Nondeterministic Finite Automata

Nondeterminism

Subset Construction

Nondeterminism

- A *nondeterministic finite automaton* has the ability to be in several states at once.
- Transitions from a state on an input symbol can be to any set of states.

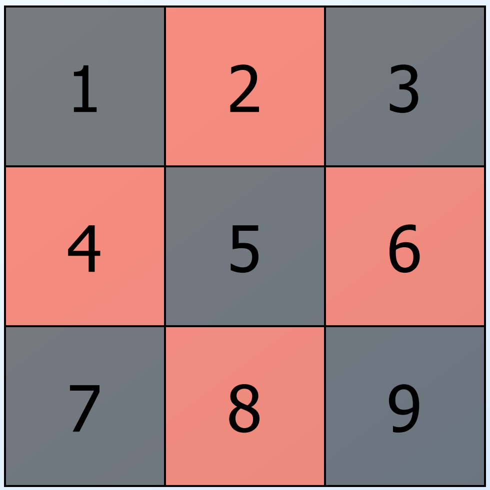
Nondeterminism – (2)

- Start in one start state.
- Accept if any sequence of choices leads to a final state.
- Intuitively: the NFA always "guesses right."

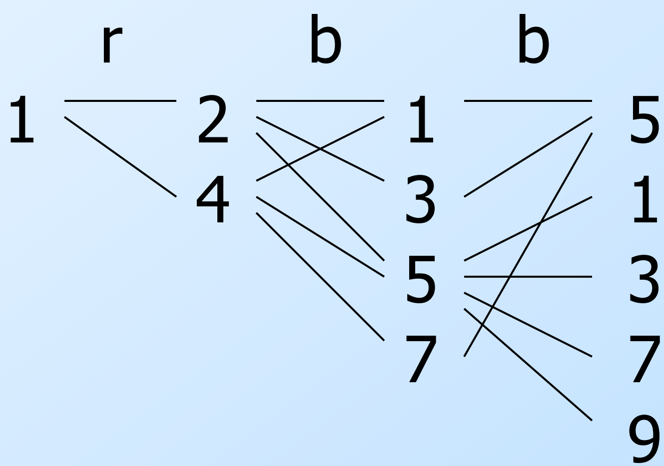
Example: Moves on a Chessboard

- States = squares.
- Inputs = r (move to an adjacent red square) and b (move to an adjacent black square).
- Start state, final state are in opposite corners.

Example: Chessboard – (2)



	r	b
→ 1	2,4	5
2	4,6	1,3,5
3	2,6	5
4	2,8	1,5,7
5	2,4,6,8	1,3,7,9
6	2,8	3,5,9
7	4,8	5
8	4,6	5,7,9
* 9	6,8	5



← Accept, since final state reached

Formal NFA

- A finite set of states, typically Q.
- An input alphabet, typically Σ.
- A transition function, typically δ.
- A start state in Q, typically q_0.
- A set of final states $F \subseteq Q$.

Transition Function of an NFA

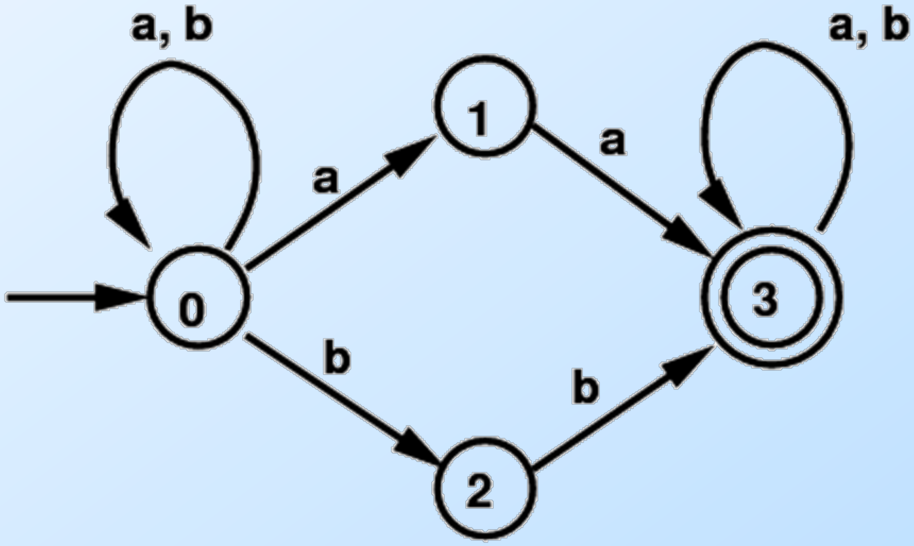
- $\delta(q, a)$ is a set of states.
- Extend to strings as follows:
- Basis: $\delta(q, \epsilon) = \{q\}$
- Induction: $\delta(q, wa)$ = the union over all states p in $\delta(q, w)$ of $\delta(p, a)$

