NFA/DFA Equivalence
CSCI 338
Definitions

DFAs consist of:
1. Finite set of states, $Q$.
2. Finite alphabet, $\Sigma$.
3. Transition function, $\delta: Q \times \Sigma \rightarrow Q$.
4. Start state, $q_0 \in Q$.
5. Set of accept states, $F \subseteq Q$.

NFAs consist of:
1. Finite set of states, $Q$.
2. Finite alphabet, $\Sigma$.
3. Transition function, $\delta: Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow \mathcal{P}(Q)$.
4. Start state, $q_0 \in Q$.
5. Set of accept states, $F \subseteq Q$. 
Definitions

DFAs consist of:
1. Finite set of states, \( Q \).
2. Finite alphabet, \( \Sigma \).
3. Transition function, \( \delta: Q \times \Sigma \to Q \).
4. Start state, \( q_0 \in Q \).
5. Set of accept states, \( F \subseteq Q \).

NFAs consist of:
1. Finite set of states, \( Q \).
2. Finite alphabet, \( \Sigma \).
3. Transition function, \( \delta: Q \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q) \).
4. Start state, \( q_0 \in Q \).
5. Set of accept states, \( F \subseteq Q \).
DFA vs NFA

NFA’s have a lot of shiny features, but do they actually get us any new capability?

How would we prove that NFA’s do provide new capability?
DFA vs NFA

NFA’s have a lot of shiny features, but do they actually get us any new capability?

How would we prove that NFA’s do provide new capability?
   Find some language that can be recognized by an NFA, but not a DFA.
DFA vs NFA

NFA’s have a lot of shiny features, but do they actually get us any new capability?

How would we prove that NFA’s do provide new capability? Find some language that can be recognized by an NFA, but not a DFA.

How would we prove that NFA’s do not provide new capability?
DFA vs NFA

NFA’s have a lot of shiny features, but do they actually get us any new capability?

How would we prove that NFA’s do provide new capability?
Find some language that can be recognized by an NFA, but not a DFA.

How would we prove that NFA’s do not provide new capability?
Show that every language recognized by an NFA can be recognized by a DFA.
DFA vs NFA

NFA’s have a lot of shiny features, but do they actually get us any new capability?

How would we prove that NFA’s do provide new capability?
Find some language that can be recognized by an NFA, but not a DFA.

How would we prove that NFA’s do not provide new capability?
Show that every language recognized by an NFA can be recognized by a DFA.
DFA vs NFA

Claim: Every NFA has an equivalent DFA.

Proof Approach:
For any NFA, turn it into a DFA.
DFA vs NFA

Claim: Every NFA has an equivalent DFA.

Proof Approach:
How did we keep track of our location in a DFA?

\[ \omega = 1110 \]
DFA vs NFA

Claim: Every NFA has an equivalent DFA.

Proof Approach:
How did we keep track of our location in an NFA?

\[
\omega = 101
\]
DFA vs NFA

Claim: Every NFA has an equivalent DFA.

Proof Approach:
How did we keep track of our location in an NFA?
Set of all states we could possibly be in.
Claim: Every NFA has an equivalent DFA.

Proof Approach:
How did we keep track of our location in an NFA?
Set of all states we could possibly be in.
DFA vs NFA

Claim: Every NFA has an equivalent DFA.

Proof Approach:
How did we keep track of our location in an NFA?
Set of all states we could possibly be in.

\[ \omega = 101 \]
DFA vs NFA

Claim: Every NFA has an equivalent DFA.

Proof Approach:
How did we keep track of our location in an NFA?
Set of all states we could possibly be in.

What is the set of all possible locations?

\[ \omega = 101 \]
DFA vs NFA

Claim: Every NFA has an equivalent DFA.

Proof Approach:

How did we keep track of our location in an NFA?
Set of all states we could possibly be in.

