BILL AND NATHAN RECORD LECTURE!!!!

BILL RECORD LECTURE!!!

Nondeterministic Finite Automata (NFA): Closure Properties

Terminology: Reg Langs

Def A lang L is **reg** if there exists a DFA M such that $L = L(M)$.

Terminology: Reg Langs

Def A lang L is **reg** if there exists a DFA M such that $L = L(M)$.

Since DFA's and NFA's are equivalent.
Def A lang L is **reg** if there exists an NFA M such that $L = L(M)$.
We use this definition of reg for this slide packet.

Terminology: Reg Langs

Def A lang L is **reg** if there exists a DFA M such that $L = L(M)$.

Since DFA's and NFA's are equivalent.
Def A lang L is **reg** if there exists an NFA M such that $L = L(M)$.
We use this definition of reg for this slide packet.

We prove closure properties (or say NO, not going to prove it) of reg langs using NFA's.

Terminology: Reg Langs

Def A lang L is **reg** if there exists a DFA M such that $L = L(M)$.

Since DFA's and NFA's are equivalent.
Def A lang L is **reg** if there exists an NFA M such that $L = L(M)$.
We use this definition of reg for this slide packet.

We prove closure properties (or say NO, not going to prove it) of reg langs using NFA's.

We will keep track of number-of-states.

Reg Langs Closed Under Complementation

How do you complement a reg lang (not a joke)?

Reg Langs Closed Under Complementation

How do you complement a reg lang (not a joke)?

Caution Swapping the final and non-final states DOES NOT WORK for an NFA.

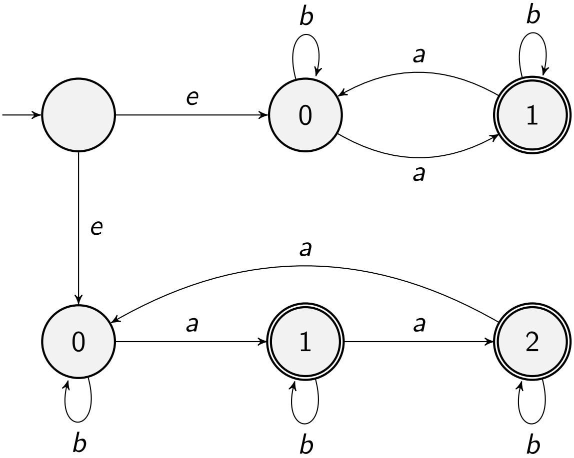
Reg Langs Closed Under Complementation

How do you complement a reg lang (not a joke)?

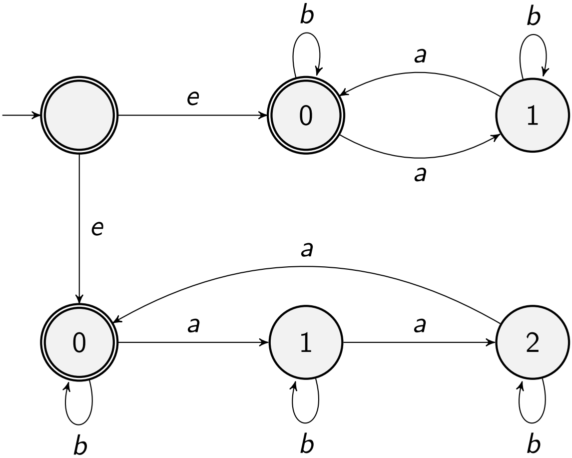
Caution Swapping the final and non-final states DOES NOT WORK for an NFA.

See next slide.

$\{a^n : n \not\equiv 0 \pmod 6\}$



Final and Non-final States Swapped



Reg Langs Closed Under Complementation (cont)

Upshot It is not possible (or very clunky) to prove closure under complementation using JUST NFA's.
Can Use NFA-DFA equivalence:

Reg Langs Closed Under Complementation (cont)

Upshot It is not possible (or very clunky) to prove closure under complementation using JUST NFA's.
Can Use NFA-DFA equivalence:

L recognized by an n-state NFA.

Reg Langs Closed Under Complementation (cont)

Upshot It is not possible (or very clunky) to prove closure under complementation using JUST NFA's.
Can Use NFA-DFA equivalence:

L recognized by an n-state NFA.

