Lexical Analysis:
DFA Minimization
Automating Scanner Construction

PREVIOUSLY
RE → NFA (Thompson’s construction)
• Build an NFA for each term
• Combine them with $\varepsilon$-moves

NFA → DFA (subset construction)
• Build the simulation

TODAY
DFA → Minimal DFA
• Hopcroft’s algorithm
DFA Minimization

Details of the algorithm

- Group states into maximal size sets, *optimistically*
- Iteratively subdivide those sets, as needed
- States that remain grouped together are equivalent
DFA Minimization

Remember DFA = (Q, Σ, δ, q₀, F)

Initial partition, P₀, has two sets: {D_F} and {D-D_F}

Splitting a set s (“partitioning a set by a”)

• Assume qᵢ and qⱼ ∈ s and δ(qᵢ, a) = qₓ and δ(qⱼ, a) = qᵧ
• If qₓ and qᵧ are not in the same set, then s must be split
  → qᵢ has transition on a, qⱼ does not ⇒ a splits s
• One state in the final DFA cannot have two transitions on a (otherwise we have an NFA!)
DFA Minimization (the algorithm)

\[ P \leftarrow \{ D_F, \{D-D_F\}\} \]

while (P is still changing)

\[ T \leftarrow \emptyset \]

for each set \( p \in P \)

\[ T \leftarrow T \cup \text{Split}(p) \]

\[ P \leftarrow T \]

\text{Split}(S)

for each \( \alpha \in \Sigma \)

\[ \text{if } \alpha \text{ splits } S \text{ into } s_1 \text{ and } s_2 \]

then return \{\(s_1, s_2\}\}

return \( S \)

This is a another fixed-point algorithm!
Key Idea: Splitting $S$ around $\alpha$

The algorithm partitions $S$ around $\alpha$

Original set $S$

$S$ has transitions on $\alpha$ to $R$, $Q$, & $T$
Key Idea: Splitting $S$ around $\alpha$

Original set $S$

$S_1$

$S_2$

$S_2$ is everything in $S - S_1$

Could we split $S_2$ further?

Yes, will do this in another iteration!
DFA Minimization

What about \( a ( b \mid c )^* \)?

First, the subset construction:

<table>
<thead>
<tr>
<th>NFA states</th>
<th>( a )</th>
<th>( b )</th>
<th>( c )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s_0 )</td>
<td>( q_0 )</td>
<td>( q_1, q_2, q_3, q_4, q_6, q_9 )</td>
<td>none</td>
</tr>
<tr>
<td>( s_1 )</td>
<td>( q_1, q_2, q_3, q_4, q_6, q_9 )</td>
<td>none</td>
<td>( q_5, q_8, q_9, q_3, q_4, q_6 )</td>
</tr>
<tr>
<td>( s_2 )</td>
<td>( q_5, q_8, q_9 )</td>
<td>none</td>
<td>( s_2 )</td>
</tr>
<tr>
<td>( s_3 )</td>
<td>( q_7, q_8, q_9 )</td>
<td>none</td>
<td>( s_2 )</td>
</tr>
</tbody>
</table>

Final states
Apply DFA Minimization algorithm

\[
P \leftarrow \{ D_F, \{D-D_F}\} \\
while (P \text{ is still changing}) \\
    T \leftarrow \emptyset \\
    \text{for each set } p \in P \\
    T \leftarrow T \cup \text{Split}(p) \\
    P \leftarrow T
\]

\text{Split}(S) \\
\text{for each } \alpha \in \Sigma \\
\quad \text{if } \alpha \text{ splits } S \text{ into } s_1 \text{ and } s_2 \\
\quad \text{then return } \{s_1, s_2\} \\
\text{return } S
DFA Minimization

Then, apply the minimization algorithm

<table>
<thead>
<tr>
<th>Current Partition</th>
<th>Split on</th>
</tr>
</thead>
<tbody>
<tr>
<td>{s_1, s_2, s_3} {s_0}</td>
<td>a</td>
</tr>
<tr>
<td>{s_1, s_2, s_3} {s_0}</td>
<td>none</td>
</tr>
</tbody>
</table>

To produce the minimal DFA

In a previous lecture, we observed that a human would design a simpler automaton than Thompson’s construction & the subset construction did.

