CPSC 313 — Supplemental Material for Lecture #5
Computation of "λ-Closures"

Introduction

The goal of this document is to describe an algorithm that can be used to compute the function

$$Cl_\lambda : Q \to \mathcal{P}(Q)$$

that was described during the first lecture on nondeterministic finite automata. As described in that lecture if $q \in Q$ then $Cl_\lambda(q)$ is the **set** of states that can be reached from q by following zero of *more* λ-transitions — so that

$$\delta^\star(q, \lambda) = Cl_\lambda(q)$$

for each state $q \in Q$. As noted in the lecture notes, this set is called the "λ-closure" of the state q[1]

Another Useful Function That is Easier To Describe and Compute

Let us begin by considering a different function instead, namely, the function

$$\widehat{Cl}_\lambda : Q \times \mathbb{N} \to \mathcal{P}(Q)$$

such that, for every state $q \in Q$ and every integer $k \geq 0$, $\widehat{Cl}_\lambda(q, k)$ is the set of states in Q that can be reached from q by following **at most** k λ-transitions.

Notice that $\widehat{Cl}_\lambda(q, 0) = \{q\}$, since q is the only state that can be reached from q without following any transitions at all. On the other hand, if $k \geq 1$ then $\widehat{Cl}_\lambda(q, k)$ includes all the states in $\widehat{Cl}_\lambda(q, k-1)$, as well as all (and only) the additional states that can be reached

---

[1] ...or it is called the "ε-closure" of q if the symbol ε is being used to represent the empty string instead of λ.

- by first following $k-1$ λ-transitions from q — so that one of the states $r \in \widehat{Cl}_\lambda(q, k-1)$ is reached, and then

- following one more λ-transition, to reach one of the states in $q(r, \lambda)$.

This provides the following "inductive" or "recursive" definition of this function: For every state $q \in Q$ and for every integer $k \geq 0$,

$$\widehat{Cl}_\lambda(q, k) = \begin{cases} \{q\} & \text{if } k = 0, \\ \widehat{Cl}_\lambda(q, k-1) \cup \bigcup_{r \in \widehat{Cl}_\lambda(q, k-1)} \delta(r, \lambda) & \text{if } k \geq 1. \end{cases} \tag{1}$$

Notice that

$$\widehat{Cl}_\lambda(q, k) \subseteq \widehat{Cl}_\lambda(q, k+1) \subseteq Cl_\lambda(q)$$

for every integer $k \geq 0$. The next two results explain more about the relationship between these sets and will be useful when an algorithm to compute $Cl_\lambda(q)$ (for a given state q) is developed and analyzed.

Claim 1. *Let q be a state and let k be a nonnegative integer. Suppose that $\widehat{Cl}_\lambda(q, k) = \widehat{Cl}_\lambda(q, k+1)$.*

(a) $\widehat{Cl}_\lambda(q, k) = \widehat{Cl}_\lambda(q, k+m)$ *as well, for every integer $m \geq 1$.*

(b) $\widehat{Cl}_\lambda(q, k) = Cl_\lambda(q)$.

Proof. Part (a) will be proved by induction on m.

Basis: If $m = 1$ then it is necessary and sufficient to prove that $\widehat{Cl}_\lambda(q, k) = \widehat{Cl}_\lambda(q, k+1)$. However, this is given as an assumption.

Inductive Step: Let m be an integer such that $m \geq 1$. It is now necessary and sufficient to use the following

    Inductive Hypothesis: $\widehat{Cl}_\lambda(q, k) = \widehat{Cl}_\lambda(q, k+m)$

to prove the following

    Inductive Claim: $\widehat{Cl}_\lambda(q, k) = \widehat{Cl}_\lambda(q, k+m+1)$.

Now, since $k + m \geq k + 1 \geq 1$,

$$\widehat{Cl}_\lambda(q, k + m + 1) = \widehat{Cl}_\lambda(q, k + m) \cup \bigcup_{r \in \widehat{Cl}_\lambda(q, k+m)} \delta(r, \lambda)$$

(by the equation at line (1), with k replaced by $k + m + 1$)

$$= \widehat{Cl}_\lambda(q, k) \cup \bigcup_{r \in \widehat{Cl}_\lambda(q, k)} \delta(r, \lambda) \qquad \text{(by the inductive hypothesis)}$$

$$= \widehat{Cl}_\lambda(q, k + 1)$$

(by an application of the equation at line (1) once again, with k replaced by $k + 1$)

$$= \widehat{Cl}_\lambda(q, k) \qquad \text{(by the result established in the basis)}$$

as needed to complete the inductive step.

Part (a) of the claim now follows by induction on m.

It is certainly true that $\widehat{Cl}_\lambda(q, k) \subseteq Cl_\lambda(q)$. It is therefore sufficient to prove that $Cl_\lambda(q) \subseteq \widehat{Cl}_\lambda(q, k)$, as well, in order to prove part (b).

With that noted, let $r \in Cl_\lambda(q)$. Then r can be reached from q by following a (finite) sequence of zero or more λ-transitions, so that $r \in \widehat{Cl}_\lambda(q, \ell)$ for some integer $\ell \geq 0$. (In particular, ℓ is the length of a sequence of λ-transitions that start at q and end at r).

If $\ell \leq k$ then $r \in \widehat{Cl}_\lambda(q, k)$ as well, since $\widehat{Cl}_\lambda(q, \ell) \subseteq \widehat{Cl}_\lambda(q, k)$.

On the other hand, if $\ell > k$ then $\ell = k + m$ for some integer $m \geq 1$ and $r \in \widehat{Cl}_\lambda(q, k)$ once again, because it follows by part (a) that $\widehat{Cl}_\lambda(q, \ell) = \widehat{Cl}_\lambda(q, k)$.