Convert to a 2^n-state DFA.

Reg Langs Closed Under Complementation (cont)

Upshot It is not possible (or very clunky) to prove closure under complementation using JUST NFA's.
Can Use NFA-DFA equivalence:

L recognized by an n-state NFA.

Convert to a 2^n-state DFA.

Take the complement.

Reg Langs Closed Under Complementation (cont)

Upshot It is not possible (or very clunky) to prove closure under complementation using JUST NFA's.
Can Use NFA-DFA equivalence:

L recognized by an n-state NFA.

Convert to a 2^n-state DFA.

Take the complement.

Now you have a 2^n state DFA, and hence a 2^n-state NFA for $\overline{L}$.

Reg Langs Closed Under Complementation (cont)

Upshot It is not possible (or very clunky) to prove closure under complementation using JUST NFA's.
Can Use NFA-DFA equivalence:

L recognized by an n-state NFA.

Convert to a 2^n-state DFA.

Take the complement.

Now you have a 2^n state DFA, and hence a 2^n-state NFA for $\overline{L}$.

Is there a more efficient proof?

Reg Langs Closed Under Complementation (cont)

Upshot It is not possible (or very clunky) to prove closure under complementation using JUST NFA's.
Can Use NFA-DFA equivalence:

L recognized by an n-state NFA.

Convert to a 2^n-state DFA.

Take the complement.

Now you have a 2^n state DFA, and hence a 2^n-state NFA for $\overline{L}$.

Is there a more efficient proof?
No. There are langs L where:

Reg Langs Closed Under Complementation (cont)

Upshot It is not possible (or very clunky) to prove closure under complementation using JUST NFA's.
Can Use NFA-DFA equivalence:

L recognized by an n-state NFA.

Convert to a 2^n-state DFA.

Take the complement.

Now you have a 2^n state DFA, and hence a 2^n-state NFA for $\overline{L}$.

Is there a more efficient proof?
No. There are langs L where:

- there is an NFA for L is size n.

Reg Langs Closed Under Complementation (cont)

Upshot It is not possible (or very clunky) to prove closure under complementation using JUST NFA's.
Can Use NFA-DFA equivalence:

L recognized by an n-state NFA.

Convert to a 2^n-state DFA.

Take the complement.

Now you have a 2^n state DFA, and hence a 2^n-state NFA for $\overline{L}$.

Is there a more efficient proof?
No. There are langs L where:

- there is an NFA for L is size n.
- any NFA for $\overline{L}$ is of size $\geq\sim 2^n$.
  See next slide for this example.

Example of a Blowup for Complementation

Example of a language L_n such that

Example of a Blowup for Complementation

Example of a language L_n such that

1. There is an NFA for L that is small.

Example of a Blowup for Complementation

Example of a language L_n such that

1. There is an NFA for L that is small.
2. Every NFA for $\overline{L}$ is large.

Example of a Blowup for Complementation

Example of a language L_n such that

1. There is an NFA for L that is small.
2. Every NFA for $\overline{L}$ is large.

Let M_n be the product of the first n primes.

Example of a Blowup for Complementation

Example of a language L_n such that

1. There is an NFA for L that is small.
2. Every NFA for $\overline{L}$ is large.

Let M_n be the product of the first n primes.

$$L_n = \{a^i : i \not\equiv M_n \pmod{M_n}\}.$$

Example of a Blowup for Complementation

Example of a language L_n such that

1. There is an NFA for L that is small.
2. Every NFA for $\overline{L}$ is large.

Let M_n be the product of the first n primes.

$$L_n = \{a^i : i \not\equiv M_n \pmod{M_n}\}.$$

1. There is an NFA for L_n of size $O(p_1 + \cdots + p_n) = O(\frac{n^2}{\log(n)^2})$.

Example of a Blowup for Complementation

Example of a language L_n such that

1. There is an NFA for L that is small.
2. Every NFA for $\overline{L}$ is large.

Let M_n be the product of the first n primes.

$$L_n = \{a^i : i \not\equiv M_n \pmod{M_n}\}.$$

1. There is an NFA for L_n of size $O(p_1 + \cdots + p_n) = O(\frac{n^2}{\log(n)^2})$.
2. Any NFA for $\overline{L_n}$ requires size $\Omega(p_1 p_2 \cdots p_n) = \Omega(e^{n \log n})$.

