COMPUTATION: DAY 2

BURTON ROSENBERG
UNIVERSITY OF MIAMI

Contents

1. Review of the first day

For historical context, ruler and compass constructions were presented. Then the finite automata, via the Think-A-Dot toy. A mathematical description was given, as well as a graphical language for finite automata. Project 2 was assigned and discussed. The project is to create finite automata for various regular languages, running the automata description on a simulator.

2. Regular Languages Closure Properties

Friday, 27 January 2023

Consider the class of all regular languages over a common alphabet Σ. The word *class* is used here to remind us of the hierarchy — languages contain words $s \in \Sigma^*$, and a language class contains subsets $S \subseteq \Sigma^*$. That is, if S and T are recognized by a finite automata then so are the sets,

- complement,
$$S^c = \{\, s \in \Sigma^* \mid s \notin S \,\}$$

- union:
$$S \cup T = \{\, s \in \Sigma^* \,|\, s \in S \text{ or } s \in T \,\}$$
- intersection,
$$S \cap T = \{\, s \in \Sigma^* \,|\, s \in S \text{ and } s \in T \,\}$$
- concatenation,
$$S \circ T = \{\, s' s'' \in \Sigma^* \,|\, s' \in S,\, s'' \in T \,\}$$
- and Kleene star,
$$S^* = \bigcup_{i=0,1,\ldots} S^i$$

The emphasis is on that the result of the set construction is a machine for which there is a finite automata that recognizes the constructed set. Suppose then, machines M_S such that $\mathcal{L}(M_S) = S$ and M_T such that $\mathcal{L}(M_T) = T$,

$$\begin{aligned} M_S &= \langle\, Q_s, \Sigma, \delta_s, q_s^o, F_s \,\rangle \\ M_T &= \langle\, Q_t, \Sigma, \delta_t, q_t^o, F_t \,\rangle \end{aligned}$$

The case of complement is clear. If a finite automata does not halt on an accepting state then it halts on a non-accepting state. So the finite automata that recognizes S^c the finite automata M_{S^c} that is identical to M_S except F_s is replaced withm

$$F_{S^c} = Q_s \setminus F_s,$$

the complement of F_s in Q_s.

The finite automata accepting the union or the intersection of S and T is built by creating a finite automata M_p on the state set $Q_p = Q_s \times Q_t$ capable of following simultaneously in M_p the computations on the two machines M_S and M_T,

$$\begin{aligned} \delta_p : \quad Q_s \times Q_t \times \Sigma &\rightarrow Q_s \times Q_t \\ ((q_s, q_t), \sigma) &\mapsto (\delta_s(q_s, \sigma), \delta_t(q_t, \sigma)) \end{aligned}$$

The final state for the union machine is,

$$\begin{aligned} F_\cup &= \{\, (q_s, q_t) \in Q_s \times Q_t \,|\, q_s \in Q_s \text{ or } q_t \in Q_t \,\} \\ &= (F_s \times Q_t) \cup (Q_s \times F_t) \end{aligned}$$

and the final state of the intersection machine is,

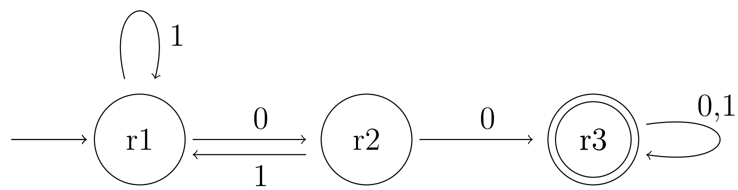
$$\begin{aligned} F_\cap &= \{\, (q_s, q_t) \in Q_s \times Q_t \,|\, q_s \in Q_s \text{ and } q_t \in Q_t \,\} \\ &= F_s \times F_t \end{aligned}$$

It was noted in this day's lecture that this *product machine construction* does not work to create machines for concatenation or Kleene star.

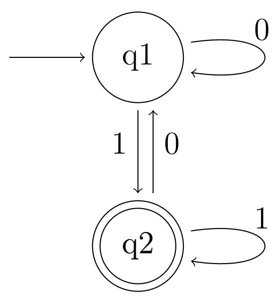
3. Example Product Machine Construction

Monday, 30 January 2023

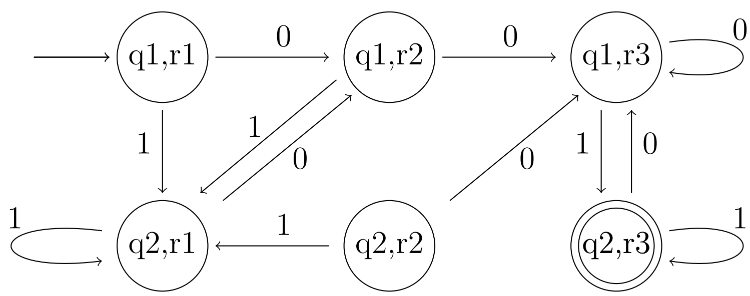
Let S_1 be the set of strings containing 00, and S_2 be the set of strings ending in a 1. The product construction is given accepting the language $S_1 \cap S_2$.



