Kleene Algebra — Lecture 7

Tobias Kappé

January 2022

1 Today's lecture

So far, we have talked a great deal about rational expressions and their semantics. We have seen a number of laws that can be used to show that rational expressions are equivalent, either under the relational semantics (w.r.t. all interpretations) or the language semantics. We have also seen that rational expressions can be converted to automata accepting the same language, and how we can use those automata as a proxy for deciding language equivalence.

So, we know that provably equivalent rational expressions have the same language. I have also been teasing that the converse is true, namely that rational expressions with the same language can be proved equivalent. Over the past lectures, we have been developing the tools to support this claim: Brzozowski's construction and its converse, the matrix-based conversion from automata back to expressions. Today, we put the capstone on the proof, and formally show that the laws of Kleene Algebra are complete for language equivalence.

2 Brzozowski's construction

It's been a while since we've talked about Brzozowski's construction, so let's do a quick review. Recall that the idea behind this approach was to create a giant (infinite) automaton, where every state was represented by an expression, and the language of that state was exactly the language of that expression. This meant that we had to define which expressions gave rise to accepting states, and we did that by defining the set of expressions $\mathbb{A}$, as follows.

Definition 7.1. We define $\mathbb{A}$ as the smallest subset of $\mathbb{E}$ satisfying the rules

$$\frac{}{1 \in \mathbb{A}} \qquad \frac{e \in \mathbb{A} \qquad f \in \mathbb{E}}{e + f, f + e \in \mathbb{A}} \qquad \frac{e, f \in \mathbb{A}}{e \cdot f \in \mathbb{A}} \qquad \frac{e \in \mathbb{E}}{e^* \in \mathbb{A}}$$

The next thing we had to do was define the transitions between expressions. The idea here was to let the target of an $\mathtt{a}$-transition from e be the expressoin representing the remainder of words in e that start with an $\mathtt{a}$. It turns out that it is indeed possible to define such a transformation on expressions.

Definition 7.2. We define $d : \mathbb{E} \times \Sigma \to \mathbb{E}$ inductively, as follows:

$$d(0, \mathtt{a}) = 0 \qquad d(1, \mathtt{a}) = 0 \qquad d(\mathtt{b}, \mathtt{a}) = [\mathtt{b} = \mathtt{a}] \qquad d(e + f, \mathtt{a}) = d(e, \mathtt{a}) + d(f, \mathtt{a})$$

$$d(e \cdot f, \mathtt{a}) = d(e, \mathtt{a}) \cdot f + [e \in \mathbb{A}] \cdot d(f, \mathtt{a}) \qquad d(e^*, \mathtt{a}) = d(e, \mathtt{a}) \cdot e^*$$

Here, $[\Phi]$ is shorthand for 1 when Φ holds, and 0 otherwise.

We were also able to show this nifty fact about derivatives of expressions:

Theorem 7.3 (Fundamental theorem)**.** *Let $e \in \mathbb{E}$. The following holds:*

$$e \equiv [e \in \mathbb{A}] + \sum_{\mathtt{a} \in \Sigma} \mathtt{a} \cdot d(e, \mathtt{a})$$

This then made it easy to show that the Brzozowski automaton satisfied our objective of having each state be the language of the expression it represents.

Lemma 7.4. *Let $B = \langle \mathbb{E}, \mathbb{A}, d \rangle$ and $e \in \mathbb{E}$. Now $L_B(e) = [\![e]\!]_{\mathbb{E}}$.*

But our efforts did not stop there — to make our decision procedure work, we needed to find a *finite* automaton representing the language of e. To that end, we had to introduce a certain equivalence relation on $\mathbb{E}$.

Definition 7.5. We define $\sim$ as the smallest equivalence on $\mathbb{E}$ satisfying

$$e \sim e + e \qquad e + f \sim f + e \qquad e + (f + g) \sim (e + f) + g$$

$$0 \cdot e \sim 0 \qquad e + 0 \sim e \qquad 1 \cdot e \sim e$$

for all $e, f, g \in \mathbb{E}$. We write $[e]$ for the $\sim$-equivalence class of e, i.e., $\{e' \in \mathbb{E} : e \sim e'\}$. Also, we write $\hat{\mathbb{E}}$ for the quotient of $\mathbb{E}$ by $\sim$, i.e., $\{[e] : e \in \mathbb{E}\}$.

