A non-DFA

missing transitions

\[ \begin{array}{c}
q_0 \quad \xrightarrow{0} \quad q_1 \quad \xrightarrow{0} \quad q_2 \\
\end{array} \]
A non-DFA

missing transitions

adding a sink state
A non-DFA

missing transitions

Adding a sink state, missing transitions
A non-DFA

missing transitions

adding a sink state, missing transitions, and self-loops
A chess board

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>b</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
<td>2, 4</td>
</tr>
<tr>
<td>2</td>
<td>1, 3, 5</td>
<td>4, 6</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>2, 4</td>
</tr>
<tr>
<td>4</td>
<td>1, 5, 7</td>
<td>2, 8</td>
</tr>
<tr>
<td>5</td>
<td>1, 3, 7, 9</td>
<td>2, 4, 6, 8</td>
</tr>
<tr>
<td>6</td>
<td>3, 5, 9</td>
<td>2, 8</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>4, 8</td>
</tr>
<tr>
<td>8</td>
<td>5, 7, 9</td>
<td>4, 6</td>
</tr>
<tr>
<td>9</td>
<td>5</td>
<td>6, 8</td>
</tr>
</tbody>
</table>

$s \to^b t$ iff square $s$ is close to the black square $t$

$s \to^r t$ iff square $s$ is close to the red square $t$
Question

Diagram:

- Nodes labeled 1 to 9
- Edges with labels 'r' and 'b'
- Paths connecting nodes

Diagram represents a network with specific connections and labels.
(I) All strings $b^{2n}$ for $n \geq 1$ are accepted
(II) No string $r^n$ for $n \geq 0$ is accepted

A. Both (I) and (II) are true
B. Only (I) is true
C. Only (II) is true
D. Both (I) and (II) are false
(I) All strings $b^{2n}$ for $n \geq 1$ are accepted

(II) No string $r^n$ for $n \geq 0$ is accepted
Non-deterministic finite automaton with $\tau$-moves

silent action $\tau \notin \Sigma$

$\Sigma_\tau = \Sigma \cup \{\tau\}$

NFA $N = (Q, \Sigma, \rightarrow, q_0, F)$
- $Q$ finite set of states
- $\Sigma$ finite alphabet
- $\rightarrow \subseteq Q \times \Sigma_\tau \times Q$ transition relation
- $q_0 \in Q$ initial state
- $F \subseteq Q$ set of final states
Non-deterministic finite automaton with $\tau$-moves

silent action $\tau \notin \Sigma$

$\Sigma_\tau = \Sigma \cup \{\tau\}$

NFA $N = (Q, \Sigma, \rightarrow, q_0, F)$

- $Q$ finite set of states
- $\Sigma$ finite alphabet
- $\rightarrow \subseteq Q \times \Sigma_\tau \times Q$ transition relation
- $q_0 \in Q$ initial state
- $F \subseteq Q$ set of final states

language accepted by NFA $N$

$\mathcal{L}(N) = \{ w \in \Sigma^* \mid \exists q \in F: (q_0, w) \vdash_N^*(q, \varepsilon) \}$
What is the language accepted by the following NFA?

A. \( \{ ab^n | n \geq 1 \} \)
B. \( \{ ab^n | n \geq 0 \} \)
C. \( \emptyset \)
D. \( \{ b^n | n \geq 0 \} \cup \{ ab^n | n \geq 0 \} \cup \{ aab^n | n \geq 0 \} \)
A DFA vs. an NFA

A DFA (Deterministic Finite Automaton) vs. an NFA (Non-deterministic Finite Automaton) demonstrates different behaviors in handling input sequences. The DFA has a single path for each input symbol, while the NFA can have multiple paths for the same symbol, allowing for more flexibility and sometimes simpler representations.

The diagram illustrates two automata: one deterministic (left) and one non-deterministic (right), showing transitions for symbols 'a' and 'b'. The DFA on the left shows a single path for each transition, whereas the NFA on the right allows for multiple paths, including a self-loop on 'a' and a transition labeled 'τ' (often representing the null string).