Language of an NFA

- A string w is **accepted** by an NFA if $\delta(q_0, w)$ contains at least one final state.
- That is, **there exists** a sequence of valid transitions from q_0 to a final state given the input w.
- The language of the NFA is the set of strings it accepts.

Example NFA

- Set of all strings with two consecutive a's or two consecutive b's:



- Note that some states have an empty transition on an a or b, and some have multiple transitions on a or b.

Example 2: Language of an NFA

1	2	3
4	5	6
7	8	9

- For our chessboard NFA we saw that rbb is accepted.
- If the input consists of only b's, the set of accessible states alternates between {5} and {1,3,7,9}, so only even-length, nonempty strings of b's are accepted.
- What about strings with at least one r?

Equivalence of DFA's, NFA's

- A DFA can be turned into an NFA that accepts the same language.
- If $\delta_D(q, a) = p$, let the NFA have $\delta_N(q, a) = \{p\}$.
- Then the NFA is always in a set containing exactly one state – the state the DFA is in after reading the same input.

Equivalence – (2)

- Surprisingly, for any NFA there is a DFA that accepts the same language.
- Proof is the *subset construction*.
- The number of states of the DFA can be exponential in the number of states of the NFA.
- Thus, NFA's accept **exactly** the regular languages.

Subset Construction

- Given an NFA with states Q, inputs Σ, transition function δ_N, state state q_0, and final states F, construct equivalent DFA with:
  - States 2^Q (Set of subsets of Q).
  - Inputs Σ.
  - Start state $\{q_0\}$.
  - Final states = all those with a member of F.

Critical Point

- The DFA states have *names* that are sets of NFA states.
- But as a DFA state, an expression like {p,q} must be read as a single symbol, not as a set.
- Analogy: a class of objects whose values are sets of objects of another class.

Subset Construction – (2)

- The transition function δ_D is defined by:

$\delta_D(\{q_1,\ldots,q_k\}, a)$ is the union over all $i = 1,\ldots,k$ of $\delta_N(q_i, a)$.

- Example: We'll construct the DFA equivalent of our "chessboard" NFA.

Example: Subset Construction

		r	b
→	1	2,4	5
	2	4,6	1,3,5
	3	2,6	5
	4	2,8	1,5,7
	5	2,4,6,8	1,3,7,9
	6	2,8	3,5,9
	7	4,8	5
	8	4,6	5,7,9
*	9	6,8	5

		r	b
→	{1}	{2,4}	{5}
	{2,4}		
	{5}		

Alert: What we're doing here is the *lazy* form of DFA construction, where we only construct a state if we are forced to.

Example: Subset Construction

		r	b
→	1	2,4	5
	2	4,6	1,3,5
	3	2,6	5
	4	2,8	1,5,7
	5	2,4,6,8	1,3,7,9
	6	2,8	3,5,9
	7	4,8	5
	8	4,6	5,7,9
*	9	6,8	5

	r	b
→ {1}	{2,4}	{5}
{2,4}	{2,4,6,8}	{1,3,5,7}
{5}		
{2,4,6,8}		
{1,3,5,7}		

Example: Subset Construction

		r	b
→	1	2,4	5
	2	4,6	1,3,5
	3	2,6	5
	4	2,8	1,5,7
	5	2,4,6,8	1,3,7,9
	6	2,8	3,5,9
	7	4,8	5
	8	4,6	5,7,9
*	9	6,8	5

		r	b
→	{1}	{2,4}	{5}
	{2,4}	{2,4,6,8}	{1,3,5,7}
	{5}	{2,4,6,8}	{1,3,7,9}
	{2,4,6,8}		
	{1,3,5,7}		
*	{1,3,7,9}		