What is the set of all possible locations?
Power set! (set of all subsets)
DFA vs NFA

NFA

DFA
DFA vs NFA

NFA

DFA states = Power set of NFA states.
DFA vs NFA

NFA

DFA

DFA states = Power set of NFA states.
Start state = ?
DFA vs NFA

NFA

DFA states = Power set of NFA states.
Start state = NFA’s start state.
DFA vs NFA

NFA

DFA

DFA states = Power set of NFA states.
Start state = NFA’s start state.
Accept states = ?
DFA states = Power set of NFA states.
Start state = NFA’s start state.
Accept states = Any state that includes accept state from NFA.
Where should transition out of \( \{S_1\} \) with character 1 go?
DFA vs NFA

NFA

0, 1

\{S_1\}, \{S_2\}, \{S_1, S_2\}

DFA

0

\{S_3\}, \{S_1, S_3\}, \{S_2, S_3\}, \{S_1, S_2, S_3\}

Where should transition out of \{S_1\} with character 1 go?
Wherever \(S_1\) goes with 1 in the NFA.
Where should transition out of \{S_1\} with character 1 go?
Wherever \(S_1\) goes with 1 in the NFA.
Where should transition out of \( \{S_1\} \) with character 0 go?
Where should transition out of \( \{S_1\} \) with character 0 go?
If transition is not handled by NFA, send it to \( \emptyset \) (junk state).
Where should transition out of $\{S_2\}$ with character 0 go?
Where should transition out of \{S_2\} with character 0 go?

NFA could stay in \(S_2\) or go to \(S_3\), so \{\(S_2, S_3\}\}
Where should transition out of \{S_2\} with character 1 go?
Where should transition out of \( \{S_2\} \) with character 1 go?
Where should transition out of \{S_1, S_3\} with character 1 go?
DFA vs NFA

Where should transition out of \( \{S_1, S_3\} \) with character 1 go?
Where should transition out of \{S_1, S_3\} with character 0 go?
DFA vs NFA

Where should transition out of \{S_1, S_3\} with character 0 go?
DFA vs NFA

Rule?

DFA state transitions to DFA state consisting of all states it’s NFA states transition to.
Rule? For each DFA state \( R \) and \( e \in \Sigma \),
\[
\text{transition}(R, e) = \{ q \in \text{NFA}: q \in \text{transition}(r, e) \text{ for some } r \in R \}
\]
Rule? For each DFA state $R$ and $e \in \Sigma$,
transition($R, e$) = \{ $q \in$ NFA: $q \in$ transition($r, e$) for some $r \in R$ \}
transition($\{S_2\}, 0$) = $\{S_2, S_3\}$
Rule? For each DFA state $R$ and $e \in \Sigma$, 
\[
\text{transition}(R, e) = \{ q \in \text{NFA}: q \in \text{transition}(r, e) \text{ for some } r \in R \}
\]
\[
\text{transition}(\{S_1\}, 0) = \{ \}
\]
DFA vs NFA
**DFA vs NFA**

**NFA**

Diagram showing NFA states and transitions:
- State $s_1$ with transitions 1 to $s_3$, $\varepsilon$ to $s_1$, and 0 to $s_2$.
- State $s_2$ with transitions 0,1 to $s_3$.
- State $s_3$ with 0 to $s_2$.

**DFA**

Diagram showing DFA states:
- State $\emptyset$.
- State $\{s_1\}$.
- State $\{s_2\}$.
- State $\{s_1, s_2\}$.
- State $\{s_3\}$.
- State $\{s_1, s_3\}$.
- State $\{s_2, s_3\}$.
- State $\{s_1, s_2, s_3\}$.

What about $\varepsilon$-transitions?

Define extension of DFA state $R$:

$$E(R) = \{q \in \text{NFA}: q \text{ reachable from } r \in R \text{ with } \geq 0 \varepsilon\text{-transitions}\}$$
DFA vs NFA

**NFA**

- States: $S_1$, $S_2$, $S_3$
- Transitions:
  - $S_1$ to $S_2$: 1
  - $S_1$ to $S_3$: $\varepsilon$
  - $S_2$ to $S_3$: 0, 1

**DFA**

- States: $\emptyset$, $\{S_1\}$, $\{S_2\}$, $\{S_1, S_2\}$, $\{S_3\}$, $\{S_1, S_3\}$, $\{S_2, S_3\}$, $\{S_1, S_2, S_3\}$

$E(R) = \{q \in \text{NFA}: q \text{ reachable from } r \in R \text{ with } \geq 0 \varepsilon-\text{transitions}\}$