Understanding these differences is crucial in automata theory, impacting areas such as compiler design and formal language theory.
A DFA vs. an NFA
both automata accept

\[ \{ \, w_1 \cdots w_n \in \{a, b\}^* \mid n \geq 0 \land \forall i, 1 \leq i \leq n: w_i = ab \lor w_i = aba \, \} \]

which is also \( (ab(1 + a))^* \)
Two theorems

THEOREM

if $L = \mathcal{L}(D)$ for some DFA, then $L = \mathcal{L}(N)$ for some NFA

PROOF

say $D = (Q, \Sigma, \delta, q_0, F)$

put $N = (Q, \Sigma, \rightarrow_N, q_0, F)$ with $\delta(q, a) = q' \iff q \xrightarrow{a}_N q'$

then $(q, w) \vdash_D (q', w')$ iff $(q, w) \vdash_N (q', w')$

thus $\mathcal{L}(N) = \mathcal{L}(D)$
THEOREM

if $L = \mathcal{L}(N)$ for some NFA, then $L = \mathcal{L}(D)$ for some DFA
THEOREM

if $L = \mathcal{L}(N)$ for some NFA, then $L = \mathcal{L}(D)$ for some DFA
Two theorems (cont.)

THEOREM

if $L = \mathcal{L}(N)$ for some NFA, then $L = \mathcal{L}(D)$ for some DFA
Two theorems (cont.)

**Lemma** for every NFA $N$

$$(q, w) \vdash^*_N (q', w') \iff (q, wv) \vdash^*_N (q', w'v)$$
PROOF say \( N = (Q_N, \Sigma, \rightarrow_N, q^0_N, F_N) \)

\( \varepsilon \)-closure \( E(q) = \{ \bar{q} \in Q_N \mid (q, \varepsilon) \vdash^*_N (\bar{q}, \varepsilon) \} \) for \( q \in Q_N \)
Two theorems (cont.)

**PROOF** say \( N = (Q_N, \Sigma, \rightarrow_N, q^0_N, F_N) \)

\[ \varepsilon\text{-closure } E(q) = \{ \bar{q} \in Q_N \mid (q, \varepsilon) \vdash^*_{N} (\bar{q}, \varepsilon) \} \text{ for } q \in Q_N \]

put \( D = (Q_D, \Sigma, \delta_D, q^0_D, F_D) \) with
- set of states \( Q_D = \mathcal{P}(Q_N) \)
- transition function \( \delta_D(Q, a) = \bigcup \{ E(q) \mid q \in Q, \ q \xrightarrow{a} N \bar{q} \} \)
- initial state \( q^0_D = E(q^0_N) \)
- final states \( F_D = \{ Q \subseteq Q_N \mid Q \cap F_N \neq \emptyset \} \)
Two theorems (cont.)

**PROOF** say \( N = (Q_N, \Sigma, \rightarrow_N, q_0^N, F_N) \)

\( \varepsilon \)-closure \( E(q) = \{ \bar{q} \in Q_N \mid (q, \varepsilon) \vdash_N^* (\bar{q}, \varepsilon) \} \) for \( q \in Q_N \)

put \( D = (Q_D, \Sigma, \delta_D, q_0^D, F_D) \) with

- set of states \( Q_D = \mathcal{P}(Q_N) \)
- transition function \( \delta_D(Q, a) = \bigcup \{ E(\bar{q}) \mid q \in Q, \ q \xrightarrow{a} N \bar{q} \} \)
- initial state \( q_0^D = E(q_0^N) \)
- final states \( F_D = \{ Q \subseteq Q_N \mid Q \cap F_N \neq \emptyset \} \)

it holds that \( (q, w) \vdash_N^* (q', \varepsilon) \) iff

\[ \exists Q' \subseteq Q_N : (E(q), w) \vdash_D^* (Q', \varepsilon) \text{ and } q' \in Q' \]

then \( \mathcal{L}(N) = \mathcal{L}(D) \) follows
An NFA-to-DFA example
Another NFA-to-DFA example