Example: Subset Construction

		r	b
→	1	2,4	5
	2	4,6	1,3,5
	3	2,6	5
	4	2,8	1,5,7
	5	2,4,6,8	1,3,7,9
	6	2,8	3,5,9
	7	4,8	5
	8	4,6	5,7,9
*	9	6,8	5

		r	b
→	{1}	{2,4}	{5}
	{2,4}	{2,4,6,8}	{1,3,5,7}
	{5}	{2,4,6,8}	{1,3,7,9}
	{2,4,6,8}	{2,4,6,8}	{1,3,5,7,9}
	{1,3,5,7}		
*	{1,3,7,9}		
*	{1,3,5,7,9}		

Example: Subset Construction

		r	b
→	1	2,4	5
	2	4,6	1,3,5
	3	2,6	5
	4	2,8	1,5,7
	5	2,4,6,8	1,3,7,9
	6	2,8	3,5,9
	7	4,8	5
	8	4,6	5,7,9
*	9	6,8	5

		r	b
→	{1}	{2,4}	{5}
	{2,4}	{2,4,6,8}	{1,3,5,7}
	{5}	{2,4,6,8}	{1,3,7,9}
	{2,4,6,8}	{2,4,6,8}	{1,3,5,7,9}
	{1,3,5,7}	{2,4,6,8}	{1,3,5,7,9}
*	{1,3,7,9}		
*	{1,3,5,7,9}		

Example: Subset Construction

		r	b
→	1	2,4	5
	2	4,6	1,3,5
	3	2,6	5
	4	2,8	1,5,7
	5	2,4,6,8	1,3,7,9
	6	2,8	3,5,9
	7	4,8	5
	8	4,6	5,7,9
*	9	6,8	5

		r	b
→	{1}	{2,4}	{5}
	{2,4}	{2,4,6,8}	{1,3,5,7}
	{5}	{2,4,6,8}	{1,3,7,9}
	{2,4,6,8}	{2,4,6,8}	{1,3,5,7,9}
	{1,3,5,7}	{2,4,6,8}	{1,3,5,7,9}
*	{1,3,7,9}	{2,4,6,8}	{5}
*	{1,3,5,7,9}		

Example: Subset Construction

		r	b
→	1	2,4	5
	2	4,6	1,3,5
	3	2,6	5
	4	2,8	1,5,7
	5	2,4,6,8	1,3,7,9
	6	2,8	3,5,9
	7	4,8	5
	8	4,6	5,7,9
*	9	6,8	5

		r	b
→	{1}	{2,4}	{5}
	{2,4}	{2,4,6,8}	{1,3,5,7}
	{5}	{2,4,6,8}	{1,3,7,9}
	{2,4,6,8}	{2,4,6,8}	{1,3,5,7,9}
	{1,3,5,7}	{2,4,6,8}	{1,3,5,7,9}
*	{1,3,7,9}	{2,4,6,8}	{5}
*	{1,3,5,7,9}	{2,4,6,8}	{1,3,5,7,9}

Proof of Equivalence: Subset Construction

- The proof is almost a pun.
- Show by induction on |w| that

$$\delta_N(q_0, w) = \delta_D(\{q_0\}, w)$$

- Basis: w = ϵ: $\delta_N(q_0, \epsilon) = \delta_D(\{q_0\}, \epsilon) = \{q_0\}$.

