exists i \in \mathbb{N} \text{ s.t. } q_i \in F$$

GNBA $G = (Q, \delta, Q_0, F)$ as NBA, but $F \subseteq 2^Q$

A run $q_0 q_1 q_2 \dots$ for some infinite word w is accepting if

$$F \cap \{q_i \mid i \in \mathbb{N}\} \neq \emptyset \text{ s.t. } q_i \in F$$

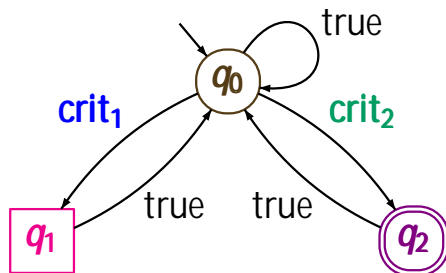
accepted language:

$L(G) \stackrel{\text{def}}{=} \{w \mid w \text{ has an accepting run in } G\}$

Example: GNBA for liveness property

ltlmc3.2-40a

GNBA G over $= 2^{AP}$ where $AP = \{\text{crit}_1, \text{crit}_2\}$

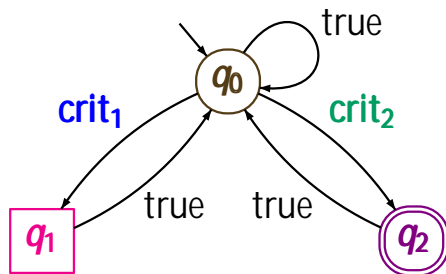


$$F = \{q_1\}, \{q_2\}$$

Example: GNBA for liveness property

ltlmc3.2-40a

GNBA G over $= 2^{AP}$ where $AP = \{\text{crit}_1, \text{crit}_2\}$



$$F = \{q_1\}, \{q_2\}$$

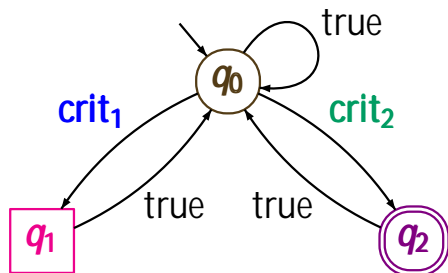
specifies the LT-property

infinitely often crit_1 and infinitely often crit_2

Example: GNBA for liveness property

ltlmc3.2-40a

GNBA G over $= 2^{AP}$ where $AP = \{\text{crit}_1, \text{crit}_2\}$



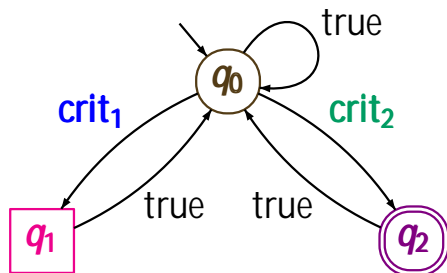
$$F = \{q_1\}, \{q_2\}$$

note: $q_0 \not\models^A q_1$ implies $A \not\models \text{crit}_1$
 $q_0 \not\models^A q_2$ implies $A \not\models \text{crit}_2$

Example: GNBA for liveness property

ltlmc3.2-40a

GNBA G over $= 2^{AP}$ where $AP = \{\text{crit}_1, \text{crit}_2\}$



$$F = \{q_1\}, \{q_2\}$$

note: $q_0 \not\models^A q_1$ implies $A \not\models \text{crit}_1$

$q_0 \not\models^A q_2$ implies $A \not\models \text{crit}_2$

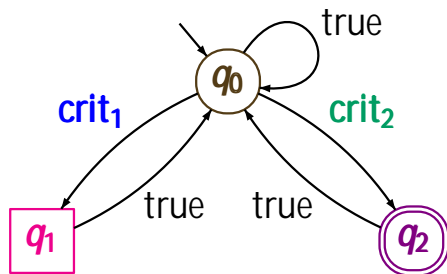
hence: if $A_0 A_1 A_2 \dots \models L(G)$ then

