Assignment G1

Morten Fausing & Jens Kristian Refsgaard Nielsen & Thomas Vinther

07-02-2019

1.

We wish to prove the following fact about pre:

$$\forall x, y \in \Sigma^* : pre(xy) = pre(x) \cup \{x\}pre(y)$$

With pre given as follows:

$$\text{pre}(w) := \{u \in \Sigma^* \mid \exists v \in \Sigma^* : uv = w\}$$

We wish to prove the equality between the two sets, and therefore start by showing the inclusion $pre(xy) \subseteq pre(x) \cup \{x\}pre(y)$
Let $z \in pre(xy)$ by definition $\exists v \in \Sigma^* : zv = xy$
When $z \in pre(xy)$, and by the just mentioned definition, we only have three possible cases for v: It is the empty word and $z = xy$, it is a part of y up to being entire y, or it is a part of x (whole x) and entire y. By proving each case, we will have proven the inclusion.

At the same time we define: $suf(w) = \{u \in \Sigma^* \mid \exists v \in \Sigma^* : vu = w\}$ and divide into the three possible cases:

Case 1: $v = \Lambda$
$z = xy \in \{x\}pre(xy)$
If v is the empty word, z must be equal to xy, and therefore be in the set of $\{x\}pre(y)$

Case 2: $v \in suf(y)$
$v \in suf(y) \Rightarrow \exists w : wv = y \Rightarrow w \in pre(y)$ which gives us: $z = xw \in \{x\}pre(y)$
If $v \in suf(y)$ which also includes y itself, there must exist a w such that $wv = y$ which in turn implies that $z = xw$ by the definition of $pre(xy)$.

Case 3: $v \in suf(x)\{y\}$
By our definition of $v \in suf(x)\{y\}$ which gives us $\exists w \in \Sigma^* : v = wy$ now we look at $zwy = zv = xy \Rightarrow zw = x \Rightarrow z \in pre(x)$
When $v \in suf(x)\{y\}$ we are able to write $v = wy$, as we have chosen $z \in pre(xy)$ we can once again use the definition and write $zwy = xy$, which implies that x can be written as zw which gives the desired: $x \in pre(x)$ as wanted

We have now proved that $pre(xy) \subseteq pre(x) \cup \{x\}pre(y)$, we will now show the other inclusion:

$pre(xy) \supseteq pre(x) \cup \{x\}pre(y)$
Let $z \in pre(x) \Rightarrow \exists v \in \Sigma^* : zv = x \Rightarrow zvy = xy \Rightarrow z \in pre(xy)$

If $z \in pre(x)$ we use the definition, and by adding y we simply enlarge the set and get $z \in pre(xy)$

Let $z \in \{x\}pre(y) \Rightarrow \exists v \in \Sigma^* : zv = xy \Rightarrow z \in pre(xy)$
We simply use the definition of pre if $z \in \{x\}pre(y)$
We have now shown $pre(x) \cup \{x\}pre(y) \subseteq pre(xy)$ and proven the desired equality.

2.

We define $Pre(L(E)) := \cup_{w \in L(E)} pre(w)$, the prefix language.
We now wish to prove that for each regular expression E there exists another regular expression P such that $Pre(L(E)) = L(E)$, i.e. that the set of regular languages is closed under taking prefix languages. We proceed by structural induction.

Basis cases:
If E = $\emptyset$, let P = $\emptyset$:

$$\text{Pre}(L(E)) = \text{ Pre}(\mathcal{L}(\emptyset)) = \bigcup_{w \in \emptyset} \text{pre}(w) = \emptyset = \mathcal{L}(\emptyset) = \mathcal{L}(P)$$

If E = a, let P = a+Λ:

$$\begin{aligned} \text{Pre}(\mathcal{L}(E)) = \text{Pre}(\mathcal{L}(a)) = & \bigcup_{w \in \mathcal{L}(a)} \text{pre}(w) = \text{pre}(a) = \{\text{a}, \Lambda\} = \mathcal{L}(a + \Lambda) \\ = \mathcal{L}(P) & \end{aligned}$$

