COMPUTATION: DAY 3

BURTON ROSENBERG
UNIVERSITY OF MIAMI

Contents

1. Review of the second day

Nondeterminism was introduced, including Arthur-Merlin Oracles, and parallel-worlds models. This allowed the completing easily the proof of the closure properties for Regular Sets: they are closed under complementation, union, intersection, concatenation and Kleene star.

2. Regular Expressions

Friday, 10 February 2023

A regular expression is built from elements of an alphabet Σ, the special symbols ε and $\emptyset$, and the operations union, $\cup$, concatenation $\circ$ and Kleene star, $*$, along with parenthesis sufficient to make the order of operations clear.

The interpretation of a regular expression R is a set $\mathcal{L}(R) \subseteq \Sigma^*$, and we shall see that only and all regular sets are given by regular expressions. The interpretation is a letter is single set of that letter, ε the empty string, $\emptyset$ the empty set, and union to union of sets, concatenation the concatenation of sets and Kleene star the Kleene star of the set.

$$\begin{array}{rcl}
\mathcal{L}((ab \cup c)^*) & = & (\mathcal{L}(ab \cup c))^* \\
& = & (\mathcal{L}(ab) \cup \mathcal{L}(c))^* \\
& = & (\mathcal{L}(a) \circ \mathcal{L}(b) \cup \{c\})^* \\
& = & (\{a\} \circ \{b\} \cup \{c\})^* \\
& = & \{a\,b \cup c\}^*
\end{array}$$

The precedence order of operations has an analogy with the precedence of arithmetic operations,

$$\begin{array}{rcl}
a\,b \cup c & = & \{a\,b, c\}, \\
a\,(b \cup c) & = & \{a\,b, a\,c\} \\
a\,b^* & = & \{a, ab, abb, \ldots\}
\end{array}$$

hence $\cup$ has the precedence of $+$, $\circ$ has the precedence of $\times$, and $*$ has the precedence of a power.

The empty set is the identity element for union and the annihilator for concatenation, $\forall R \subseteq \Sigma^*$ (think 0),

$$R \circ \emptyset = \emptyset \text{ and } R \cup \emptyset = R.$$

The empty string is the identity element for concatenation, $\forall R \subseteq \Sigma^*$ (think 1),

$$R \circ \varepsilon = R.$$

The star operation is a power series,

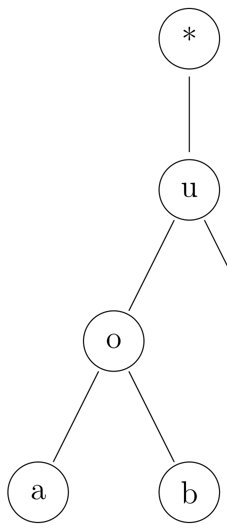
$$S^* = \varepsilon\,(1 + S + S^2 + \ldots) = \frac{\varepsilon}{1 - S}$$

and therefore.

$$\emptyset^* = \frac{\varepsilon}{1 - \emptyset} = \varepsilon$$

3. All Regular Expressions are Regular Languages

Monday, 13 February 2023



The parsing of regular expression $(((a \circ b) \cup c))^*$.

A regular expression can be parsed into a tree where the nodes of the tree are labeled with the operations $\cup, \circ$ or $*$, or with the basic characters $\sigma \in \Sigma, \varepsilon$ or $\emptyset$. Nodes labeled with $\cup$ and $\circ$ will have two children, distinguished as a left child and a right child; nodes labeled $*$ will have on child, and the other nodes are leaves of the tree.

A node in this tree can be replaced by a NFA, provided that the labels on the nodes of all children of the node have already been replaced with NFA's. The leaf node NFA's are acceptors for a single letter language, for the language of the empty string or the empty language. The other nodes use the constructions we have already described for star, union and concatenation.

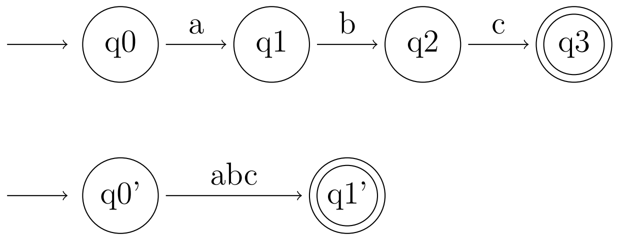
Ultimately the root node is replaced by an NFA. That NFA accepts exactly the language of the regular expression.

4. The Generalized NFA

Given an NFA, a Regular Expression is constructed that accepts exactly the language of the NFA.