$i = 0. \text{crit}_1 \quad A_i \quad i = 0. \text{crit}_2 \quad A_i$

Example: GNBA for liveness property

ltlmc3.2-40a

GNBA G over $\Sigma = 2^{AP}$ where $AP = \{\text{crit}_1, \text{crit}_2\}$

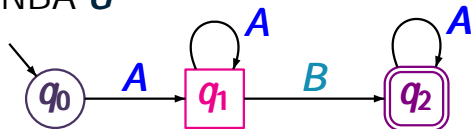


$$F = \{q_1\}, \{q_2\}$$

all words $A_0 A_1 A_2 \dots$ s.t. $i \geq 0$. $\text{crit}_1 \mid A_i$ and $i \geq 0$. $\text{crit}_2 \mid A_i$ have an accepting run of the form:

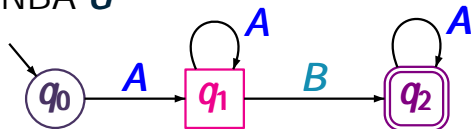
$q_0 \dots q_0 q_1 q_0 \dots q_0 q_2 q_0 \dots q_0 q_1 q_0 \dots q_0 q_2 \dots$

GNBA G



$F = \{q_1\}, \{q_2\}$

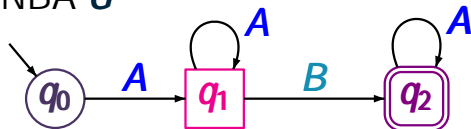
GNBA G



$$F = \{q_1\}, \{q_2\}$$

$$L(G) = ?$$

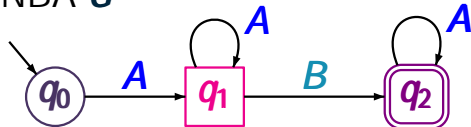
GNBA G



$$F = \{q_1\}, \{q_2\}$$

$$L(G) =$$

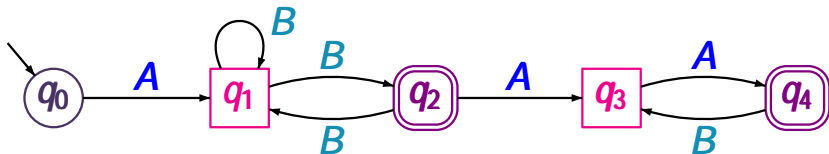
GNBA G



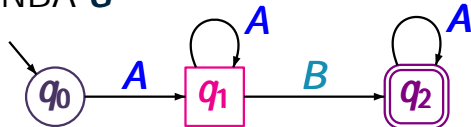
$F = \{q_1\}, \{q_2\}$

$L(G) =$

GNBA G with $F = \{q_1, q_3\}, \{q_2, q_4\}$



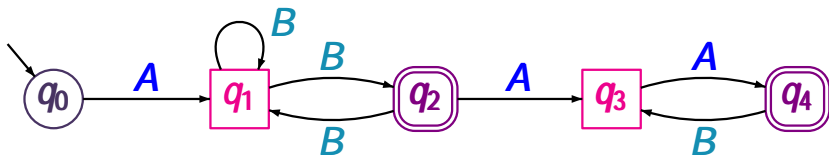
GNBA G



$F = \{q_1\}, \{q_2\}$

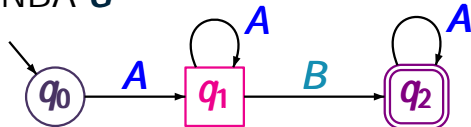
$L(G) =$

GNBA G with $F = \{q_1, q_3\}, \{q_2, q_4\}$



accepted language: ?