Induction step:
If E=E_1+E_2, the induction hypothesis gives regular expressions $E'_i : \text{Pre}(\mathcal{L}(E_i)) = \mathcal{L}(E'_i)$, choose then the regular expression $P = (E'_1 + E'_2)$:
Recall that $\Omega : \cup_{d \in (A \cup B)} f(d) = (\cup_{d \in A} f(d)) \cup (\cup_{d \in B} f(d))$

$$\begin{aligned} \text{Pre}(\mathcal{L}(E)) &= \text{Pre}(\mathcal{L}(E_1 + E_2)) = \text{Pre}(\mathcal{L}(E_1) \cup \mathcal{L}(E_2)) \\ &= \bigcup_{w \in (\mathcal{L}(E_1) \cup \mathcal{L}(E_2))} \text{pre(w)} \\ &\overset{\Omega}{=} \left(\bigcup_{w \in \mathcal{L}(E_1)} \text{pre}(w) \right) \cup \left(\bigcup_{w \in \mathcal{L}(E_2)} \text{pre}(w) \right) \\ &= \text{Pre}(\mathcal{L}(E_1)) \cup \text{Pre}(\mathcal{L}(E_2)) \overset{I.H.}{=} \mathcal{L}(E'_1) \cup \mathcal{L}(E'_2) \\ &= \mathcal{L}(E'_1 + E'_2) = \mathcal{L}(P) \end{aligned}$$

If E=$E_1 E_2$, the induction hypothesis once again gives regular expressions $E'_i : \text{Pre}(\mathcal{L}(E_i)) =$

$\mathcal{L}(E_i')$, now we choose the regular expression $P = (E_1' + E_1 E_2')$

$$\begin{aligned}
\operatorname{Pre}(\mathcal{L}(E)) &= \operatorname{Pre}(\mathcal{L}(E_1E_2)) = \operatorname{Pre}(\mathcal{L}(E_1)\mathcal{L}(E_2)) \\
&= \bigcup_{w\in\mathcal{L}(E_1)\mathcal{L}(E_2)} \operatorname{pre}(w) \\
&= \bigcup_{w_1\in\mathcal{L}(E_1), w_2\in\mathcal{L}(E_2)} \operatorname{pre}(w_1w_2) \\
&\overset{1.}{=} \bigcup_{w_1\in\mathcal{L}(E_1), w_2\in\mathcal{L}(E_2)} (\operatorname{pre}(w_1) \cup \{w_1\}\operatorname{pre}(w_2)) \\
&= \left(\bigcup_{w_1\in\mathcal{L}(E_1), w_2\in\mathcal{L}(E_2)} \operatorname{pre}(w_1)\right) \cup \left(\bigcup_{w_1\in\mathcal{L}(E_1), w_2\in\mathcal{L}(E_2)} \{w_1\}\operatorname{pre}(w_2))\right) \\
&= \left(\bigcup_{w_1\in\mathcal{L}(E_1)} \operatorname{pre}(w_1)\right) \cup \left(\bigcup_{w_1\in\mathcal{L}(E_1)} \{w_1\}\right)\left(\bigcup_{w_2\in\mathcal{L}(E_2)} \operatorname{pre}(w_2))\right) \\
&= \operatorname{Pre}(\mathcal{L}(E_1)) \cup \mathcal{L}(E_1)\operatorname{Pre}(\mathcal{L}(E_2)) \\
&\overset{I.H.}{=} \mathcal{L}(E_1') \cup \mathcal{L}(E_1)\mathcal{L}(E_2') \\
&= \mathcal{L}(E_1' + E_1E_2') = \mathcal{L}(P)
\end{aligned}$$

Kleene star case:
The following will become useful, consider for $i > 0$

$$\begin{aligned}
\nabla : \operatorname{Pre}(\mathcal{L}(E)^i) &= \operatorname{Pre}(\mathcal{L}(E)^{i-1}\mathcal{L}(E)) \\
&\overset{1)}{=} \operatorname{Pre}(\mathcal{L}(E)^{i-1}) \cup \mathcal{L}(E)^{i-1} \cdot \operatorname{Pre}(\mathcal{L}(E)) \\
&= \left(\operatorname{Pre}(\mathcal{L}(E)^{i-2}) \cup \mathcal{L}(E)^{i-2} \cdot \operatorname{Pre}(\mathcal{L}(E))\right) \cup \mathcal{L}(E)^{i-1} \cdot \operatorname{Pre}(\mathcal{L}(E)) \\
&\vdots \\
&= \bigcup_{0\le j<i} \mathcal{L}(E)^j \cdot \operatorname{Pre}(\mathcal{L}(E)) \\
&= \left(\bigcup_{0\le j<i} \mathcal{L}(E)^j\right) \cdot \operatorname{Pre}(\mathcal{L}(E))
\end{aligned}$$