To do this, a Generalized NFA (GNFA) is proposed. The edges of a GFNA can be RE's. This is a great convenience in general conversation. For example, the 4 state machine accepting the $\{abc\}$ can be replaced by a two state machine.

There are some other requirements for a GNFA which are there specifically to make this proof work out neatly,



The helpfulness of a Generalized NFA.

(1) There is only one final state and this state has no outgoing transitions. Achieve this in general with a dedicated final state q_f to which all candidate final states transition with a ε-move.
(2) The start state has no incoming transitions. Achieve this in general with a dedicated start state q_s that transitions to by ε-move to a start state that would have had an incoming transition.
(3) Let q_i and q_j be any two not necessarily distinct states other than q_s and q_f. There are transitions, $q_s \to q_i$, $q_i \to q_f$, and $q_i \to q_j$. Achieve this in general with transitions labeled with the regular expression $\emptyset$.

5. All Regular Languages are Regular Expressions

The plan:

(1) Any NFA can be turned into a GNFA.
(2) Any GNFA with k states, $k > 2$ can be turned into an GNFA with $k-1$ states that accepts exactly the same language.
(3) A GNFA with exactly 2 states has a unique edge $q_s \to q_f$ whose label is a regular expression for the language.

Wednesday, 15 February 2023

The conversion of a DFA to a GNFA and then to a RE was presented (figure 1.67 in the class text).

6. Context Free Grammars

We did this informally in class, and will do it formally a bit later. Here is the formal version.

A *Context Free Grammar* is a mathematical complex $\langle V, T, R, S\rangle$ where,

- V is a finite set of *variables*, also called *non-terminals*,
- T is a finite set of *terminals*,
- R is a set of *rules*, and is a subset of $V \times (V \cup T)^*$,

- and $S \in V$ is the start symbol.

What you do with a CFG is repeatedly apply a rule to a string $s \in (V \cup T)^*$ according to,

$$wXv \text{ and } (X, u) \in R \implies wuv$$

Write this as $wXv \to wuv$ and consider the reflective transitive closer of $\to$, by abuse of notation also denoted $\to$. Then $S \to w$ where $w \in T^*$ is in the language of the grammar.

Denote by $\mathcal{L}(G)$ all possible such w, as the language of the grammar.

$$\mathcal{L}(G) = \{\, w \in \Sigma^* \mid S \to w \text{ by the grammar } G \,\}$$

A *Context Free Grammar* is a any subset of $\mathcal{G} \subseteq T^*$ for which there is a grammar G and $\mathcal{G} = \mathcal{L}(G)$.

7. Example Context Free Grammers

7.1. Regular Expressions. The language of Regular Expressions over the alphabet Σ,

$$\begin{array}{rcl} V & = & \{S\} \\ T & = & \{\circ, \cup, *, \varepsilon, \emptyset\} \cup \Sigma \\ S & \mapsto & (S \circ S) \mid (S \cup S) \mid (S)^* \mid \varepsilon \mid \emptyset \mid \sigma \in \Sigma \end{array}$$

7.2. Balanced Parentheses. The language of balanced parentheses,

$$\begin{array}{rcl} V & = & \{S\} \\ T & = & \{(,)\} \\ S & \mapsto & S\,(S) \mid \varepsilon \end{array}$$

8. The Pumping Lemma

Friday, 17 February 2023

Theorem 8.1 (The pumping lemma)**.** If $\mathcal{M}$ is a regular language, and M is a machine recognizing that language, $\mathcal{L}(M) = \mathcal{M}$, then every sufficiently long string in $\mathcal{M}$ can be pumped.

There are two things to clarify,

(1) There are many different machines M that all recognize $\mathcal{M}$. A string is sufficiently long is it is longer than the number of states in M. This is sort of an odd thing, because you can propose a sequence of machines each with more states than the last, each accepting the same language.

(2) To pump $s \in \mathcal{M}$ is to find a looping part y somewhere not too far from the start of s. That is, to write $s = xyz$ so that,

$$|xy| < p,\ y \neq \varepsilon \text{ and } \forall i \geq 0, xy^i z \in \mathcal{M}.$$

Note well, s must be in the language. If the language is finite it is regular. However the pumping lemma will not apply, because the set of sufficiently long strings can be made to be empty.

9. Proving a language non-regular

Monday, 20 February 2023

The pumping lemma is used in the contrapositive to show that some languages are not regular. For instance, this language is not regular,

$$\mathcal{B} = \{0^k 1^k \mid k \geq 0\}$$

The use of the pumping lemma to prove a language is not regular, requires taking the negation of the theorem, which poses reversals of *for all* and *for some*. This can be confusing.