Reg Langs Closed Under Union-Intuition

IF L_1, L_2 are reg we want to show that $L_1 \cup L_2$ is reg.

Reg Langs Closed Under Union-Intuition

IF L_1, L_2 are reg we want to show that $L_1 \cup L_2$ is reg.

Informally Create an NFA that branches both ways with e-transitions.

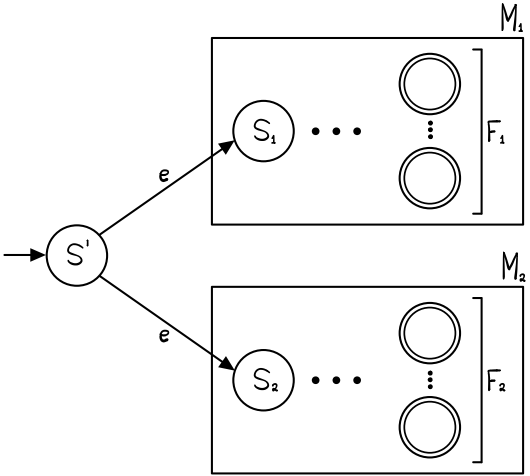
Reg Langs Closed Under Union-Intuition

IF L_1, L_2 are reg we want to show that $L_1 \cup L_2$ is reg.

Informally Create an NFA that branches both ways with e-transitions.

See next slide.

Reg Langs Closed Under Union-Picture



Reg Langs Closed Under Union-Formally

Formally If L_1 is reg via NFA

$$(Q_1, \Sigma, \Delta_1, s_1, F_1). \text{ We will take } |Q_1| = n_1 \ .$$

and L_2 is reg via NFA

$$(Q_2, \Sigma, \Delta_2, s_2, F_2). \text{ We will take } |Q_2| = n_2 \ .$$

then $L_1 \cup L_2$ is reg via NFA

Reg Langs Closed Under Union-Formally

Formally If L_1 is reg via NFA

$$(Q_1, \Sigma, \Delta_1, s_1, F_1). \text{ We will take } |Q_1| = n_1 \ .$$

and L_2 is reg via NFA

$$(Q_2, \Sigma, \Delta_2, s_2, F_2). \text{ We will take } |Q_2| = n_2 \ .$$

then $L_1 \cup L_2$ is reg via NFA

$$(\{s'\} \cup Q_1 \cup Q_2, \Sigma, \Delta', s', F_1 \cup F_2).$$

where for $i = 1$ or 2,
If $q \in Q_i$, $\sigma \in \Sigma \cup \{e\}$ then $\Delta'(q, \sigma) = \Delta_i(q, \sigma)$.

Reg Langs Closed Under Union-Formally

Formally If L_1 is reg via NFA

$$(Q_1, \Sigma, \Delta_1, s_1, F_1). \text{ We will take } |Q_1| = n_1 \ .$$

and L_2 is reg via NFA

$$(Q_2, \Sigma, \Delta_2, s_2, F_2). \text{ We will take } |Q_2| = n_2 \ .$$

then $L_1 \cup L_2$ is reg via NFA

$$(\{s'\} \cup Q_1 \cup Q_2, \Sigma, \Delta', s', F_1 \cup F_2).$$

where for $i = 1$ or 2,
If $q \in Q_i$, $\sigma \in \Sigma \cup \{e\}$ then $\Delta'(q, \sigma) = \Delta_i(q, \sigma)$.
$\Delta'(s', e) = \{s_1, s_2\}$.

Reg Langs Closed Under Union-Formally

Formally If L_1 is reg via NFA

$$(Q_1, \Sigma, \Delta_1, s_1, F_1). \text{ We will take } |Q_1| = n_1 \ .$$

and L_2 is reg via NFA

$$(Q_2, \Sigma, \Delta_2, s_2, F_2). \text{ We will take } |Q_2| = n_2 \ .$$

then $L_1 \cup L_2$ is reg via NFA

$$(\{s'\} \cup Q_1 \cup Q_2, \Sigma, \Delta', s', F_1 \cup F_2).$$

where for $i = 1$ or 2,
If $q \in Q_i$, $\sigma \in \Sigma \cup \{e\}$ then $\Delta'(q, \sigma) = \Delta_i(q, \sigma)$.
$\Delta'(s', e) = \{s_1, s_2\}$.
Note The number of states in NFA for $L_1 \cup L_2$ is $n_1 + n_2 + 1$.