By applying part 1 of the assignment a finite number of times or a short induction omitted here, we see that taking powers of prefix languages it is sufficient to concatenate on the right by the prefix language of the base expression.
If $E = E_1^*$, the induction hypothesis grants us a regular expression $E_1' : \operatorname{Pre}(\mathcal{L}(E_1)) = E_1'$, choose then the regular expression $P = E_1^* E_1$

$$
\begin{aligned}
\mathrm{Pre}(\mathcal{L}(E)) = \mathrm{Pre}(\mathcal{L}(E_1^*)) &= \mathrm{Pre}\left(\bigcup_{i\geq 0} \mathcal{L}(E_1)^i\right) \\
&= \bigcup_{i\geq 0} \mathrm{Pre}(\mathcal{L}(E_1)^i) \overset{\triangledown}{=} \bigcup_{i\geq 0} \mathcal{L}(E_1)^i \mathrm{Pre}(\mathcal{L}(E_1)) \\
&= \mathcal{L}(E_1^*)\mathrm{Pre}(\mathcal{L}(E_1)) \overset{I.H.}{=} \mathcal{L}(E_1^*)\mathcal{L}(E_1') \\
&= \mathcal{L}(E_1^* E_1') = \mathcal{L}(P)
\end{aligned}
$$

Thus concluding the proof by structural induction. $\square$

3.

By using the previous proof, we extract a function pref on regular expressions, such that $\mathcal{L}(\mathrm{pref}(E)) = \mathrm{Pre}(\mathcal{L}(E))$. Here we show the recursive definition:

$$
\mathrm{pref}(E) = \begin{cases} \emptyset & \text{if } E = \emptyset \\ a + \Lambda & \text{if } E = a \\ \mathrm{pref}(E_1) + \mathrm{pref}(E_2) & \text{if } E = E_1 + E_2 \\ \mathrm{pref}(E_1) + E_1\mathrm{pref}(E_2) & \text{if } E = E_1E_2 \\ E_1^*\mathrm{pref}(E_1) & \text{if } E = E_1^* \end{cases}
$$

4.

We compute the result of $\mathrm{pref}(a^*(b+cd)^*)$ by using our definition of pref.

$$
\begin{aligned}
\mathrm{pref}(a^*(b+cd)^*) &= \mathrm{pref}(a^*) + a^*\mathrm{pref}((b+cd)^*) \\
&= a^*\mathrm{pref}(a) + a^*(b+cd)^*\mathrm{pref}(b+cd) \\
&= a^*(a+\Lambda) + a^*(b+cd)^*(\mathrm{pref}(b) + \mathrm{pref}(cd)) \\
&= a^*(a+\Lambda) + a^*(b+cd)^*(b+\Lambda+\mathrm{pref}(c) + (c)\mathrm{pref}(d)) \\
&= a^*(a+\Lambda) + a^*(b+cd)^*(b+\Lambda+(c+\Lambda)+c(d+\Lambda))
\end{aligned}
$$

We can simplify this expression a bit

$$
\begin{aligned}
\mathrm{pref}(a^*(b+cd)^*) &= a^*(a+\Lambda) + a^*(b+cd)^*(b+\Lambda+(c+\Lambda)+c(d+\Lambda)) \\
&\equiv a^+ + a^* + a^*(b+cd)^*(b+\Lambda+c+\Lambda+cd+c) \\
&\equiv a^* + a^*(b+cd)^*(\Lambda+b+c+cd)
\end{aligned}
$$

In the above we use the equivalence relation on regular expressions:

$$
E \equiv P \iff \mathcal{L}(E) = \mathcal{L}(P)
$$

Recall that by definition $a^+ \subsetneq a^*$ so $\mathcal{L}(a^+ + a^*) = \mathcal{L}(a^*)$ and $a^+ + a^* \equiv a^*$.