Eagle-eyed readers will notice that we have added three more rules to the definition of $\sim$, compared to its original definition. Luckily, these rules do not disqualify any of the claims that follow — the proofs still go through, in the same way. We add them here merely because they make our life easier later on.

It turned out that $\sim$ was compatible with $\mathbb{A}$ and d, in the sense that both were invariant under $\sim$-equivalence. This gave rise to the set $\hat{\mathbb{A}} = \{[e] : e \in \mathbb{A}\}$ and the function $\hat{d} : \hat{\mathbb{E}} \times \Sigma \to \hat{\mathbb{E}}$ given by $\hat{d}([e], \mathtt{a}) = [d(e, \mathtt{a})]$. We could then define the *quotiented* Brzozowski automaton, which also satisfied our need.

Lemma 7.6. *Let $\hat{B} = \langle \hat{\mathbb{E}}, \hat{\mathbb{A}}, \hat{d} \rangle$ and $e \in \mathbb{E}$. Now $L_{\hat{B}}([e]) = [\![e]\!]_{\mathbb{E}}$.*

Thanks to the quotienting operation, we could then show that the part of $\hat{B}$ that we were interested in, namely the states (expressions) reachable from a given expression e, is finite. Formally, this came down to the following.

Lemma 7.7. *Let $e \in \mathbb{E}$. The set $r(e) = \{\hat{d}^*([e], w) : w \in \Sigma^*\}$ is finite.*

Clearly, $r(e)$ is closed under $\hat{d}$ — that is, if $[f] \in r(e)$ and $\mathtt{a} \in \Sigma$, then $\hat{d}([f], \mathtt{a}) \in r(e)$ as well. This means that we can regard $\hat{d}$ as a function from $r(e) \times \Sigma$ to $r(e)$, and finally obtain the finite automaton we were looking for.

Lemma 7.8. *Let $e \in \mathbb{E}$ and $\hat{B}_e = \langle r(e), r(e) \cap \hat{\mathbb{A}}, \hat{d} \rangle$. Now $L_{\hat{B}_e}([e]) = [\![e]\!]_{\mathbb{E}}$.*

For the sake of convenience, let's shorten our subscripts from this point on: we write L_e instead of $L_{\hat{B}_e}$, M_e instead of $M_{\hat{B}_e}$, and b_e instead of $b_{\hat{B}_e}$.

3 The round-trip theorem

In Lecture 5, we saw that we could obtain an expression from a state in a finite automaton. Because every expression can be converted into an automaton, you might wonder: what happens if we take an expression, turn it into an automaton, and then back into an expression? Can we prove that this round-tripped expression is equivalent to the expression we started out with? Or perhaps equivalence of these programs is "lost in translation"?

If we leverage the notion of least solutions to automata, we can show that the old expression is at least an upper bound on the new expression. For the sake of convenience and brevity, let's write $K(e)$ for $(M_e^* \cdot b_e)([e])$.

Lemma 7.9. *Let $e \in \mathbb{E}$. Now $K(e) \leqq e$.*

Proof. Note that $M_e^* \cdot b_e$, as an $r(e)$-vector, is the least solution to $\hat{B}_e$ as an automaton. Specifically, this means that if s is an $r(e)$-vector where

$$\forall [f] \in r(e).\ [[f] \in \hat{\mathbb{A}}] + \sum_{\mathtt{a} \in \Sigma} \mathtt{a} \cdot s(\hat{d}([f], \mathtt{a})) \leqq s([f]) \implies M_e^* \cdot b_e \leqq s \qquad (1)$$

Let's choose one specific $r(e)$-vector, namely the one where $s([f]) = f$ for all $[f] \in r(e)$. Note that this is slightly ambiguous — we do not specify which representative of the equivalence class $[f]$ we choose to fill $s([f])$. Fortunately, our choice here does not matter: we can choose *any* such representative; the important part is that $s([f]) \equiv f$, since $\sim$ is contained in $\equiv$.

We now claim that s satisfies the premise of (1). To see this, note that, if we fill out the definition of $\hat{d}$, $\hat{\mathbb{A}}$ and s, this premise comes down to the following:

$$\forall [f] \in r(e).\ [f \in \mathbb{A}] + \sum_{\mathtt{a} \in \Sigma} \mathtt{a} \cdot d(f, \mathtt{a}) \leqq f$$

but this follows exactly from the Fundamental Theorem! We conclude that $M_e^* \cdot b_e \leqq s$, and thus in particular $K(e) = (M_e^* \cdot b_e)([e]) \leqq s([e]) \equiv e$. □

For the other direction, we first consider another technical lemma, which shows that — at the very least — equivalent terms result in equivalent terms when hit with the round-trip transformation. For the sake of the proofs that follow, it is easiest to state this in terms of the containment relation $\leqq$.