Reg Langs Closed Under Union-Formally

Formally If L_1 is reg via NFA

$$(Q_1, \Sigma, \Delta_1, s_1, F_1). \text{ We will take } |Q_1| = n_1 \ .$$

and L_2 is reg via NFA

$$(Q_2, \Sigma, \Delta_2, s_2, F_2). \text{ We will take } |Q_2| = n_2 \ .$$

then $L_1 \cup L_2$ is reg via NFA

$$(\{s'\} \cup Q_1 \cup Q_2, \Sigma, \Delta', s', F_1 \cup F_2).$$

where for $i = 1$ or 2,
If $q \in Q_i$, $\sigma \in \Sigma \cup \{e\}$ then $\Delta'(q, \sigma) = \Delta_i(q, \sigma)$.
$\Delta'(s', e) = \{s_1, s_2\}$.
Note The number of states in NFA for $L_1 \cup L_2$ is $n_1 + n_2 + 1$.
Note When we did closure using DFA's, we got $n_1 n_2$.

Reg Langs Closed Under Intersection

IF L_1, L_2 are reg we want to show that $L_1 \cap L_2$ is reg.

Reg Langs Closed Under Intersection

IF L_1, L_2 are reg we want to show that $L_1 \cap L_2$ is reg.

Vote

Reg Langs Closed Under Intersection

IF L_1, L_2 are reg we want to show that $L_1 \cap L_2$ is reg.

Vote

1. Impossible or clunky to do with NFAs.

Reg Langs Closed Under Intersection

IF L_1, L_2 are reg we want to show that $L_1 \cap L_2$ is reg.

Vote

1. Impossible or clunky to do with NFAs.
2. One CAN do this with NFAs but still gets $n_1 n_2$ states.

Reg Langs Closed Under Intersection

IF L_1, L_2 are reg we want to show that $L_1 \cap L_2$ is reg.

Vote

1. Impossible or clunky to do with NFAs.
2. One CAN do this with NFAs but still gets $n_1 n_2$ states.
3. One CAN do this with NFAs and we get $< n_1 n_2$ states.

Reg Langs Closed Under Intersection

IF L_1, L_2 are reg we want to show that $L_1 \cap L_2$ is reg.

Vote

1. Impossible or clunky to do with NFAs.
2. One CAN do this with NFAs but still gets $n_1 n_2$ states.
3. One CAN do this with NFAs and we get $< n_1 n_2$ states.

Answer Option 2: Can do with NFAs but gets $n_1 n_2$ states.
It is a cross product construction. Next Slide.

Reg Langs Closed Under Intersection: Proof

Let $M_1 = (Q_1, \Sigma, \Delta_1, s_1, F_1)$ be an NFA for L_1
Let $M_2 = (Q_2, \Sigma, \Delta_2, s_2, F_2)$ be an NFA for L_2
From M_1 and M_2 construct an NFA M for $L_1 \cap L_2$.

Reg Langs Closed Under Intersection: Proof

Let $M_1 = (Q_1, \Sigma, \Delta_1, s_1, F_1)$ be an NFA for L_1
Let $M_2 = (Q_2, \Sigma, \Delta_2, s_2, F_2)$ be an NFA for L_2
From M_1 and M_2 construct an NFA M for $L_1 \cap L_2$.

$M = (Q_1 \times Q_2, \Sigma, \Delta, (s_1, s_2), F_1 \times F_2)$ where

Reg Langs Closed Under Intersection: Proof

Let $M_1 = (Q_1, \Sigma, \Delta_1, s_1, F_1)$ be an NFA for L_1
Let $M_2 = (Q_2, \Sigma, \Delta_2, s_2, F_2)$ be an NFA for L_2
From M_1 and M_2 construct an NFA M for $L_1 \cap L_2$.