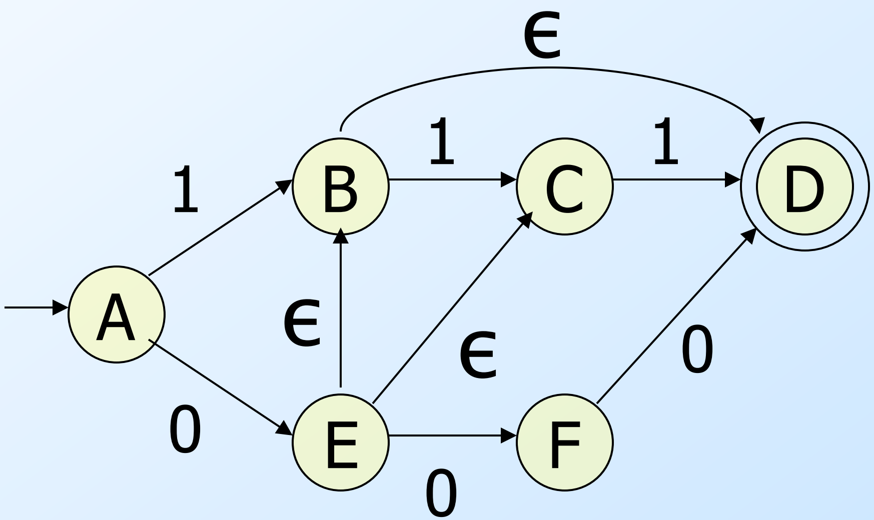
Induction

- Assume IH for strings shorter than w.
- Let w = xa; IH holds for x.
- Let $\delta_N(q_0, x) = \delta_D(\{q_0\}, x) = S$.
- Let T = the union over all states p in S of $\delta_N(p, a)$.
- Then $\delta_N(q_0, w) = \delta_D(\{q_0\}, w) = T$.
  - For NFA: the extension of δ_N.
  - For DFA: definition of δ_D plus extension of δ_D.
    - That is, $\delta_D(S, a) = T$; then extend δ_D to w = xa.

NFA's With ϵ-Transitions

- We can allow state-to-state transitions on ϵ input.
- These transitions are done spontaneously, without looking at the input string.
- A convenience at times, but still only regular languages are accepted.

Example: ϵ-NFA

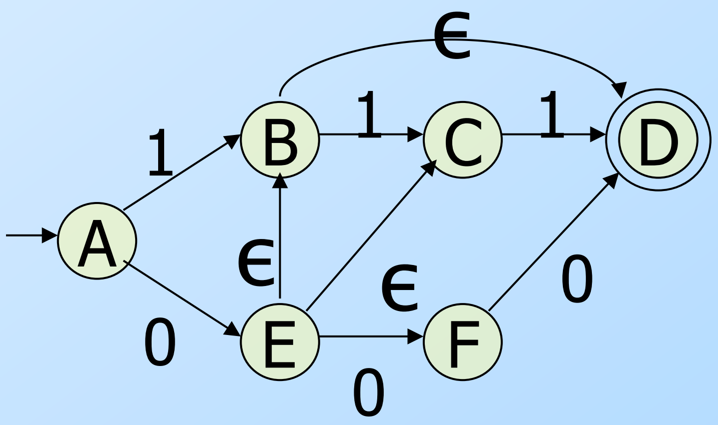


		0	1	ϵ
→	A	{E}	{B}	∅
	B	∅	{C}	{D}
	C	∅	{D}	∅
*	D	∅	∅	∅
	E	{F}	∅	{B, C}
	F	{D}	∅	∅

Closure of States

- CL(q) = set of states you can reach from state q following only arcs labeled ϵ.
- Example: CL(A) = {A};

  CL(E) = {B, C, D, E}.



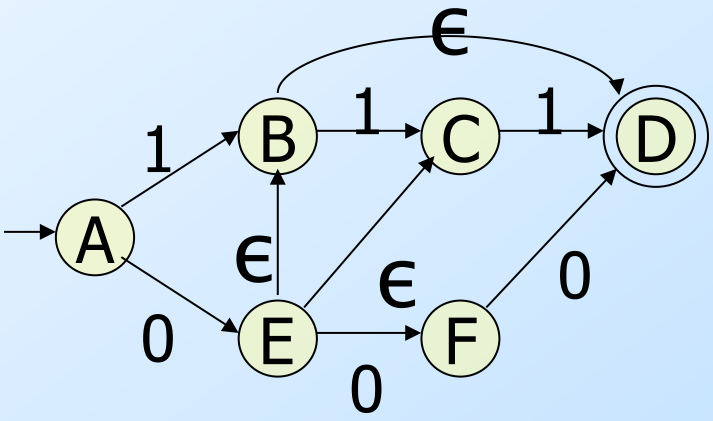
- Closure of a set of states = union of the closure of each state.