Minimizing that DFA produces the one that a human would design!
Abbreviated Register Specification

Start with a regular expression

\( r_0 \mid r_1 \mid r_2 \mid r_3 \mid r_4 \mid r_5 \mid r_6 \mid r_7 \mid r_8 \mid r_9 \)

The Cycle of Constructions

\[ \text{RE} \rightarrow \text{NFA} \rightarrow \text{DFA} \rightarrow \text{min-DFA} \]
Abbreviated Register Specification

Thompson's construction produces

The Cycle of Constructions

To make it fit, we've eliminated the $\varepsilon$-transition between “r” and “0...9”.
Abbreviated Register Specification

Thompson’s construction produces

\[ \varepsilon\text{-closure}(n_0) \]

The Cycle of Constructions

To make it fit, we’ve eliminated the \( \varepsilon \)-transition between “r” and “0...9”.

\( S_0 \)
Abbreviated Register Specification

Thompson’s construction produces

\[ \varepsilon\text{-closure}(\Delta(s_0, r)) \]

The Cycle of Constructions

To make it fit, we’ve eliminated the \( \varepsilon \)-transition between “r” and “0…9”.

\[
\begin{align*}
S_0 & \xrightarrow{\varepsilon} S_1 \\
S_1 & \xrightarrow{\varepsilon} S_f
\end{align*}
\]
Abbreviated Register Specification

Thompson’s construction produces

\[ r_0 r_1 r_2 \ldots r_9 \]

The Cycle of Constructions

To make it fit, we’ve eliminated the \( \varepsilon \)-transition between “r” and “0...9”.

\[ \varepsilon\text{-closure}(\Delta(s_1,0)) \]
Abbreviated Register Specification

Thompson's construction produces

\[ r_0 \quad r_1 \quad r_2 \quad r_8 \quad r_9 \ldots \]

\[ s_0 \quad s_f \]

To make it fit, we've eliminated the \( \epsilon \)-transition between “r” and “0...9”.

The Cycle of Constructions
Abbreviated Register Specification

Thompson's construction produces

\[ r_0 r_1 r_2 r_8 r_9 \ldots \]

To make it fit, we've eliminated the \( \varepsilon \) transition between "r" and "0...9".

The Cycle of Constructions
Abbreviated Register Specification

Thompson’s construction produces

\[ r_0 r_1 r_2 \ldots r_9 \]

To make it fit, we’ve eliminated the \( \varepsilon \)-transition between “r” and “0...9”.

\[ \varepsilon \text{-closure}(\Delta(s_1,8)) \]

The Cycle of Constructions

\[ S_0 \rightarrow S_1 \rightarrow S_f \]

RE → NFA → DFA → minimal DFA
Abbreviated Register Specification

Thompson’s construction produces

\[ r_0 r_1 r_2 r_8 r_9 \ldots \]

\[ s_0 s_f \]

\[ \varepsilon\text{-closure}(\Delta(s_1,9)) \]

The Cycle of Constructions

To make it fit, we’ve eliminated the \( \varepsilon \)-transition between “r” and “0...9”.

RE \( \rightarrow \) NFA \( \rightarrow \) DFA \( \rightarrow \) minimal DFA
Abbreviated Register Specification

The subset construction builds

This is a DFA, but it has a lot of states ...

The Cycle of Constructions

RE → NFA → DFA → minimal DFA
Abbreviated Register Specification

The subset construction builds

\[ P_0 = \{s_{f0}, s_{f1}, s_{f2}, \ldots, s_{f8}, s_{f9}, s0\} \]

The Cycle of Constructions

[Diagram showing the cycle of constructions: RE -> NFA -> DFA -> minimal DFA]
Abbreviated Register Specification

The DFA minimization algorithm builds

This looks like what a skilled compiler writer would do!

The Cycle of Constructions