$M = (Q_1 \times Q_2, \Sigma, \Delta, (s_1, s_2), F_1 \times F_2)$ where

$\Delta((q_1, q_2), \sigma) =$

Reg Langs Closed Under Intersection: Proof

Let $M_1 = (Q_1, \Sigma, \Delta_1, s_1, F_1)$ be an NFA for L_1
Let $M_2 = (Q_2, \Sigma, \Delta_2, s_2, F_2)$ be an NFA for L_2
From M_1 and M_2 construct an NFA M for $L_1 \cap L_2$.

$M = (Q_1 \times Q_2, \Sigma, \Delta, (s_1, s_2), F_1 \times F_2)$ where

$\Delta((q_1, q_2), \sigma) =$

$\{(p_1, p_2) : p_1 \in \Delta_1(q_1, \sigma) \wedge p_2 \in \Delta_2(q_2, \sigma)\}$

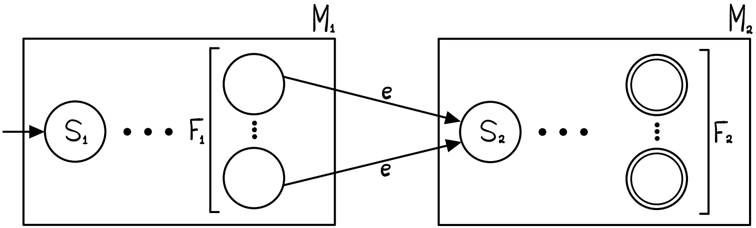
Reg Langs Closed Under Concat-Intuitively

Have an e-transition from final state of M_1 to start state of M_2.

Reg Langs Closed Under Concat-Intuitively

Have an e-transition from final state of M_1 to start state of M_2.
Generic picture on next slide.

Reg Langs Closed Under Concat-Picture



Reg Langs Closed Under Concat-Formally

Formally If L_1 is reg via NFA

$$(Q_1, \Sigma, \Delta_1, s_1, F_1).\text{ We will take } |Q_1| = n_1 \ .$$

Reg Langs Closed Under Concat-Formally

Formally If L_1 is reg via NFA

$$(Q_1, \Sigma, \Delta_1, s_1, F_1). \text{ We will take } |Q_1| = n_1 \ .$$

and L_2 is reg via NFA

$$(Q_2, \Sigma, \Delta_2, s_2, F_2). \text{ We will take } |Q_2| = n_2 \ .$$

Reg Langs Closed Under Concat-Formally

Formally If L_1 is reg via NFA

$$(Q_1, \Sigma, \Delta_1, s_1, F_1). \text{ We will take } |Q_1| = n_1 \ .$$

and L_2 is reg via NFA

$$(Q_2, \Sigma, \Delta_2, s_2, F_2). \text{ We will take } |Q_2| = n_2 \ .$$

then $L_1 L_2$ is reg via NFA

$$(Q_1 \cup Q_2, \Sigma, \Delta', s_1, F_2)$$

Reg Langs Closed Under Concat-Formally

Formally If L_1 is reg via NFA

$$(Q_1, \Sigma, \Delta_1, s_1, F_1). \text{ We will take } |Q_1| = n_1 \ .$$

and L_2 is reg via NFA

$$(Q_2, \Sigma, \Delta_2, s_2, F_2). \text{ We will take } |Q_2| = n_2 \ .$$

then L_1L_2 is reg via NFA

$$(Q_1 \cup Q_2, \Sigma, \Delta', s_1, F_2)$$

If $q \in Q_1 - F_1$, $\sigma \in \Sigma \cup \{e\}$ then $\Delta'(q, \sigma) = \Delta_1(q, \sigma)$.

Reg Langs Closed Under Concat-Formally

Formally If L_1 is reg via NFA

$$(Q_1, \Sigma, \Delta_1, s_1, F_1). \text{ We will take } |Q_1| = n_1 \ .$$

and L_2 is reg via NFA

$$(Q_2, \Sigma, \Delta_2, s_2, F_2). \text{ We will take } |Q_2| = n_2 \ .$$

then $L_1 L_2$ is reg via NFA

$$(Q_1 \cup Q_2, \Sigma, \Delta', s_1, F_2)$$

If $q \in Q_1 - F_1$, $\sigma \in \Sigma \cup \{e\}$ then $\Delta'(q, \sigma) = \Delta_1(q, \sigma)$.
If $q \in F_1$, $\sigma \in \Sigma$ then $\Delta'(q, \sigma) = \Delta_1(q, \sigma)$.