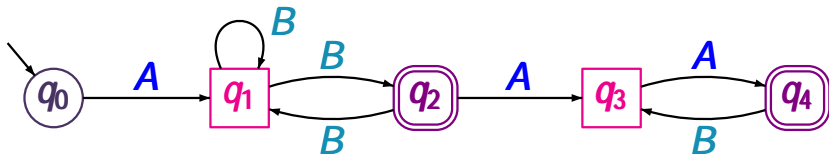
GNBA G



$F = \{q_1\}, \{q_2\}$

$L(G) =$

GNBA G with $F = \{q_1, q_3\}, \{q_2, q_4\}$

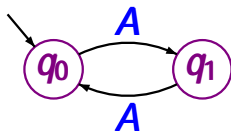


accepted language: $A.B + A.B^+.A.(A.B)$

Empty acceptance condition

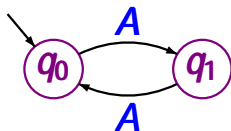
ltlmc3.2-42

NBA \mathbf{A} over $= \{\mathbf{A}, \mathbf{B}\}$:



acceptance set $\mathbf{F} =$

GNBA \mathbf{G} over $= \{\mathbf{A}, \mathbf{B}\}$:

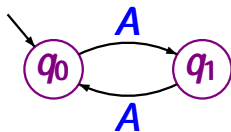


set of acceptance sets
 $\mathbf{F} =$

Empty acceptance condition

ltlmc3.2-42

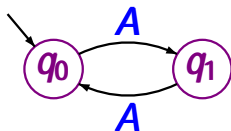
NBA A over $= \{A, B\}$:



acceptance set $F =$

$L(A) =$

GNBA G over $= \{A, B\}$:



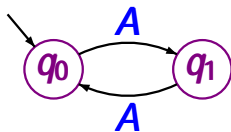
set of acceptance sets

$F =$

Empty acceptance condition

ltlmc3.2-42

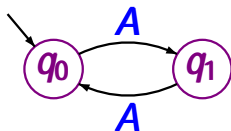
NBA A over $= \{A, B\}$:



acceptance set $F =$

$L(A) =$

GNBA G over $= \{A, B\}$:



set of acceptance sets

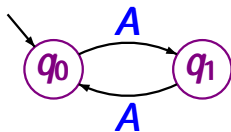
$F =$

$L(G) = ?$

Empty acceptance condition

ltlmc3.2-42

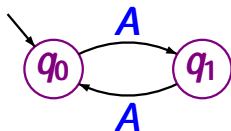
NBA A over $= \{A, B\}$:



acceptance set $F =$

$L(A) =$

GNBA G over $= \{A, B\}$:



set of acceptance sets

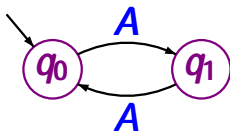
$F =$

$L(G) = A$

Empty acceptance condition

ltlmc3.2-42

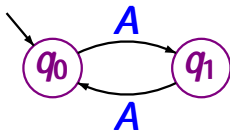
NBA \mathbf{A} over $= \{\mathbf{A}, \mathbf{B}\}$:



acceptance set $\mathbf{F} =$

$\mathbf{L}(\mathbf{A}) =$

GNBA \mathbf{G} over $= \{\mathbf{A}, \mathbf{B}\}$:



set of acceptance sets

$\mathbf{F} =$

$\mathbf{L}(\mathbf{G}) = \mathbf{A}$

$\mathbf{L}(\mathbf{G}) =$ set of all infinite words
that have an infinite run

For every GNBA G there exists a GNBA G such that

$$\not\exists L(G) = L(G)$$

$\not\exists$ the set of acceptance sets of G is nonempty

For every GNBA G there exists a GNBA G such that

$$\mathcal{L}(G) = \mathcal{L}(G)$$

the set of acceptance sets of G is nonempty

correct

$$\text{GNBA } G = (Q, \delta, Q_0, \{Q\})$$

$$\text{GNBA } G = (Q, \delta, Q_0, \{Q\})$$

For each **GNBA** G there exists an **NBA** A with

$$L(G) = L(A)$$

For each **GNBA** G there exists an **NBA** A with

$$L(G) = L(A)$$

Proof. Let $G = (Q, \rightarrow, Q_0, F)$ with $F = \{F_1, \dots, F_k\}$

For each **GNBA** G there exists an **NBA** A with

$$L(G) = L(A)$$

Proof. Let $G = (Q, \rightarrow, Q_0, F)$ with $F = \{F_1, \dots, F_k\}$ and $k \geq 1$

