Are 2 DFAs Equivalent?

5 minute Talk
Tim Sheard
Scholarship Skills
Feb. 7, 2013
Are the two DFAs below equivalent (accept the same sets of strings) if the one on the left starts at state x, and the one on the right starts at the state u.

Red states are accepting (final) states, black states are non-accepting.
Traditional approach

• Minimize both
• Compare results for isomorphism
  – Quite expensive, proportional to the size of the powerset of states
New Approach - BiSimulation

- Every string describes a path through a DFA
- For all possible strings, study their paths through both machines.
- Show that each step in both paths pass through equivalent states.

- “babba”
- \[[x, y, z, z, z, z, z]\]
- \[u, w, v, w, v, w]\]
What is Equivalence

• Equivalent states, makes transitions on all symbols to equivalent states.
• Equivalent states are either both accepting, or both non-accepting

• If you try and equivate\(^1\) two states with different accepting status, the algorithm FAILS!

1. Is this a word? Place two items in the same equivalence class
Algorithm

• Start with a proposed equivalence \((x = u)\)

• If it has the same accepting status add it to the answer, otherwise FAIL

• Propose new equivalences based on possible steps on each character from the old pair

• Repeat
i = 1
equiv = []
todo = [(x,u)]

i = 2
equiv = [(x,u,1)]
todo = [(y,v),(y,w)]

i = 3
equiv = [(y,v,2),(x,u,1)]
todo = [(z,w),(z,w),(y,w)]
\[ i = 4 \]
\[ \text{equiv} = [(z, w, 3), (y, v, 2), (x, u, 1)] \]
\[ \text{todo} = [(z, v), (z, v), (z, w), (y, w)] \]

\[ i = 5 \]
\[ \text{equiv} = [(z, v, 4), (z, w, 3), (y, v, 2), (x, u, 1)] \]
\[ \text{todo} = [(z, w), (z, w), (z, v), (z, w), (y, w)] \]
bisimulate \((d@(DFA \text{ final next alphabet states}))\)

\[(x,y)\]

\[= \text{help } 1 \, [] \, [(x,y)]\]

where \(\text{add } (a,b) \, c = (\text{next } a \, c, \text{next } b \, c)\)

\[\text{help } i \, r \, [] = \text{return } (\text{Just } r)\]

\[\text{help } i \, r \, ((a,b): \, \text{todo }) =\]

\[\text{if elem } (a,b) \, (\text{map } \lambda \, (x,y,i) \rightarrow (x,y)) \, r)\]

\[\text{then help } i \, r \, \text{todo}\]

\[\text{else if final } a \neq \text{ final } b\]

\[\text{then return Nothing}\]

\[\text{else help } (i+1) \, ((a,b,i):r)\]

\[(\text{map } (\text{add } (a,b)) \, \text{alphabet } ++ \, \text{todo})\]
Conclusion

• Equivalence by Bisimulation
  – More efficient than minimization
    • Algorithm shown is $N^2$ in number of states
    • Can be made almost linear by storing equivalence classes rather than single equivalences.
  – Extends to NFAs as well as DFAs
  – Extends to Inclusion as well as Equivalence
  – Read about it
    • Checking NFA Equivalence with Bisimulations up to Congruence
    • Filippo Bonchi & Damien Pous
    • POPL 2013, Rome