Reg Langs Closed Under Concat-Formally

Formally If L_1 is reg via NFA

$$(Q_1, \Sigma, \Delta_1, s_1, F_1). \text{ We will take } |Q_1| = n_1 \ .$$

and L_2 is reg via NFA

$$(Q_2, \Sigma, \Delta_2, s_2, F_2). \text{ We will take } |Q_2| = n_2 \ .$$

then $L_1 L_2$ is reg via NFA

$$(Q_1 \cup Q_2, \Sigma, \Delta', s_1, F_2)$$

If $q \in Q_1 - F_1$, $\sigma \in \Sigma \cup \{e\}$ then $\Delta'(q, \sigma) = \Delta_1(q, \sigma)$.
If $q \in F_1$, $\sigma \in \Sigma$ then $\Delta'(q, \sigma) = \Delta_1(q, \sigma)$.
If $q \in F_1$, $\Delta'(q, e) = \Delta_1(q, e) \cup \{s_2\}$.

Reg Langs Closed Under Concat-Formally

Formally If L_1 is reg via NFA

$$(Q_1, \Sigma, \Delta_1, s_1, F_1). \text{ We will take } |Q_1| = n_1 \ .$$

and L_2 is reg via NFA

$$(Q_2, \Sigma, \Delta_2, s_2, F_2). \text{ We will take } |Q_2| = n_2 \ .$$

then $L_1 L_2$ is reg via NFA

$$(Q_1 \cup Q_2, \Sigma, \Delta', s_1, F_2)$$

If $q \in Q_1 - F_1$, $\sigma \in \Sigma \cup \{e\}$ then $\Delta'(q, \sigma) = \Delta_1(q, \sigma)$.
If $q \in F_1$, $\sigma \in \Sigma$ then $\Delta'(q, \sigma) = \Delta_1(q, \sigma)$.
If $q \in F_1$, $\Delta'(q, e) = \Delta_1(q, e) \cup \{s_2\}$.
If $q \in Q_2$, $\sigma \in \Sigma \cup \{e\}$ then $\Delta'(q, \sigma) = \Delta_2(q, \sigma)$.

Reg Langs Closed Under Concat-Formally

Formally If L_1 is reg via NFA

$$(Q_1, \Sigma, \Delta_1, s_1, F_1). \text{ We will take } |Q_1| = n_1 \ .$$

and L_2 is reg via NFA

$$(Q_2, \Sigma, \Delta_2, s_2, F_2). \text{ We will take } |Q_2| = n_2 \ .$$

then $L_1 L_2$ is reg via NFA

$$(Q_1 \cup Q_2, \Sigma, \Delta', s_1, F_2)$$

If $q \in Q_1 - F_1$, $\sigma \in \Sigma \cup \{e\}$ then $\Delta'(q, \sigma) = \Delta_1(q, \sigma)$.

If $q \in F_1$, $\sigma \in \Sigma$ then $\Delta'(q, \sigma) = \Delta_1(q, \sigma)$.

If $q \in F_1$, $\Delta'(q, e) = \Delta_1(q, e) \cup \{s_2\}$.

If $q \in Q_2$, $\sigma \in \Sigma \cup \{e\}$ then $\Delta'(q, \sigma) = \Delta_2(q, \sigma)$.

Number of states: $n_1 + n_2$.

Reg Langs Closed Under $*$?-Intuition-1st Try

Have an e-transition from final states of M to start state of M.

Reg Langs Closed Under $*$?-Intuition-1st Try

Have an e-transition from final states of M to start state of M.

Next slide has a generic picture of this approach.

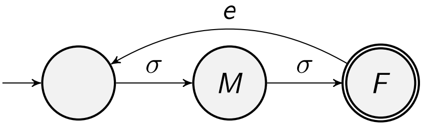
Reg Langs Closed Under $*$?-Intuition-1st Try

Have an e-transition from final states of M to start state of M.

