LING/C SC/PSYC 438/538

Lecture 13
Sandiway Fong
Administrivia

• Homework 6 not yet graded...
• but we'll go over it anyway

• More on FSA and Regex...
Ungraded Homework Exercise Review

• Converting a NDFSA into a DFSA

Note: this machine with an \(\varepsilon\)-transition is non-deterministic

Note: this machine is deterministic

[Powerpoint animation]
Ungraded Homework Exercise Review

• Converting a NDFSA into a DFSA

Note: this machine with an $\epsilon$-transition is non-deterministic

Note: this machine is deterministic

[Powerpoint animation]
Question 1:

- Assume $\Sigma = \{a, b\}$. Draw a non-deterministic FSA that accepts all strings that contain the substring $abba$.
- Examples:
  - *abbba, babba, *abb, abba, *aaba, aabbbabba
Homework 6 Review

• Question 2: convert the machine into a deterministic FSA.

[Powerpoint animation]
Homework 6 Review

• Not guaranteed to be minimized
Homework 6 Review

• Question 3: Implement your machine in Perl

```perl
open($fh, $ARGV[0]) or die "ARGV[0] not found!\n"

$start = "{0}";
$end = "{0,a,abba}";
$delta = ("{0}",
   "{0,a}",
   "a","{0,a}","b","{0}"),
   "{0,ab}",
   "a","{0,a}","b","{0,ab}"},
   "{0,abb}",
   "a","{0,a,abba}","b","{0}"),
   
while ($line = <$fh>){
   $state = $start;
   # initial state
   chomp $line;
   # remove eol
   foreach $c (split //,$line) {
      $state = $delta{$state}{$c}
   }
   print "$line: ";
   if ($state eq $end) {
      print "accept\n"
   } else {
      print "reject\n"
   }
}
```

Converting FSA to REs

• Example:
  – Give a RE for the FSA:

  State by-pass method:
  1. Delete one state at a time
  2. Calculate the possible paths passing through the deleted state
  3. Add the regex calculated at each stage as an arc
    – e.g.
      • eliminate state 3
      • then 2...
Converting FSA to REs

- eliminate state 3
- eliminate state 2

Answer: \((0(1^+0|1)*1^+1 ~|~ 1)^*\)
Homework 6 Revisited

• Recall:

• Let's eliminate state 1:
  • 0 - aa*b -> 2
  • 2 – aa*b -> 2

[Powerpoint animation]
Homework 6 Revisited

• Let's eliminate state 2:
• $0 \rightarrow (aa \cdot b)(aa \cdot b) \cdot b \rightarrow 3$

[Powerpoint animation]
Homework 6 Revisited

• Let's eliminate state 3:
• $0 - aa*b(aa*b)*ba \rightarrow 4$
• $0 - aa*b(aa*b)*bb \rightarrow 0$

[Powerpoint animation]
Homework 6 Revisited

• Let's merge the state 0 loops:

[Powerpoint animation]
Homework 6 Revisited

• Make the whole machine vanish:
  – (b | aa*b(aa*b)*bb)*aa*b(aa*b)*ba(a | b)*
Homework 6 Revisited

• Implement it:
  • \(^{(b|aa*b(aa*b)*bb)*aa*b(aa*b)*ba(a|b)*}$

```perl
1 open($fh, $ARGV[0]) or die "$ARGV[0] not found!

2 while ($line = <$fh>) {
3   chomp $line;
4   print "$line: ";
5   if ($line =~ /^\((ba*ba*bb)*aa*ba*ba(abla)*$/) {
6     print "Accept\n"
7   } else {
8     print "Reject\n"
9   }
10 }
```

```perl
$ perl hw6revisited.pl hw6_exs.txt
abbba: Reject
babba: Accept
abb: Reject
abba: Accept
aaba: Reject
aabbba: Accept
bab: Reject
b: Reject
da: Reject
: Reject
abbbabba: Accept
$`
Homework 6 Revisited

• FSA code:

```perl
1 open($fh, $ARGV[0]) or die "$ARGV[0] not found!\n";
2
3 $start = "\0";
4 $end = "\{0,abba\}";
5
6 $delta = "\{0\}",
7   "\{0,a\}",
8   "\{a,\{0,a\}\,\{0,ab\}\}",
9   "\{0,ab\}",
10  "\{a,\{0,a,abba\}\,\{ab,\{0\}\}\}",
11  "\{0,abba\}",
12
13 while ($line = <$fh>) {
14   chomp $line;
15   foreach $c (split '//',$line) {
16     $state = $delta{$state}{$c}
17   }
18   if ($state eq $end) {
19     print "accept\n"
20   } else {
21     print "reject\n"
22   }
23   
```

• Regex code:

```perl
1 open($fh, $ARGV[0]) or die "$ARGV[0] not found!\n";
2
3 while ($line = <$fh>) {
4   chomp $line;
5   print "$line: ";
6   if ($line =~ /(^baab|baa|aba|aab)\*(ba|aa)\*(ba|aa)\*$/) {
7     print "Accept\n"
8   } else {
9     print "Reject\n"
10 }
11 }
```

• Which one would you prefer to write and why?