Accepts strings over $\{0, 1\}$ containing the sequence 00.



Accepts strings over $\{0, 1\}$ ending with a 1.



The product of the above two machines, with final state to accept the intersection.

4. Nondeterministic Finite Automata

Wednesday, 1 February 2023

A nondeterministic finite automate introduces two innovations to the deterministic machine,

(1) State transitions can occur spontaneously; this is modeled as a transition of the letter ε, the represents the empty string in the string algebra[1] of Σ^*. This new machine alphabet is,

$$\Sigma_\varepsilon = \Sigma \cup \{ \varepsilon \}$$

(2) The transition function takes in its values as a subset of states,

$$\delta : Q \times \Sigma_\epsilon \to \mathcal{P}(Q)$$

(3) Given an NFA N the language of the machine $\mathcal{L}(N)$ is the set of all strings s in Σ^* such that there is an equivalent string $\tilde{s}$ in Σ_ε^* and at least one computation from the start state to an accepting state on the string $\tilde{s}$.

$$\mathcal{L}(N) = \{ s \in \Sigma^* \,|\, \exists\, \tilde{s}, f : \tilde{s} \in \Sigma_\varepsilon^*, f \in F, s = \tilde{s} \text{ and } q_o \xrightarrow{\tilde{s}} f \}$$

(4) A DFA is an NFA which it just so happens the values of δ are always singleton sets, and $\delta(q, \varepsilon)$ is always the empty set, for all $q \in Q$.

5. Tree Model of Nondeterminism

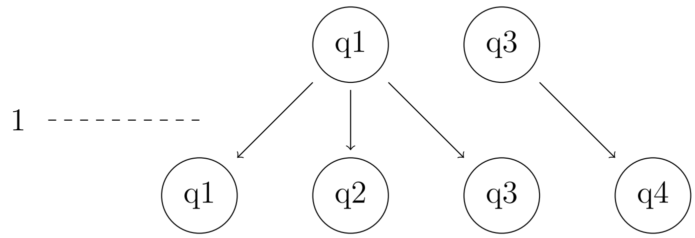
A nondeterminism machine cannot be built, as it does not resolve how the accepting computation path is found, assuming the string is in the language. Just as intriguingly, now the lack of any computation path is verified, if the string is not in the language.

In the tree model, all computation paths are searched in parallel, in an input-letter by input-letter fashion. The tree has nodes, each node tables with a state. The root of the tree is labelled with the start state. The children of a node v with label state q on letter σ are the several children each labelled with a stage from $\delta(q, \sigma)$. If the transition function is the empty set, there are no children and the computation path up to that node is not considered accepting.

Here is a sample layer of the tree for $N1$ on the input 010110, the layer of the transition on the second 1,

[1]The set Σ^* with the operation $\circ$ for sting concatentation can be seen as a calculation system where the empty string ε is the unit element, $s \circ \varepsilon = \varepsilon \circ s = s$. It is non-commutative, in general $s \circ t \neq t \circ s$ and there are no inverses, $s \circ x = t$ is solvable for x only if s is a prefix of t. Traditionally, we drop the concatenation operator and just write $s\,t$ for the concatenation of s and t.

A string $\tilde{s}$ in the extended algebra Σ_ε^* over the extended alphabet Σ_ε is similar to a string s in Σ^* if they are the same string after removing any ε that appear. With some abuse of notation we write $\tilde{s} = s$.



One layer in the tree.

6. ε-Closure

See the previous footnote about the extended alphabet Σ_ε^*. Formally it describes the string homomorphism $\phi : \Sigma_\varepsilon^* \to \Sigma^*$ which discards ε characters. Then $\tilde{s} = s$ with $\tilde{s} \in \Sigma_\varepsilon^*$ and $s \in \Sigma^*$ if $\phi(\tilde{s}) = s$. This is all big language for saying, when considering whether s is in the language, you must also try all patterns of inserting one or more ε anywhere into the string.