Lemma 7.10. *Let $e, f \in \mathbb{E}$ If $e \leqq f$, then $K(e) \leqq K(f)$.*

Proof. Consider we choose the set Q and the function $\delta : Q \times \Sigma \to Q$, where

$$Q = \{\langle \hat{d}^*([e], w), \hat{d}^*([f], w)\rangle : w \in \Sigma^*\} \qquad \delta(\langle [g], [h]\rangle, \mathtt{a}) = \langle \hat{d}([g], \mathtt{a}), \hat{d}([h], \mathtt{a})\rangle$$

Furthermore, we choose $F_1, F_2 \subseteq Q$ by setting

$$F_1 = \{\langle [g], [h]\rangle \in Q : g \in \mathbb{A}\} \qquad F_2 = \{\langle [g], [h]\rangle \in Q : h \in \mathbb{A}\}$$

This gives us two automata, $A_1 = \langle Q, F_1, \delta\rangle$ and $A_2 = \langle Q, F_2, \delta\rangle$. Finally, we choose $R_1 \subseteq r(e) \times Q$ and $R_2 \subseteq Q \times r(f)$ as the smallest relations satisfying

$$\frac{\langle [g], [h]\rangle \in Q}{[g]\ R_1\ \langle [g], [h]\rangle} \qquad \frac{\langle [g], [h]\rangle \in Q}{\langle [g], [h]\rangle\ R_1\ [h]}$$

It is not too hard to see that this makes R_1 a bisimulation between $\hat{B}_e$ and A_1, and R_2 a bisimulation between A_2 and $\hat{B}_f$. Since $\langle [e],[f] \rangle \in Q$, $[e] \mathrel{R_1} \langle [e],[f] \rangle$ and $\langle [e],[f] \rangle \mathrel{R_2} [f]$, and bisimilar states have equivalent solutions, we know

$$K(e) \equiv (M_{A_1}^* \cdot b_{A_1})(\langle [e],[f] \rangle) \qquad (M_{A_2}^* \cdot b_{A_2})(\langle [e],[f] \rangle) \equiv K(f)$$

We now claim that $b_{A_1} \leqq b_{A_2}$. To see this, let $\langle [g],[h] \rangle \in Q$, and note that $[g] = \hat{d}^*([e], w)$ and $[h] = \hat{d}^*([f], w)$ for some $w \in \Sigma^*$. Now, if $\langle [g],[h] \rangle \in F_1$, then $\hat{d}^*([e], w) = [g] \in \hat{\mathbb{A}}$, and hence $w \in L_e([e]) = [\![e]\!]_{\mathbb{E}}$. But since $e \leqq f$, also $[\![e]\!]_{\mathbb{E}} \subseteq [\![f]\!]_{\mathbb{E}}$, meaning that $w \in [\![f]\!]_{\mathbb{E}} = L_f([f])$, and thus $[h] = \hat{d}^*([f], w) \in \hat{\mathbb{A}}$, whence $\langle [g],[h] \rangle \in F_2$. Thus, if $b_{A_1}(\langle [g],[h] \rangle) = 1$, then $b_{A_2}(\langle [g],[h] \rangle) = 1$.

Since A_1 and A_2 have the same transition function, $M_{A_1} = M_{A_2}$. This then allows us to derive that $M_{A_1}^* \cdot b_{A_1} \leqq M_{A_2}^* \cdot b_{A_2}$, and hence $K(e) \leqq K(f)$. □

This lemma then allows us to show the other direction of the round-trip property we were looking for. Key to the proof here is that we first prove a slightly more general property, which makes the inductive argument possible.

Lemma 7.11. *Let $e \in \mathbb{E}$. Now $e \leqq K(e)$.*

Proof. We claim that, for all $f \in \mathbb{E}$, it holds that $e \cdot K(f) \leqq K(e \cdot f)$. To see this, we proceed by induction on e. In the base, there are three cases to consider.