For each **GNBA** G there exists an **NBA** A with

$$L(G) = L(A)$$

Proof. Let $G = (Q, \rightarrow, Q_0, F)$ with $F = \{F_1, \dots, F_k\}$ and $k \geq 1$

note: if $k = 1$ then G is an NBA

For each **GNBA** G there exists an **NBA** A with

$$L(G) = L(A)$$

Proof. Let $G = (Q, \rightarrow, Q_0, F)$ with $F = \{F_1, \dots, F_k\}$
and $k \geq 2$

note: if $k = 1$ then G is an NBA

For each **GNBA** G there exists an **NBA** A with

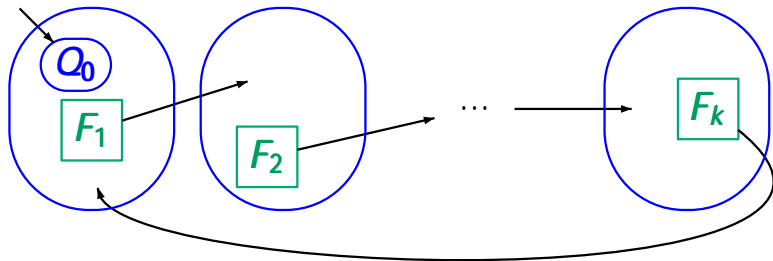
$$L(G) = L(A)$$

Proof. Let $G = (Q, \rightarrow, Q_0, F)$ with $F = \{F_1, \dots, F_k\}$ and $k \geq 2$. NBA A results from k copies of G :

For each **GNBA** G there exists an **NBA** A with

$$L(G) = L(A)$$

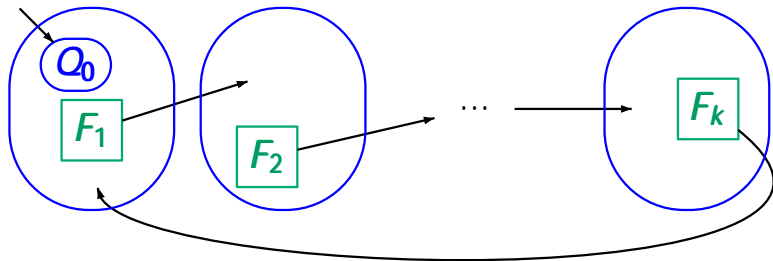
Proof. Let $G = (Q, \rightarrow, Q_0, F)$ with $F = \{F_1, \dots, F_k\}$ and $k \geq 2$. NBA A results from k copies of G :



For each **GNBA** G there exists an **NBA** A with

$$L(G) = L(A)$$

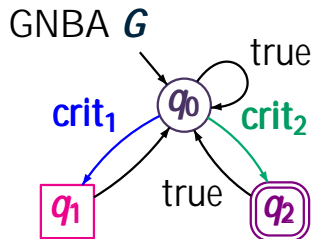
Proof. Let $G = (Q, \rightarrow, Q_0, F)$ with $F = \{F_1, \dots, F_k\}$ and $k \geq 2$. NBA A results from k copies of G :



size of the NBA: $size(A) = O(size(G) \cdot |F|)$

Example: from GNBA to NBA

ltlmc3.2-45

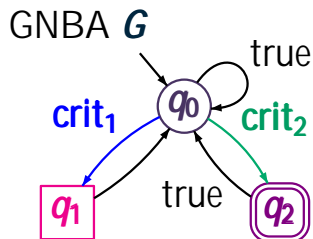


alphabet = 2^{AP} where
 $AP = \{\text{crit}_1, \text{crit}_2\}$

infinitely often crit_1 and
infinitely often crit_2

Example: from GNBA to NBA

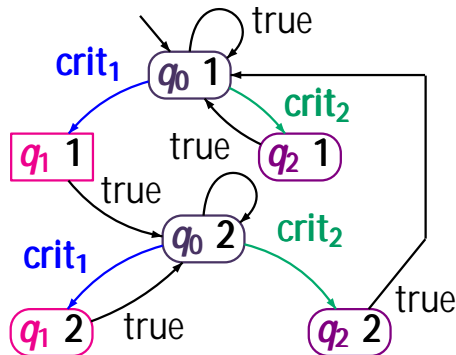
ltlmc3.2-45



alphabet = 2^{AP} where
 $AP = \{\text{crit}_1, \text{crit}_2\}$

infinitely often crit_1 and
infinitely often crit_2

NBA A



The class of ϵ -regular languages is closed under union, intersection and complementation.