However the tree model does not show any ε steps. Every step is associated with a character in Σ. The rule applied is that, on entering a state, follow all ϵ transitions and add those states among the children. This rule is applied repeatedly, in case the new states also have ϵ transitions.

A set of states $T \subseteq Q$ is ε-closed if for all $t \in T$, and all possible computations $t \xrightarrow{\varepsilon} t'$, then $t \in T$. The ε-closure of a set S is the smallest possible set T such that $S \subseteq T$ and T is ε-closed.

Each level in the tree model includes all the states in the ε-closure of $\delta(q, \sigma)$.

7. Oracle Model of Nondeterminism

Friday, 3 February 2023

The oracle model was demonstrated using machine $N2$ as the second example. The oracle model can also be described as a guess and verify model.

The nondeterministic approach is summarized as a game between Arthur and Merlin, from the legends of the Knights of the Round Table. Merlin is a wizard assisting Arthur.[2] When Arthur confronts multiple outgoing transition in the NFA, Arthur asks Merlin which transition to take. We do not know the source of Merlin's wisdom, just that his advice is prompt and perfect.

[2]Arthur-Merlin games were introduced by László Babai in the paper *Trading group theory for randomness* (1985). This line of research lead to Zero Knowledge prof systems, which are the mathematics which fuel such technologies as Z-Cash.

Consider a regular language S over Σ^* and a $s \in \Sigma^*$. For $s \notin S$ there are no accepting paths. For $s \in S$ there are also at least one accepting path, but there maybe also be non-accepting paths.

(1) For $s \in S$, Merlin is *magical* in that he can answer all questions to lead Arthur along a computation path to an accepting state. He is also *honest* in that he will.
(2) For $s \notin S$, Merlin is neither honest nor dishonest, neither magical nor muggle, as he answers randomly and the result is a computation path that does not lead to an accepting state.
(3) Although Merlin's ability to answer is magical, the answer is not. If $s \in S$, then the collection of Merlin's answers can be used by any muggle to rerun the computation and witness that $s \in S$.
(4) Likewise, if $s \notin S$, then Merlin cannot magically make is so — no pattern of answers by Merlin will compute out in an accepting computation.
(5) If Arthur were to banish Merlin, and cast aspersions on his advice, Merlin's answers for $s \in S$ remain convincing, but his answers for $s \notin S$ show very little either of membership or non-membership.

Nota bene, problem 1.38 of the class textbook, the definition of the the all–NFA. This is a machine of equivalent power to the NFA but flips the script on accepting versus non-accepting evidence.

8. Closure properties by Oracle

Monday, 6 February 2023

The produce construction is not able to show that regular languages are closed by concatenation. Here is an example where the construction fails.

On the alphabet $\{0\}$, let $S_k = (0^k)^*$, strings with length a multiple of k. Consider the language,

$$S_3 \circ S_5 = 0^* \setminus \{0, 0^2, 0^4, 0^7\}$$

The product construction gives a machine with 15 states. Let q be the final state for the string 00. It is also the final state for the string 0^{17}. However $00 \notin S_3 \circ S_5$ and 0^{17} is.

Let M_1 and M_2 be FA for two languages. If $s \in \mathcal{L}(M_1) \circ \mathcal{L}(M_2)$ then $s = s_1\, s_2$ such that $s_1 \in \mathcal{L}(M_1)$ and $s_2 \in \mathcal{L}(M_2)$. Merlin instructs Arthur to follow an ε transition from an accepting final state after the computation to the initial state of M_2, so the computation is,

$$q_1^o \xrightarrow{s_1} f_1 \xrightarrow{\varepsilon} q_2^o \xrightarrow{s_2} f_2$$

where f_1 and f_2 are in the final states of M_1 and M_2, respectively and q_1^o and q_2^o are the respective start states of M_1 and M_2. For strings that cannot be so written, Merlin instructs Arthur randomly.

If $s \in \mathcal{L}(M_1)^*$ then $s = s_1 \, s_2 \, \ldots \, s_k$ such that $s_i \in \mathcal{L}(M_1)$ for every $i = 1, 2, \ldots, k$. Merlin instructs Arthur to take ε transitions from the accepting state found at the conclusion of the string up ti the send of s_i to the start state, if $i < k$. Hence the accepting computation is,

$$q_1^o \xrightarrow{s_1} f_1 \xrightarrow{\varepsilon} q_1^o \xrightarrow{s_2} f_2 \xrightarrow{\varepsilon} \ldots \xrightarrow{\varepsilon} q_1^o \xrightarrow{s_k} f_l$$

for some final states f_i of M_1. For strings that cannot be so written, Merlin instructs Arthur randomly.

