Automata and Formal Languages

Email: christian.urban at kcl.ac.uk
Office: S1.27 (1st floor Strand Building)
Slides: KEATS (also home work is there)

2nd CW

Remember we showed that

$$der\ c\ (r^+) = (der\ c\ r) \cdot r^*$$

2nd CW

Remember we showed that

$$der\ c\ (r^+) = (der\ c\ r) \cdot r^*$$

Does the same hold for $r^{\{n\}}$ with $n > 0$

$$der\ c\ (r^{\{n\}}) = (der\ c\ r) \cdot r^{\{n-1\}}\ ?$$

2nd CW

- *der*

$$der\ c\ (r^{\{n\}}) \stackrel{\text{def}}{=} \begin{cases} \varnothing & \text{if } n = 0 \\ der\ c\ (r \cdot r^{\{n-1\}}) & \text{o'wise} \end{cases}$$

- *mkeps*

$$mkeps(r^{\{n\}}) \stackrel{\text{def}}{=} \underbrace{[mkeps(r), \ldots, mkeps(r)]}_{n\ times}$$

- *inj*

$$inj\ r^{\{n\}}\ c\ (v_1, [vs]) \stackrel{\text{def}}{=} [inj\ r\ c\ v_1 :: vs]$$

$$inj\ r^{\{n\}}\ c\ Left(v_1, [vs]) \stackrel{\text{def}}{=} [inj\ r\ c\ v_1 :: vs]$$

$$inj\ r^{\{n\}}\ c\ Right([v :: vs]) \stackrel{\text{def}}{=} [mkeps(r) :: inj\ r\ c\ v :: vs]$$

Compilers in Boeings 777

They want to achieve triple redundancy in hardware faults.

They compile 1 Ada program to

- Intel 80486
- Motorola 68040 (old Macintosh's)
- AMD 29050 (RISC chips used often in laser printers)

Proofs about Rexps

Remember their inductive definition:

$$
\begin{aligned}
r \ ::= \ & \varnothing \\
| \ & \epsilon \\
| \ & c \\
| \ & r_1 \cdot r_2 \\
| \ & r_1 + r_2 \\
| \ & r^*
\end{aligned}
$$

If we want to prove something, say a property $P(r)$, for all regular expressions r then ...

Proofs about Rexp (2)

- P holds for $\varnothing$, ϵ and c

- P holds for $r_1 + r_2$ under the assumption that P already holds for r_1 and r_2.

- P holds for $r_1 \cdot r_2$ under the assumption that P already holds for r_1 and r_2.

- P holds for r^* under the assumption that P already holds for r.

$$zeroable(\varnothing) \quad \overset{\text{def}}{=} \quad true$$
$$zeroable(\epsilon) \quad \overset{\text{def}}{=} \quad false$$
$$zeroable(c) \quad \overset{\text{def}}{=} \quad false$$
$$zeroable(r_1 + r_2) \overset{\text{def}}{=} zeroable(r_1) \wedge zeroable(r_2)$$
$$zeroable(r_1 \cdot r_2) \quad \overset{\text{def}}{=} zeroable(r_1) \vee zeroable(r_2)$$
$$zeroable(r^*) \quad \overset{\text{def}}{=} \quad false$$

$$zeroable(r) \text{ if and only if } L(r) = \{\}$$

Correctness of the Matcher

- We want to prove

$$\textit{matches } r \textit{ s} \text{ if and only if } s \in L(r)$$

where $\textit{matches } r \textit{ s} \stackrel{\text{def}}{=} \textit{nullable}(\textit{ders } s \textit{ r})$

Correctness of the Matcher

- We want to prove

  matches r s if and only if $s \in L(r)$

  where *matches r s* $\overset{\text{def}}{=}$ *nullable*(*ders s r*)

- We can do this, if we know

  $$L(\textit{der c r}) = \textit{Der c } (L(r))$$

Induction over Strings

- case []:

  We need to prove

  $$\forall r. \ nullable(ders \ [] \ r) \ \Leftrightarrow \ [] \in L(r)$$

  $$nullable(ders \ [] \ r) \ \stackrel{\text{def}}{=} \ nullable \ r \ \Leftrightarrow \ \ldots$$

Induction over Strings

- case $c :: s$

We need to prove

$$\forall r.\ \textit{nullable}(\textit{ders}\ (c :: s)\ r)\ \Leftrightarrow\ (c :: s) \in L(r)$$

We have by IH

$$\forall r.\ \textit{nullable}(\textit{ders}\ s\ r)\ \Leftrightarrow\ s \in L(r)$$

$$\textit{ders}\ (c :: s)\ r \stackrel{\text{def}}{=} \textit{ders}\ s\ (\textit{der}\ c\ r)$$

Induction over Regexps

- The proof hinges on the fact that we can prove

$$L(der\ c\ r) = Der\ c\ (L(r))$$

Some Lemmas

- $Der\ c\ (A \cup B) = (Der\ c\ A) \cup (Der\ c\ B)$

- If $[] \in A$ then
  $$Der\ c\ (A\,@\,B) = (Der\ c\ A)\,@\,B\ \cup\ (Der\ c\ B)$$

- If $[] \notin A$ then
  $$Der\ c\ (A\,@\,B) = (Der\ c\ A)\,@\,B$$

- $Der\ c\ (A^*) = (Der\ c\ A)\,@\,A^*$

  (interesting case)

Why?

Why does $Der\ c\ (A^*) = (Der\ c\ A)\,@\,A^*$ hold?

$$
\begin{aligned}
Der\ c\ (A^*) &= Der\ c\ (A^* - \{[]\}) \\
&= Der\ c\ ((A - \{[]\})\,@\,A^*) \\
&= (Der\ c\ (A - \{[]\}))\,@\,A^* \\
&= (Der\ c\ A)\,@\,A^*
\end{aligned}
$$

using the facts $Der\ c\ A = Der\ c\ (A - \{[]\})$ and
$$(A - \{[]\})\,@\,A^* = A^* - \{[]\}$$