Lexical Analysis

DFA Minimization &
Equivalence to Regular Expressions

Copyright 2016, Pedro C. Diniz, all rights reserved.
Students enrolled in the Compilers class at the University of Southern California have explicit permission to make copies of these materials for their personal use.
DFA State Minimization

- How to Reduce the Number of States of a DFA?
  - Find unique minimum-state DFA (up to state names)
  - Need to recognize the same language

- Normalization
  - Assume every state has a transition on every symbol
  - If not, just add missing transitions to a dead state

- Key Idea
  - Find string $w$ that distinguishes states $s$ and $t$

- Algorithm
  - Start with accepting vs. non-accepting states partition of states
  - Refine state groups on all input sequences, i.e. by tracing all transitions
  - Until no refinement is possible
DFA State Minimization

• Algorithm
  – Start with accepting vs. non-accepting states partition of states
  – Refine state groups on all input sequences, i.e. by tracing all transitions
  – Until no refinement is possible

• Does this Terminate?
  – Refinement will end; in the limit 1 partition is 1 state

• What to do When Refinement Terminates?
  – Elect representative state for each partition
  – Merge edges
  – Remove unneeded states in each partition
Minimization Example

\[
\begin{array}{cccc}
\text{a} & \text{b} & \text{c} & \text{d} \\
0 & 0 & 0 & 0 \\
1 & 0 & 1 & 1 \\
\end{array}
\]

\[
\begin{array}{cccc}
\text{c} & \text{d} & \text{e} \\
0 & 1 & 0,1 \\
1 & 1 & 0,1 \\
\end{array}
\]
Minimization Example

$P_1$

- a
- b
- c
- d

$P_2$

- e
Minimization Example

- Label 0 does not split any partition!
Minimization Example

• Label 1 splits $P_1$ and $P_2$ partitions!
Minimization Example

- Label 1 splits $P_1$ and $P_2$ partitions!
Minimization Example

P_3  P_1  P_2

\(a\)  \(b\)  \(e\)
\(c\)  \(d\)
Minimization Example

- Label 0 does not split any partition!
Minimization Example

- Label 1 does not splits any partition!
Minimization Example

• Elect Representative and Merge Edges
Minimization Example

- Elect Representative and Merge Edges
DFA State Minimization: Algorithm

\[ \text{DFA} = \{ D, \Sigma, d, s_0, D_F \} \]

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

while (P is still changing)

\[ T \leftarrow \emptyset \]

for each set \( p \in P \)

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

\[ P \leftarrow T \]

Split(S)

for each \( c \in \Sigma \)

if \( c \) splits \( S \) into \( s_1 \) and \( s_2 \)

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

return \( S \)
DFA to RE: Kleene Construction

• Path Problem over the DFA
  – Starting from state $s_1$ (numbering of states is 1 … N - important)
  – Label all edges through all states to an accepting state
  – What to do with cycles in the DFA, as they are infinite paths?

• Kleene Construction
  – Iterate and merge path expressions for every pair of nodes $i$ and $j$
    not going through any node with label higher then $k$
  – Increase $k$ up to $N$
  – In the end do the union of all path expressions that start at $s_1$ and
    end in a final state.
DFA to RE: Kleene Construction

for $i = 1$ to $N$

for $j = 1$ to $N$

$R^0_{ij} = \{ a | \delta(s_i, a) = s_j \}$

if $(i = j)$ then

$R^0_{ij} = R^0_{ij} \cup \{ \varepsilon \}$

for $k = 1$ to $N$

for $i = 1$ to $N$

for $j = 1$ to $N$

$R^k_{ij} = R^{k-1}_{ik} (R^{k-1}_{kk})^* R^{k-1}_{kj} \cup R^{k-1}_{ij}$

$L = \{ s_j \in S_F | R^N_{1j} \}$
for i = 1 to N
  for j = 1 to N
    $R^0_{ij} = \{ a \mid \delta(s_i, a) = s_j \}$
    if (i = j) then
      $R^0_{ij} = R^0_{ij} \cup \{ \varepsilon \}$
  for k = 1 to N
    for i = 1 to N
      for j = 1 to N
        $R^k_{ij} = R^{k-1}_{ik}(R^{k-1}_{kk})^* R^{k-1}_{kj} \cup R^{k-1}_{ij}$
  $L = \{ s_j \in S_F R^N_{1j} \}$