Example: $E(\{S_2, S_3\}) = ?$
DFA vs NFA

NFA

DFA

$E(R) = \{q \in \text{NFA}: q \text{ reachable from } r \in R \text{ with } \geq 0 \varepsilon\text{-transitions}\}$

Example: $E(\{S_2, S_3\}) = \{S_2, S_3\}$

$E(\{S_1, S_2\}) = \?$
DFA vs NFA

NFA

DFA

Example: $E(\{S_2, S_3\}) = \{S_2, S_3\}$

$E(\{S_1, S_2\}) = \{S_1, S_2, S_3\}$

$E(R) = \{q \in \text{NFA}: q \text{ reachable from } r \in R \text{ with } \geq 0 \text{ } \varepsilon\text{-transitions} \}$
DFA vs NFA

NFA

DFA

Make start state = ?
DFA vs NFA

NFA

DFA

Make start state = \( E(\{S_1\}) = \{S_1, S_3\} \)
DFA vs NFA

**NFA**

- **States:** $S = \{S_1, S_2, S_3\}$
- **Transitions:**
  - $S_1 \xrightarrow{0} S_2$
  - $S_1 \xrightarrow{1} S_3$
  - $S_1 \xrightarrow{\epsilon} S_3$
  - $S_2 \xrightarrow{0,1} S_3$

**DFA**

- **States:** $Q = \{\emptyset, \{S_1\}, \{S_2\}, \{S_1, S_2\}\}$
- **Transitions:**
  - $\emptyset \xrightarrow{0} \emptyset$
  - $\emptyset \xrightarrow{1} \emptyset$
  - $\emptyset \xrightarrow{\epsilon} \emptyset$
  - $\{S_1\} \xrightarrow{0,1} \{S_3\}$
  - $\{S_1\} \xrightarrow{\epsilon} \{S_3\}$
  - $\{S_2\} \xrightarrow{0,1} \{S_3\}$
  - $\{S_2\} \xrightarrow{\epsilon} \{S_3\}$
  - $\{S_1, S_2\} \xrightarrow{0,1} \{S_3\}$
  - $\{S_1, S_2\} \xrightarrow{\epsilon} \{S_3\}$

Make transitions:

$E(\text{transition}(r, e)) = \{q \in \text{NFA}: q \in \text{transition}(r, e) \text{ for some } r \in R\}$
DFA vs NFA

NFA

0

1

ε

s1

s2

s3

1

0

0,1

0

1

ε

s

s

s

0

{S1, S3}

{S2, S3}

{S1, S2}

{S1, S2, S3}

DFA

∅

{S1}

{S2}

{S1, S2}

Make transitions:

transition(R, e) = \{q \in \text{NFA}: q \in \text{transition}(r, e) \text{ for some } r \in R\}

E(\text{transition}(r, e))
DFA vs NFA

NFA

DFA

Make transitions:

\[ \text{transition}(R, e) = \{ q \in \text{NFA} : q \in \text{transition}(r, e) \text{ for some } r \in R \} \]
DFA vs NFA

NFA

\[
\begin{align*}
&s_1 \quad 1 \quad \varepsilon \quad 0 \\
&s_2 \quad 0,1 \\
&s_3
\end{align*}
\]

DFA

\[
\begin{align*}
&\emptyset \quad \{S_1\} \\
&\{S_3\} \quad \{S_1, S_3\} \\
&\{S_2\} \quad \{S_2, S_3\} \\
&\{S_1, S_2, S_3\}
\end{align*}
\]

Make transitions:

\[E(\text{transition}(r, e))\]

\[
\text{transition}(R, e) = \{q \in \text{NFA}: q \in \text{transition}(r, e) \text{ for some } r \in R\}
\]
DFA vs NFA

NFA

DFA

Make transitions:

\[
\text{transition}(R, e) = \{q \in \text{NFA}: q \in \text{transition}(r, e) \text{ for some } r \in R\}
\]
DFA vs NFA

**NFA**

- Finite set of states?
- Transition function for every state/character pair?
- Single start state?