- If $e = 0$, then the claim holds immediately.
- If $e = 1$, then we can derive as follows:

$$e \cdot K(f) = 1 \cdot K(f) \equiv K(f) \equiv K(1 \cdot f) = K(e \cdot f)$$

- If $e = \mathsf{a}$ for some $\mathsf{a} \in \Sigma$, then note that $r(f) \subseteq r(\mathsf{a} \cdot f)$. After all, if $[g] \in r(f)$, then $[g] = \hat{d}^*([f], w)$ for some $w \in \Sigma^*$. It then follows that $[g] = [1 \cdot g] = [1 \cdot d^*(f, w)] = [d^*(\mathsf{a} \cdot f, \mathsf{a}w)] \in r(\mathsf{a} \cdot f)$.

  It is not too hard to show that this makes $\mathsf{id}_{r(f)}$ a bisimulation between $\hat{B}_f$ and $\hat{B}_{\mathsf{a} \cdot f}$, relating $[f]$ to $[f]$. By the result from the previous lecture, this tells us that $(M_f^* \cdot b_f)([f]) \equiv (M_{\mathsf{a} \cdot f}^* \cdot b_{\mathsf{a} \cdot f})([f])$. We then derive:

$$\begin{aligned}
e \cdot K(f) &= \mathsf{a} \cdot K(f) \\
&\equiv \mathsf{a} \cdot (M_{\mathsf{a} \cdot f}^* \cdot b_{\mathsf{a} \cdot f})([f]) \\
&\leqq M_{\mathsf{a} \cdot f}([\mathsf{a} \cdot f], [f]) \cdot (M_{\mathsf{a} \cdot f}^* \cdot b_{\mathsf{a} \cdot f})([f]) \\
&\leqq (M_{\mathsf{a} \cdot f} \cdot M_{\mathsf{a} \cdot f}^* \cdot b_{\mathsf{a} \cdot f})([\mathsf{a} \cdot f]) \\
&\leqq (M_{\mathsf{a} \cdot f}^* \cdot b_{\mathsf{a} \cdot f})([\mathsf{a} \cdot f]) \\
&= K(\mathsf{a} \cdot f) = K(e \cdot f)
\end{aligned}$$

For the inductive step, there are again three cases to consider, based on the top-level compositional operator. For each case, our induction hypothesis is that the claim holds for each of the direct subexpressions.

- If $e = e_0 + e_1$, then derive

$$\begin{aligned}
e \cdot K(f) &= (e_0 + e_1) \cdot K(f) \\
&\equiv e_0 \cdot K(f) + e_1 \cdot K(f) \\
&\leqq K(e_0 \cdot f) + K(e_1 \cdot f) && \text{(IH)} \\
&\leqq K(e_0 \cdot f + e_1 \cdot f) && \text{(Lemma 7.10)} \\
&\equiv K((e_0 + e_1) \cdot f) && \text{(Lemma 7.10)} \\
&= K(e \cdot f)
\end{aligned}$$

- If $e = e_0 \cdot e_1$, then derive

$$\begin{aligned}
e \cdot K(f) &= (e_0 \cdot e_1) \cdot K(f) \\
&\equiv e_0 \cdot (e_1 \cdot K(f)) \\
&\leqq e_0 \cdot K(e_1 \cdot f) && \text{(IH)} \\
&\leqq K(e_0 \cdot (e_1 \cdot f)) && \text{(IH)} \\
&\equiv K((e_0 \cdot e_1) \cdot f) && \text{(Lemma 7.10)} \\
&= K(e \cdot f)
\end{aligned}$$

- If $e = e_0^*$, then we derive as follows:

$$\begin{aligned}
K(f) + e_0 \cdot K(e \cdot f) &\leqq K(f) + K(e_0 \cdot e \cdot f) && \text{(IH)} \\
&\leqq K(f + e_0 \cdot e \cdot f) && \text{(Lemma 7.10)} \\
&\equiv K((1 + e_0 \cdot e) \cdot f) && \text{(Lemma 7.10)} \\
&\leqq K(e \cdot f) && \text{(Lemma 7.10)}
\end{aligned}$$

  It then follows that $e \cdot K(f) = e_0^* \cdot K(f) \leqq K(e \cdot f)$.

To reach the claim, we note that $1 \leqq K(1)$, since

$$1 = b_1([1]) \leqq (M_1^* \cdot b_1)([1]) = K(1)$$

With this in hand, we conclude by deriving that

$$e \equiv e \cdot 1 \leqq e \cdot K(1) \leqq K(e \cdot 1) \equiv K(e)$$

□

These two lemmas together now imply the property we were looking for.