For the union, Merlin instructs Arthur to take an ε transition to the start state of which ever machine accepts the string, if either does; and randomly if neither does.

9. Closure properties concluded, constructions

(1) Given machines M_1 and M_2 create the machine M_3 for which,

$$\mathcal{L}(M_3) = \mathcal{L}(M_1) \circ \mathcal{L}(M_2)$$

by the machine[3],

$$M_3 = \langle\, Q_1 \sqcup Q_2, \Sigma, \delta_1 \sqcup \delta_2 \sqcup \delta_b, q_1^o, F_2 \,\rangle$$

where,

$$\delta_b(q, \varepsilon) = q_2^o, \text{ for all } q \in F_1.$$

(2) Given machine M_1 create the machine M_4 for which,

$$\mathcal{L}(M_4) = \mathcal{L}(M_1)^*$$

by the machine,

$$M_4 = \langle\, Q_1 \sqcup \{q_4^o\}, \Sigma, \delta_1 \sqcup \delta_l, q_4^o, F_1 \sqcup \{q_4^o\} \,\rangle$$

where,

$$\delta_l(q, \varepsilon) = q_1^o, \text{ for all } q \in F_1 \sqcup \{q_4^o\}$$

(3) Given machines M_1 and M_2 create the machine M_5 for which,

$$\mathcal{L}(M_5) = \mathcal{L}(M_1) \cup \mathcal{L}(M_2)$$

by the machine,

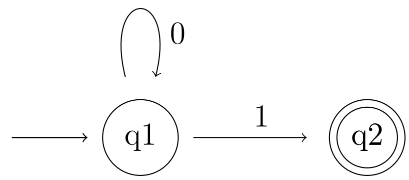
$$M_5 = \langle\, Q_1 \sqcup Q_2 \sqcup \{q_5^o\}, \Sigma, \delta_1 \sqcup \delta_2 \sqcup \delta_u, q_5^o, F_1 \sqcup F_2 \,\rangle$$

where,

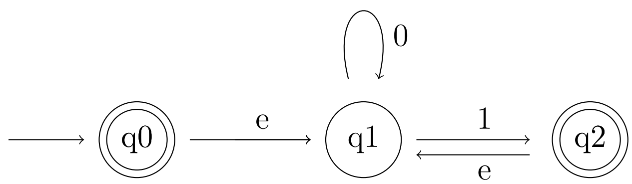
$$\delta_u(q_5^o, \varepsilon) = \{q_1^o, q_2^o\}.$$

Nota bene: The new state is needed in the Kleene star construction. For instance, the language 0^*1 is accepted by a two state machine,

[3] The simbol $\sqcup$ is the disjoint union. It assumes (without loss of generality by relabeling) that the two sets to be joined are disjoint

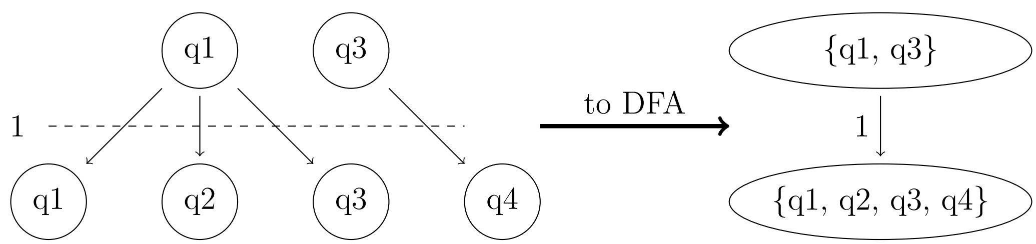


To accept the empty string include in $(0^*1)^*$ without accepting strings ending in a zero which are not included in $(0^*1)^*$, a new start state is needed.



10. NFA's accept exactly the regular sets

Since every DFA is an NFA, the class of languages accepted by and NFA is at least those accepted by a DFA. Any NFA can be made a DFA by created a machine on the power set of the state set of the NFA, and attaching arrows to states, rather than sets of state. In the tree model, at each level the set of states seen across the level is now a single state; and the multiple arrows connecting levels, is a single arrow.



Wednesday, 8 February 2023

On this day was given an example of the general construction of turning an NFA into DFA. This included review of the ϵ-closure operation.

Regular Expressions were introduced, briefly.