The class of ϵ -regular languages is closed under union, intersection and complementation.

~~Proof~~ union:

obvious from definition of ϵ -regular expressions

~~Proof~~ intersection:

via some product construction

~~Proof~~ complementation:

via other types of ϵ -automata
(not discussed here)

The class of ϵ -regular languages is closed under union, intersection and complementation.

~~%~~ union:

obvious from definition of ϵ -regular expressions

~~%~~ intersection:

via some product construction

~~%~~ using **GNBA**

~~%~~ complementation:

via other types of ϵ -automata
(not discussed here)

$$A_1 = (Q_1, \rightarrow, 1, Q_{0,1}, F_1)$$

two NBA

$$A_2 = (Q_2, \rightarrow, 2, Q_{0,2}, F_2)$$

goal: define an NBA A s.t. $L(A) = L(A_1) \cap L(A_2)$

$$A_1 = (Q_1, \delta_1, Q_{0,1}, F_1)$$

two NBA

$$A_2 = (Q_2, \delta_2, Q_{0,2}, F_2)$$

goal: define an NBA A s.t. $L(A) = L(A_1) \cap L(A_2)$

recall:

intersection for the automata NFA A_1 and A_2 is realized by a product construction that

runs A_1 and A_2 in parallel (synchronously)

checks whether both end in a final state

$$A_1 = (Q_1, \quad , \quad 1, Q_{0,1}, F_1)$$

two NBA

$$A_2 = (Q_2, \quad , \quad 2, Q_{0,2}, F_2)$$

goal: define an NBA A s.t. $L(A) = L(A_1) \cap L(A_2)$

idea: define A_1 and A_2 as for NFA, i.e.,

A_1 and A_2 run in parallel (synchronously)

and check whether both are accepting

Intersection for NBA

ltlmc3.2-45a

$$A_1 = (Q_1, \rightarrow, 1, Q_{0,1}, F_1)$$

two NBA

$$A_2 = (Q_2, \rightarrow, 2, Q_{0,2}, F_2)$$

goal: define an NBA A s.t. $L(A) = L(A_1) \cap L(A_2)$

idea: define A_1 and A_2 as for NFA, i.e.,

A_1 and A_2 run in parallel (synchronously)

and check whether both are accepting

i.e., both F_1 and F_2 are visited infinitely often

$A_1 = (Q_1, \delta_1, Q_{0,1}, F_1)$ two **NBA**
 $A_2 = (Q_2, \delta_2, Q_{0,2}, F_2)$

goal: define an **NBA** A s.t. $L(A) = L(A_1) \cap L(A_2)$

idea: define A_1 and A_2 as for **NFA**, i.e.,

A_1 and A_2 run in parallel (synchronously)