Theorem 7.12 (Round-trip). *Let $e \in \mathbb{E}$. Now $e \equiv K(e)$.*

4 Completeness

Finally, we have every ingredient in place to reap the fruit of our labors.

Theorem 7.13 (Completeness). *Let $e, f \in \mathbb{E}$. If $[\![e]\!]_{\mathbb{E}} = [\![f]\!]_{\mathbb{E}}$, then $e \equiv f$.*

Proof. Because $[\![e]\!]_{\mathbb{E}} = [\![f]\!]_{\mathbb{E}}$, we know that $L_e([e]) = L_f([f])$. This implies that there is a bisimulation between $\hat{B}_e$ and $\hat{B}_f$ that relates $[e]$ to $[f]$. Since bisimilar states are converted into provably equivalent expressions, it then follows that $K(e) \equiv K(f)$. But then, by Theorem 7.12, $e \equiv f$. □

Some remarks are in order. First, note that the theorem we have is strictly about *equations*. There is a more general claim you can make, and we have seen it pop up in some form a number of times: the *Horn equation*, of the form

$$e_0 \equiv f_0 \wedge \cdots \wedge e_{n-1} \equiv f_{n-1} \implies e \equiv f$$

There are several of these that hold in Kleene Algebra, such as $e \cdot f \leqq g \cdot e \implies e \cdot f^* \leqq g^* \cdot e$. However, Theorem 7.13 does *not* guarantee that they are all provable. The study of *Kleene Algebra with Hypotheses* is dedicated to finding out which kinds of premises can be used to recover a completeness result.

Second, the pattern of the proof in Theorem 7.13 is actually quite common in the realm of completeness results: given that the semantics of two expressions are equivalent, first show that you can convert them into some kind of equivalent "normal form" — in this case, $K(e)$ and $K(f)$ — and then argue that the semantic equivalence of terms in normal form implies that they are provably equivalent. We see a similar pattern in, for example, the completeness proof of Boolean Algebra, or several similar results within Process Algebra.

Lastly, note that the completeness result, when composed with decidability of language equivalence, gives us a new decidability result, namely that given $e, f \in \mathbb{E}$, it is decidable whether $e \equiv f$: simply decide whether $[\![e]\!]_{\mathbb{E}} = [\![f]\!]_{\mathbb{E}}$. In effect, this gives us a road towards *proof mechanization*: if you want to prove that $e \equiv f$, you do not need to finagle with axioms, but can instead ask an algorithm to figure it out for you. This is particularly useful when you are working through a larger proof about your program, and want to verify whether a certain equivalence holds in general. Proof assistants such as Coq can be scripted to try and automatically proof statements of Kleene Algebra.

5 Bibliographic notes

There are a number of completeness results for laws about programs, including those by Salomaa [Sal66], Conway [Con71], Krob [Kro90], Boffa [Bof90], and Kozen [Koz94]. The result discussed in this lecture was first shown by Kozen [Koz94], but his tactic was rather different. The presentation we used is adapted from Jacobs [Jac06], who credits the proofs to Kozen [Koz01].

References

[Bof90] Maurice Boffa. Une remarque sur les systèmes complets d'identités rationnelles. *ITA*, 24:419–428, 1990.

[Con71] John Horton Conway. *Regular Algebra and Finite Machines.* Chapman and Hall, Ltd., London, 1971.

[Jac06] Bart Jacobs. A bialgebraic review of deterministic automata, regular expressions and languages. In *Algebra, Meaning, and Computation, Essays Dedicated to Joseph A. Goguen on the Occasion of His 65th Birthday*, pages 375–404, 2006. `doi:10.1007/11780274_20`.

[Koz94] Dexter Kozen. A completeness theorem for Kleene algebras and the algebra of regular events. *Inf. Comput.*, 110(2):366–390, 1994. `doi:10.1006/inco.1994.1037`.

[Koz01] Dexter Kozen. Myhill-Nerode relations on automatic systems and the completeness of Kleene algebra. In *STACS*, pages 27–38, 2001. `doi:10.1007/3-540-44693-1_3`.

[Kro90] Daniel Krob. A complete system of b-rational identities. In *ICALP*, pages 60–73, 1990. `doi:10.1007/BFb0032022`.

[Sal66] Arto Salomaa. Two complete axiom systems for the algebra of regular events. *J. ACM*, 13(1):158–169, 1966. `doi:10.1145/321312.321326`.