Extended Delta

- Basis: $\hat{\delta}(q, \epsilon) = CL(q)$.
- Induction: $\hat{\delta}(q, xa)$ is computed as follows:
  1. Start with $\hat{\delta}(q, x) = S$.
  2. Take the union of $CL(\delta(p, a))$ for all p in S.
- Intuition: $\hat{\delta}(q, w)$ is the set of states you can reach from q following a path labeled w.

And notice that $\delta(q, a)$ is *not* that set of states, for symbol a.

Example: Extended Delta



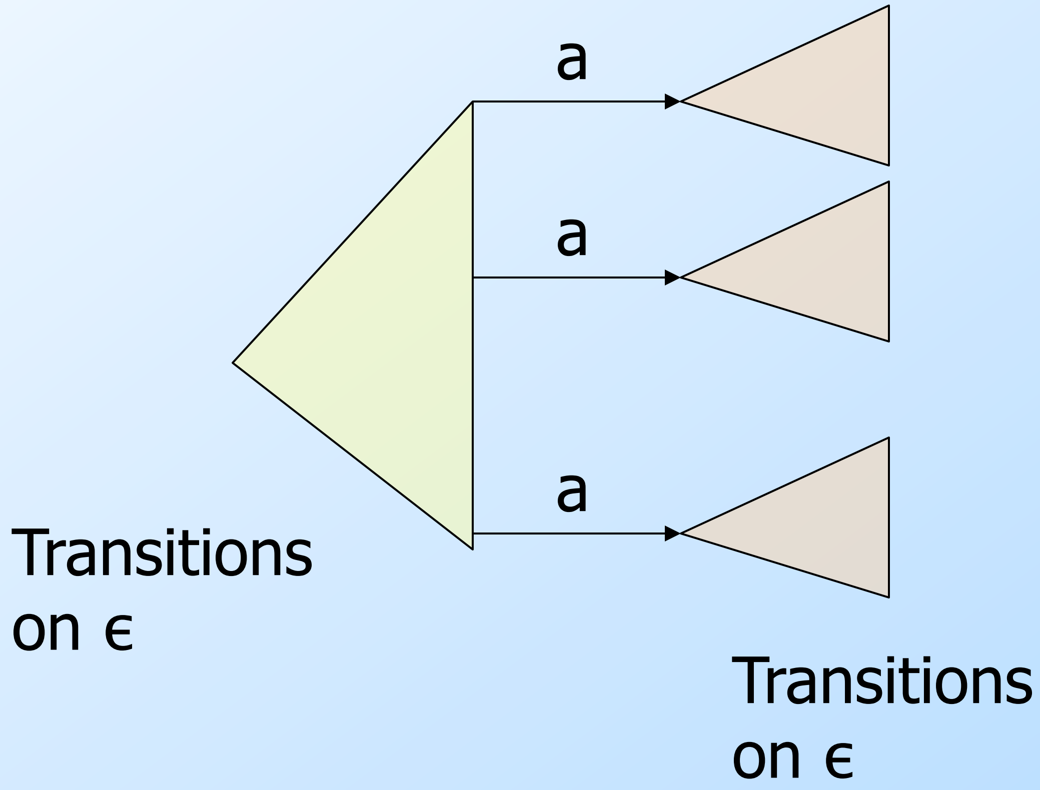
- $\hat{\delta}(A, \epsilon) = CL(A) = \{A\}$.
- $\hat{\delta}(A, 0) = CL(\{E\}) = \{B, C, D, E\}$.
- $\hat{\delta}(A, 01) = CL(\{C, D\}) = \{C, D\}$.
- *Language* of an ϵ-NFA is the set of strings w such that $\hat{\delta}(q_0, w)$ contains a final state.