and check whether both are accepting

i.e., both F_1 and F_2 are visited infinitely often

product of A_1 and A_2 yields a **GNBA**

Intersection for NBA

ltlmc3.2-45b

$$A_1 = (Q_1, \rightarrow, 1, Q_{0,1}, F_1)$$

two NBA

$$A_2 = (Q_2, \rightarrow, 2, Q_{0,2}, F_2)$$

goal: define an NBA A s.t. $L(A) = L(A_1) \cap L(A_2)$

$$\text{GNBA } G = A_1 \cap A_2$$

Intersection for NBA

ltlmc3.2-45b

$$A_1 = (Q_1, \rightarrow_1, Q_{0,1}, F_1)$$

two NBA

$$A_2 = (Q_2, \rightarrow_2, Q_{0,2}, F_2)$$

goal: define an NBA A s.t. $L(A) = L(A_1) \cap L(A_2)$

$$\text{GNBA } G = A_1 \cap A_2$$

$$\text{state space } Q = Q_1 \cap Q_2$$

Intersection for NBA

ltlmc3.2-45b

$$A_1 = (Q_1, \rightarrow_1, Q_{0,1}, F_1)$$

two NBA

$$A_2 = (Q_2, \rightarrow_2, Q_{0,2}, F_2)$$

goal: define an NBA A s.t. $L(A) = L(A_1) \cap L(A_2)$

$$\text{GNBA } G = A_1 \cap A_2$$

state space $Q = Q_1 \cap Q_2$

alphabet

Intersection for NBA

ltlmc3.2-45b

$$A_1 = (Q_1, \rightarrow_1, Q_{0,1}, F_1)$$

two NBA

$$A_2 = (Q_2, \rightarrow_2, Q_{0,2}, F_2)$$

goal: define an NBA A s.t. $L(A) = L(A_1) \cap L(A_2)$

$$\text{GNBA } G = A_1 \cap A_2$$

$$\text{state space } Q = Q_1 \cap Q_2$$

alphabet

$$\text{set of initial states: } Q_0 = Q_{0,1} \cap Q_{0,2}$$

Intersection for NBA

ltlmc3.2-45b

$$A_1 = (Q_1, \rightarrow_1, Q_{0,1}, F_1)$$

two NBA

$$A_2 = (Q_2, \rightarrow_2, Q_{0,2}, F_2)$$

goal: define an NBA A s.t. $L(A) = L(A_1) \cap L(A_2)$

$$\text{GNBA } G = A_1 \cap A_2$$

$$\text{state space } Q = Q_1 \cap Q_2$$

alphabet

$$\text{set of initial states: } Q_0 = Q_{0,1} \cap Q_{0,2}$$

$$\text{acceptance condition: } F = F_1 \cap F_2$$

Intersection for NBA

ltlmc3.2-45b

$$A_1 = (Q_1, \rightarrow_1, Q_{0,1}, F_1)$$

two NBA

$$A_2 = (Q_2, \rightarrow_2, Q_{0,2}, F_2)$$

goal: define an NBA A s.t. $L(A) = L(A_1) \cap L(A_2)$

$$\text{GNBA } G = A_1 \cap A_2$$

state space $Q = Q_1 \cap Q_2$

alphabet

set of initial states: $Q_0 = Q_{0,1} \cap Q_{0,2}$

acceptance condition: $F = F_1 \cap F_2$

transition relation:

$$((q_1, q_2), a) \in \rightarrow : p_1 \rightarrow_1(q_1, a), p_2 \rightarrow_2(q_2, a)$$

Intersection for NBA

ltlmc3.2-45b

$$A_1 = (Q_1, \rightarrow_1, Q_{0,1}, F_1)$$

two NBA

$$A_2 = (Q_2, \rightarrow_2, Q_{0,2}, F_2)$$

goal: define an NBA A s.t. $L(A) = L(A_1) \cap L(A_2)$

$$\text{GNBA } G = A_1 \cap A_2$$

equivalent NBA A

state space $Q = Q_1 \cap Q_2$

alphabet

set of initial states: $Q_0 = Q_{0,1} \cap Q_{0,2}$

acceptance condition: $F = F_1 \cap F_2$

transition relation:

$$((q_1, q_2), a) \in \rightarrow : q_1 \rightarrow_1(q_1, a), q_2 \rightarrow_2(q_2, a)$$

Summary: -regular languages

It1mc3.2-45c

Summary: -regular languages

It1mc3.2-45c

The class of -regular languages agrees with

~~the~~ the class of languages given by -regular expressions

~~the~~ the class of **NBA**-recognizable languages

~~the~~ the class of **GNBA**-recognizable languages

Summary: ϵ -regular languages

It1mc3.2-45c

The class of ϵ -regular languages agrees with

~~the~~ the class of languages given by ϵ -regular expressions

~~the~~ the class of **NBA**-recognizable languages

~~the~~ the class of **GNBA**-recognizable languages

but **DBA** are strictly less expressive

Summary: -regular languages

It1mc3.2-45c

The class of -regular languages agrees with

~~the~~ the class of languages given by -regular expressions

~~the~~ the class of **NBA**-recognizable languages

~~the~~ the class of **GNBA**-recognizable languages

but **DBA** are strictly less expressive

The class of -regular languages is closed under union, intersection and complementation.