Since r was arbitrarily chosen from $Cl_\lambda(q)$ it now follows that $Cl_\lambda(q) \subseteq \widehat{Cl}_\lambda(q, k)$, as required to complete the proof. □

Claim 2. *If k is a positive integer such that $\widehat{Cl}_\lambda(q, k - 1) \neq \widehat{Cl}_\lambda(q, k)$ then*

$$|\widehat{Cl}_\lambda(q, k)| \geq k + 1.$$

The ***proof*** of this claim is left as an easy exercise. (*Hint:* Use induction on k and remember that $\widehat{Cl}_\lambda(q, k - 1) \subseteq \widehat{Cl}_\lambda(q, k)$.)

Corollary 3. *If $k \geq |Q| - 1$ then $\widehat{Cl}_\lambda(q, k) = Cl_\lambda(q)$.*

Proof. Suppose, to the contrary, that $k \geq |Q| - 1$ and $\widehat{Cl}_\lambda(q, k) \neq Cl_\lambda(q)$. Then it follows by the first claim that $\widehat{Cl}_\lambda(q, k) \neq \widehat{Cl}_\lambda(q, k + 1)$. However, it now follows by the second claim (with k replaced by $k + 1$) that

$$|\widehat{Cl}_\lambda(q, k + 1)| \geq k + 2 \geq |Q| + 1,$$

and this is certainly impossible, since $\widehat{Cl}_\lambda(q, k + 1) \subseteq Q$. □

3

```
On input q ∈ Q:

int k := 0;

R := {q};                        // Now R = Ĉl_λ(q, k)
S := R ∪ ⋃_{r∈R} δ(r, λ);        // Now S = Ĉl_λ(q, k + 1)

while (R ≠ S) do

  k := k + 1;

  R := S                          // Now R = Ĉl_λ(q, k), once again
  S := R ∪ ⋃_{r∈R} δ(r, λ);       // Now S = Ĉl_λ(q, k + 1), once again

end while;

return R;
```

Figure 1: An Algorithm To Compute $Cl_\lambda(q)$

An Algorithm To Compute $Cl_\lambda(q)$

An algorithm to compute $Cl_\lambda(q)$ (for a given state q) can now be described and analyzed. Pseudocode for this algorithm is shown in Figure 1. In this algorithm, R and S are both subsets of Q.

Claim 4. *Suppose that the body of the* while *loop is executed at least ℓ times, for an integer $\ell \geq 1$. Then*

- *$k = \ell - 1$, $R = \widehat{Cl}_\lambda(q, \ell - 1)$, and $S = \widehat{Cl}_\lambda(q, \ell)$ at the* beginning *of the ℓth execution of the body of the loop, and*

- *$k = \ell$, $R = \widehat{Cl}_\lambda(q, \ell)$, and $S = \widehat{Cl}_\lambda(q, \ell + 1)$ at the* end *of the ℓth execution of the body of the loop.*

How To Prove This: Use mathematical induction on ℓ and inspection of the code (the comments that are included in the code are intended to make this easy to do).

Claim 5. *If the algorithm is executed with a state $q \in Q$ as input then the body of the* while *loop is executed at most $|Q| - 1$ times before the* while *loop's test fails and the* while *loop is exited.*

Proof. If the body of the `while` loop is executed *fewer* than $|Q| - 1$ times then the claim is certainly correct. On the other hand, it follows by the previous claim that if the body of the loop *is* executed $|Q| - 1$ times then $R = \widehat{Cl}_\lambda(q, |Q| - 1)$ and $S = \widehat{Cl}_\lambda(q, |Q|)$ at the end of this execution of the loop body. It now follows by Corollary 3 that $R = S = Cl_\lambda(q)$ at this point, so that the loop test fails — is needed to establish the claim. $\qquad\square$

Claim 6. *The algorithm is correct. That is, if is executed with a state $q \in Q$ as input then the algorithm always halts, and it always returns* $\mathrm{Cl}_\lambda(q)$ *as output.*

Proof. Claim 5 establishes that this program always halts. One can see by an inspection of the code (as well as a consideration of Claim 4 that, when it halts, it returns a set $\widehat{Cl}_\lambda(q, \ell)$ as output, where $S = \widehat{Cl}_\lambda(q, \ell+1) = \widehat{Cl}_\lambda(q, \ell)$. It now follows by part (b) of Claim 1 that the output returned is the set $Cl_\lambda(q)$, as needed to complete the proof that the algorithm is correct. $\qquad\square$

Suggested Exercises

1. Trace the execution of this algorithm, by hand, to compute $Cl_\lambda(q)$ for each of the states q in the nondeterministic finite automaton considered in Tutorial Exercise #4.

2. Look for ways to make this algorithm more efficient (by eliminating computations that do not change the sets that are computed, so that the algorithm remains correct).

3. Students who have already completed CPSC 331 or 319 should consider a **directed graph** $G = (V, E)$ in which $V = Q$ and, for $r, s \in V$, such that $r \neq s$, there is an edge *from r to s* if and only if $s \in \delta(r, \lambda)$.

   Try to prove the following[2]: For all states r and s, $s \in Cl_\lambda(r)$ if and only if there is a **path** *from r to s* (with edges followed in the right direction) in G.

   Now, try to identify one (or more!) algorithms that you learned about in CPSC 319 or 331 that can also be used to compute $Cl_\lambda(q)$ for a given state q, and that are probably more efficient than the one that has been described in this document.

---

[2] The proof is actually not very difficult to find!