Equivalence of NFA, ϵ-NFA

- Every NFA is an ϵ-NFA.
  - It just has no transitions on ϵ.
- Converse requires us to take an ϵ-NFA and construct an NFA that accepts the same language.
- We do so by combining ϵ–transitions with the next transition on a real input.

Warning: This treatment is a bit different from that in the text.

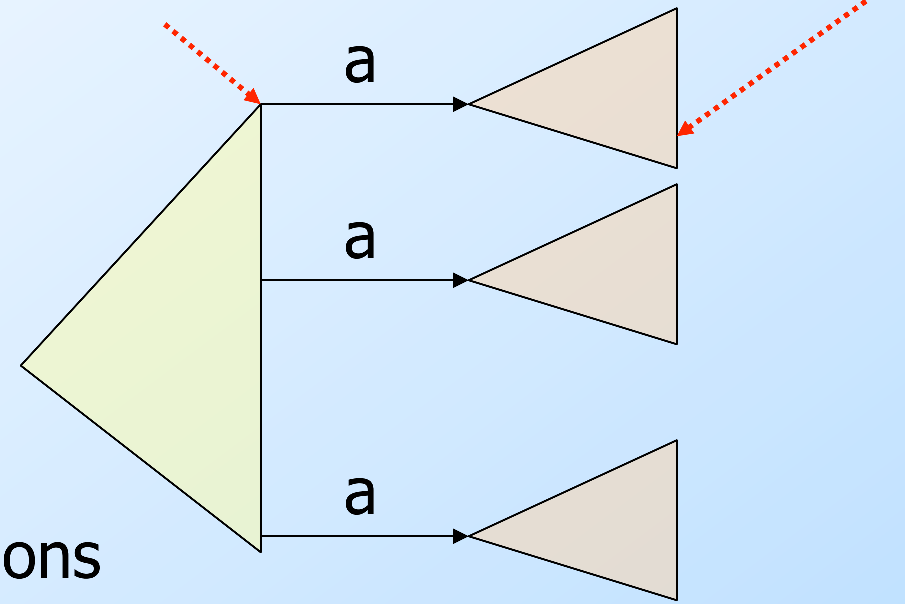
Picture of ϵ-Transition Removal



Picture of ϵ-Transition Removal

Text goes from here

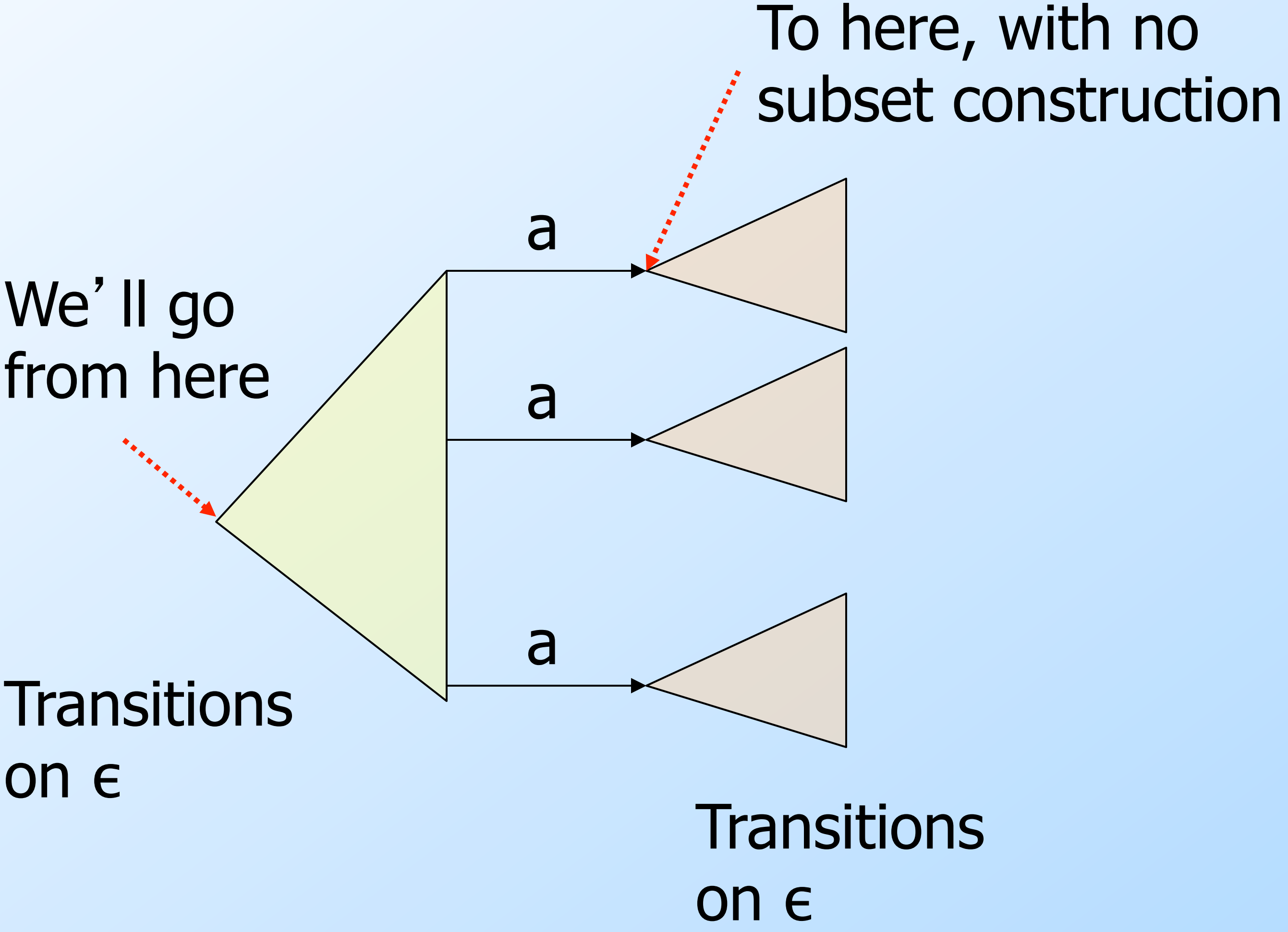
To here, and performs the subset construction



Transitions on ϵ

Transitions on ϵ

Picture of ϵ-Transition Removal



Equivalence – (2)

- Start with an ϵ-NFA with states Q, inputs Σ, start state q_0, final states F, and transition function δ_E.
- Construct an "ordinary" NFA with states Q, inputs Σ, start state q_0, final states F', and transition function δ_N.

Equivalence – (3)

- Compute $\delta_N(q, a)$ as follows:
  1. Let S = CL(q).