Next slide has a generic picture of this approach.

Spoiler Alert This will not work.

Reg Langs Closed Under ∗?-Picture-1st Try



What Goes Wrong with 1st Try?

What goes wrong?

What Goes Wrong with 1st Try?

What goes wrong?

We want e to be accepted.

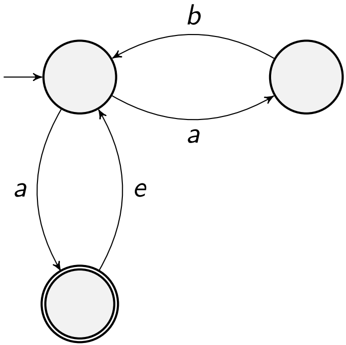
What Goes Wrong with 1st Try?

What goes wrong?

We want e to be accepted.

Next slide has an NFA where this does not work.

What Goes Wrong with 1st Try?-Picture



Reg Langs Closed Under $*$?-Intuition-2nd Try

Have an e-transition from final states of M to start state of M AND make s a final state.

Reg Langs Closed Under $*$?-Intuition-2nd Try

Have an e-transition from final states of M to start state of M AND make s a final state.

Next slide has a generic picture of this approach.

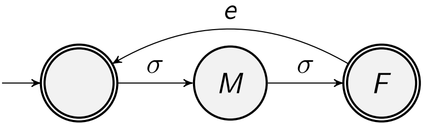
Reg Langs Closed Under $*$?-Intuition-2nd Try

Have an e-transition from final states of M to start state of M AND make s a final state.

Next slide has a generic picture of this approach.

Spoiler Alert This will not work.

Reg Langs Closed Under $*$?-Picture-2nd Try



What Goes Wrong with 2nd Try

What goes wrong?

What Goes Wrong with 2nd Try

What goes wrong?

Might accept too much.

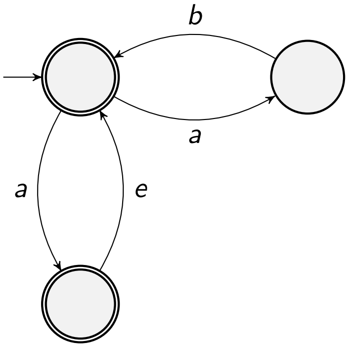
What Goes Wrong with 2nd Try

What goes wrong?

Might accept too much.

Next slide has an NFA where this does not work.

What Goes Wrong with 2nd Try-Picture



Reg Langs Closed Under $*$?-Intuition-3rd Try

Have an e-transition from final states of M to a NEW start state of M. That NEW start state is a final state and has an e-trans to old start state.

Reg Langs Closed Under $*$?-Intuition-3rd Try

Have an e-transition from final states of M to a NEW start state of M. That NEW start state is a final state and has an e-trans to old start state.

Next slide has a generic picture of this approach.

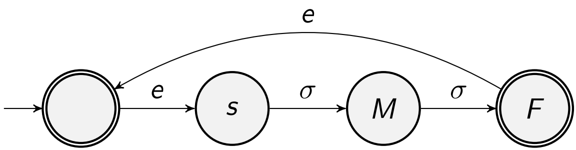
Reg Langs Closed Under $*$?-Intuition-3rd Try

Have an e-transition from final states of M to a NEW start state of M. That NEW start state is a final state and has an e-trans to old start state.

Next slide has a generic picture of this approach.

Spoiler Alert This will work.

Reg Langs Closed Under ∗?-Picture-3rd Try



Reg Langs Closed Under $*$?-Formally

Might be a HW or exam question.

Summary of Closure Properties and Proofs

X means **can't prove easily**
$n_1 + n_2$ (and similar) is number of states in new machine if L_i reg via n_i-state machine.

Closure Property	DFA	NFA
$L_1 \cup L_2$	$n_1 n_2$	$n_1 + n_2 + 1$
$L_1 \cap L_2$	$n_1 n_2$	$n_1 n_2$
$L_1 \cdot L_2$	X	$n_1 + n_2$
$\overline{L}$	n	X
L^*	X	$n + 1$

BILL AND NATHAN STOP RECORDING LECTURE!!!!

BILL AND NATHAN STOP RECORDING LECTURE!!!