**Direct Path**
for i = 1 to N
   for j = 1 to N
      \( R^0_{ij} = \{ a \mid \delta(s_i, a) = s_j \} \)
      if (i = j) then
         \( R^0_{ij} = R^0_{ij} \sqcup \{ \varepsilon \} \)
   
for k = 1 to N
   for i = 1 to N
      for j = 1 to N
         \( R^k_{ij} = R^{k-1}_{ik} \cdot (R^{k-1}_{kk})^* \cdot R^{k-1}_{kj} \sqcup R^{k-1}_{ij} \)

L = \( s_j \in S_F \cdot R^N_{1j} \)
DFA to RE: Example

\[ R_{12}^0 = r \]
\[ R_{23}^0 = [0..9] \]
\[ R_{33}^0 = [0..9] \mid \varepsilon \]
DFA to RE: Example

\[ R_{12}^0 = r \]
\[ R_{23}^0 = [0..9] \]
\[ R_{33}^0 = [0..9] | \epsilon \]
\[ R_{kk}^0 = \text{nil otherwise} \]
DFA to RE: Example

\[ R_{12}^0 = r \]
\[ R_{23}^0 = \{0..9\} \]
\[ R_{33}^0 = \{0..9\} | \varepsilon \]
\[ R_{kk}^0 = \text{nil otherwise} \]

\[ R_{13}^1 = R_{11}^0 (R_{11}^0)^* R_{13}^0 | R_{13}^0 = \text{nil} \]
\[ R_{23}^1 = R_{21}^0 (R_{11}^0)^* R_{13}^0 | R_{23}^0 = \{0..9\} \]
\[ R_{33}^1 = R_{31}^0 (R_{11}^0)^* R_{13}^0 | R_{13}^0 = \{0..9\} | \varepsilon \]
\[ R_{13}^2 = R_{12}^1 (R_{22}^1)^* R_{23}^1 | R_{13}^1 = r \cdot \varepsilon^* \{0..9\} \]
\[ R_{33}^2 = R_{32}^1 (R_{22}^1)^* R_{23}^1 | R_{33}^1 = \{0..9\} | \varepsilon \]
\[ R_{33}^2 = R_{32}^1 (R_{22}^1)^* R_{23}^1 | R_{33}^1 = \{0..9\} | \varepsilon \]
DFA to RE: Example

\[ R^0_{12} = r \]
\[ R^0_{23} = [0..9] \]
\[ R^0_{33} = [0..9] \mid \varepsilon \]
\[ R^0_{kk} = \text{nil otherwise} \]

\[ R^3_{13} = R^2_{13} (R^2_{33})^* R^2_{33} \mid R^2_{13} \]
DFA to RE: Example

\[ R^0_{12} = r \]
\[ R^3_{13} = R^2_{13} (R^2_{33})^* \]
\[ R^0_{23} = [0..9] \]
\[ = (r \cdot \varepsilon^* [0..9])([0..9]*)((0..9]) \]
\[ R^0_{33} = [0..9] | \varepsilon \]
\[ = (r \cdot [0..9]+) | r \cdot [0..9] \]
\[ R^0_{kk} = \text{nil otherwise} \]
DFA to RE: Example

$L(M) = R^{3}_{13} = r \cdot [0..9]^+$
DFA, NFA and REs

- Kleene’s Construction
- DFA Minimization
- Thompson’s Construction
- Subset Construction
- Code for a scanner
- RE
- NFA
- DFA

Code for a scanner

DFA Minimization

Subset Construction

Thompson’s Construction

Kleene’s Construction
DFA, NFA and REs

Kleene’s Construction

DFA

Minimization

RE

Thompson’s Construction

NFA

Subset Construction

DFA

Code for a scanner

Regular Expressions and FA are Equivalent
Summary

• DFA Minimization
  – Find sequence $w$ that discriminates states
  – Iterate until no possible refinement

• DFA to RE
  – Kleene construction
  – Combine Path Expression for an increasingly large set of states

• DFA and RE are Equivalent
  – Given one you can derive an equivalent representation in the other