  2. $\delta_N(q, a)$ is the union over all p in S of $\delta_E(p, a)$.
- F' = the set of states q such that CL(q) contains a state of F.
- Intuition: δ_N incorporates ϵ–transitions before using *a* but not after.

Equivalence – (4)

- Prove by induction on |w| that

  $$CL(\delta_N(q_0, w)) = \hat{\delta}_E(q_0, w).$$

- Thus, the ϵ-NFA accepts w if and only if the "ordinary" NFA does.

Example: ϵ-NFA-to-NFA

Interesting closures: CL(B) = {B,D}; CL(E) = {B,C,D,E}

		0	1	ϵ
→	A	{E}	{B}	∅
	B	∅	{C}	{D}
	C	∅	{D}	∅
*	D	∅	∅	∅
	E	{F}	∅	{B, C}
	F	{D}	∅	∅

ϵ-NFA

		0	1
→	A	{E}	{B}
*	B	∅	{C}
	C	∅	{D}
*	D	∅	∅
*	E	{F}	{C, D}
	F	{D}	∅

Since closures of B and E include final state D.

Since closure of E includes B and C; which have transitions on 1 to C and D.

Summary

- DFA's, NFA's, and ϵ–NFA's all accept exactly the same set of languages: the regular languages.
- The NFA types are easier to design and may have exponentially fewer states than a DFA.
- But only a DFA can be implemented in linear time!