Modeling Computation

Introduction to Formal Languages and Automata

Regular Expressions and Automata
Any regular expression $\alpha$ can be converted to a finite automaton $A$ such that $L(\alpha) = L(A)$.

n.b. Regular expressions are built from ground elements $a \in \Sigma, \varepsilon, \emptyset$ to which operations $\cup, \text{concat}, \emptyset$ are applied.
Example:
Convert $ab^*a$ to a finite automaton
Algorithm: RegEx to NFA

1. Convert every ground singleton $a \in \Sigma$ or $\varepsilon$ or $\emptyset$ (if any) to an automaton accepting just this singleton.

{\{a\}} becomes

{\{\varepsilon\}} becomes

$\emptyset$ becomes

\((\alpha\beta)\) becomes

![Diagram](image-url)
2. Apply concatenations, **unions**, and Kleene stars.

\((\alpha \cup \beta)\) becomes
2. Apply concatenations, unions, and **Kleene stars**.

$\alpha^*$ becomes
Exercise: convert $a(a \cup b)^* b$ to an NFA accepting the specified language.

form $(a\cup b)$
form $(a\cup b)$
form $a(a\cup b)$
form $a(a\cup b)^* b$
Any finite automaton $A$ can be converted to a regular expression $\alpha$ such that $L(A) = L(\alpha)$.

Convert finite state diagram to expression diagram with a single edge from the initial state to the favorable state.
Assumptions

1. $A$ has a single favorable state
   • If not, make a new favorable state and use $\varepsilon$-jumps to connect old favorable states to new one

2. $A$’s initial state has no incoming edges
   • If not, make a new initial state and use an $\varepsilon$-jump to connect it to the old initial state

3. $A$’s favorable state has no outgoing edges
   • If not, make a new favorable state and use an $\varepsilon$-jump to connect the old favorable state to it

4. Nodes are labeled 1 (initial state) to $n$ (favorable state)
   • If not, renumber the nodes
Algorithm: NFA to RegEx

Let $l_{j,k}^i$ denote the label of the $i$-th edge from node $j$ to node $k$. $l_{j,k}$ is used when there is only 1 edge between the nodes.

Whenever there are multiple edges from node $j$ to node $k$ with labels $l_{j,k}^1, l_{j,k}^2, \ldots, l_{j,k}^m$

Replace them with a single edge with label $l_{j,k}^1 \cup l_{j,k}^2 \cup \ldots \cup l_{j,k}^m$
for $i = 2$ to $n - 1$

for each pair of nodes $j, k$ such that there is an edge from $j$ to $i$
and an edge from $i$ to $k$,

if there is no edge from $i$ to $i$
then add an edge from $j$ to $k$ with label $(l_{j, i}, l_{i, k})$
else add an edge from $j$ to $k$ with label $(l_{j, i}, l_{i, k}^*, l_{i, k})$
end

combine multiple edges

end

remove from the diagram node $i$ and all edges connected to $i$

end
Exercise: convert the automaton to a regular expression representing the language accepted by the automaton.
\( a^*b(a^*b)^* = a(aUb)^*b \)

not easily simplified

Verify equivalence by comparing the DFAs