**DFA**

- Finite Alphabet?
- Set of accept states?
DFA vs NFA

NFA

Finite set of states? Yes
Transition function for every state/character pair? Yes
Single start state? Yes

DFA

Finite set of states? Yes
Finite Alphabet? Yes
Transition function for every state/character pair? Yes
Single start state? Yes
Set of accept states? Yes
DFA vs NFA

NFA

Finite set of states? Yes
Transition function for every state/character pair? Yes
Single start state? Yes

DFA

Finite Alphabet? Yes
Set of accept states? Yes
DFA vs NFA

NFA

Finite set of states?  Yes
Finite Alphabet?  Yes
Transition function for every state/character pair?  Yes
Single start state?  No
Set of accept states?  {S_1, S_2, S_3}

DFA

Finite set of states?  Yes
Finite Alphabet?  Yes
Transition function for every state/character pair?  Yes
Single start state?  No
Set of accept states?  {S_1, S_2, S_3}
DFA vs NFA

NFA

DFA

Finite set of states? Yes  Finite Alphabet? Yes
Transition function for every state/character pair? Yes
Single start state? Yes  Set of accept states?
DFA vs NFA

**Finite set of states?**  Yes  |  **Finite Alphabet?**  Yes
**Transition function for every state/character pair?**  Yes  |  **Single start state?**  Yes
**Set of accept states?**  Yes
DFA vs NFA

**Finite set of states?**  Yes

**Finite Alphabet?**  Yes

**Transition function for every state/character pair?**  Yes

**Single start state?**  Yes

**Set of accept states?**  Yes

It’s a DFA
DFA vs NFA

NFA

DFA

Equivalent?
DFA vs NFA

Suppose $w$ accepted by NFA.
DFA vs NFA

Suppose $w$ accepted by NFA. At each step of its processing, DFA will be in state that corresponds to all possible NFA states.
DFA vs NFA

NFA

DFA

Equivalent?

Suppose $w$ accepted by NFA. At each step of its processing, DFA will be in state that corresponds to all possible NFA states. If NFA ends on accept state, corresponding DFA state will accept too.
Suppose $w$ accepted by DFA.
DFA vs NFA

NFA

Equivalent?

Suppose $w$ accepted by DFA. At each step of its processing, DFA will be in state that corresponds to all possible NFA states.
DFA vs NFA

Suppose \( w \) accepted by DFA. At each step of its processing, DFA will be in state that corresponds to all possible NFA states. If DFA ends on accept state, it includes an NFA accept state.
Suppose $w$ is accepted by DFA. At each step of its processing, DFA will be in state that corresponds to all possible NFA states. If DFA ends on accept state, it includes an NFA accept state. 

Anything that can be done with NFAs can be done with DFAs!
Definitions

A language is called a **regular language** if some DFA recognizes it.
A language is called a **regular language** if some DFA or **NFA** recognizes it.
Definitions

A language is called a regular language if some DFA or NFA recognizes it.

How do you prove a language is regular?
Make a DFA that recognizes it.
Definitions

A language is called a **regular language** if some DFA or **NFA** recognizes it.

How do you prove a language is regular?
Make a DFA or **NFA** that recognizes it.
Make an NFA with three states for: 
\{\omega: \omega \text{ has the form } 0^*1^*0^+.\}

Proof:

Additional string notation:

- $0^*$: Zero or more 0s (e.g. 0, 0000, $\varepsilon$)
- $0^+$: One or more 0s (e.g. 0, 0000)
Make an NFA with three states for:
\{\omega: \omega \text{ has the form } 0^*1^*0^+.\}

Proof:

Additional string notation:
0*: Zero or more 0s (e.g. 0, 0000, \varepsilon)
0+: One or more 0s (e.g. 0, 0000)
Make an NFA with three states for: $\{\omega : \omega \text{ has the form } 1^*(001^+)^*.\}$

e.g. 0010011111, 1100111001

Proof:

Additional string notation:
- $0^*$: Zero or more 0s (e.g. 0, 0000, $\varepsilon$)
- $0^+$: One or more 0s (e.g. 0, 0000)
Make an NFA with three states for:
\{ \omega: \omega \text{ has the form } 1^* (001^+)^*. \}

Proof:

Additional string notation:
- $0^*$: Zero or more 0s (e.g. $0, 0000, \varepsilon$)
- $0^+$: One or more 0s (e.g. $0, 0000$)