(1) Since for a regular language *for some* machine M, $\mathcal{L}(M) = \mathcal{M}$, to show a language is not regular *for all* machines M, $\mathcal{L}(M) \neq \mathcal{M}$; hence *for some* p, becomes *for all* p when consider whether $|s| > p$ is sufficiently long.
(2) Since for a regular language, *for all* $s, |s| > p$, s succeeds when pumped, to show a language is not regular it is sufficient that *for some* $s, |s| > p$, s fails when pumped.
(3) Since for regular language $xy^i z$ succeeds *for some* splitting $s = xyz$, to show a language is not regular $xy^i z$ fails *for all* splittings of $s = xyz, |s| > p > |xy|$ and $y \neq \varepsilon$.
(4) Given an $s = xyz$, since for a regular language *for all* values of i, $xy^i z \in \mathcal{M}$, to show a language is not regular find *for some* i, that $xy^i z \notin \mathcal{M}$.

The alternation *for some* and *for all* are like a two person game, between the *for all* player For All and the *for some* player For Some.

For the language $\mathcal{B} = \{0^i 1^i \mid i = 0, 1, 2, \ldots\}$,

(1) The For All, trying to cover all possible cases, chooses a very large p.
(2) The For Some, trying to isolate a particular situation, proposes a specific string that can show this: $w = 0^p 1^p \in \mathcal{B}$.
(3) The For All, who is allowed all possible splittings, chooses a splitting $w = xyz$ that will show the looping, if possible. In this case, since $|wy| < p$ the adversary is constrained to picking k_1, k_2, k_3 with $k_2 > 0$ and,

$$x = 0^{k_1},\ y = 0^{k_2},\ z = 0^{k_3} 1^{k_1+k_2+k_3}$$

(4) The For Some proves that For All cannot avoid the conclusion of not regular, and the advocate will pump as
$$xy^2z \notin \mathcal{B}.$$

This also proves that the language $\mathcal{D}$,
$$\mathcal{D} = \{\, w \in \{\,0,1\,\}^* \,|\, \text{the number of 1s equals the number of 0s}\,\}$$
is also regular.

The For Some player should choose 0^p1^p because they are allowed to, and it works. They should chose $w- = (01)^p$, because the For All can break this string up with $y = 01$. Then pumping will just give $(01)^{p+i}$, which is still in the language. This does not show that For All is evil, but that since it is playing *for all* it is obliged to find the contraction to For Some's logic.

For the language $\mathcal{E}$,
$$\mathcal{E} = \{\, 0^i1^j \,|\, i > j\,\}$$

(1) For All provides a p.
(2) For Some proposes $w = 0^{p+1}1^p \in \mathcal{E}$. This string is crafty.
(3) The For All finds any (or all) $w = xyz$. Since $|wy| < p$ we have,
$$x = 0^{k_1},\ y = 0^{k_2},\ z = 0^{k_3}1^{k_1+k_2+k_3-1}$$
This splitting is completely general.
(4) The For Some choses specification to pump down, that is removes the loop: $xz \notin \mathcal{E}$, hence the language is not regular.

10. Properties of Context Free Languages

The languages proved non-regular are context free. However, all regular languages are context free. Hence we have the proper containment,
$$\text{RegL} \subset \text{CFL}$$
If the Pushdown Automata (PDA) is introduced, and its equivalence to CFG's shown, then the containment is obvious. A PDA is a NFA with a stack store. If the stack store is not used, the PDA is exactly a NFA. The containment can also be shown by noting that the CFG for a RE gives a construction for realizing the CFG for the language (not for the RE describing the language).

However we do it directly. Given a Regular Language $\mathcal{L}$, let M be a DFA recognizing $\mathcal{L} = \mathcal{L}(M)$.

(1) For each each state $r \in Q$, in the state set of M, define a variable $R \in V$ in the CFG.
(2) For each transition $\delta(r, \sigma) = r'$, define a grammar rule $R \to \sigma R'$.
(3) For each final state $f \in F$, define a grammar rule $F \to \varepsilon$.

(4) For the start state r_o of M, add the rule $S \to R_o$, with S a fresh variable.

The CFL's are closed under: union, concatenation and star. If L_1 and L_2 are CFL's, let G_1 and G_2 be their grammars.

- For *union*, create a new grammar G_3 which is the disjoint union of G_1 and G_2 (renaming variables if needed), adding a new start variable with two rules, transitioning to the start states of either grammars.
- For *concatenation* proceed similarly, except the new start state transitions to a variable pair, the start variable for G_1 followed by that of G_2.
- For *star*, proceed similarly, except the new start state transitions to either ε or to the start variable for G_1 followed by the new start symbol.

However CFL's are not closed under complementation and intersection. Given a string s of length n it is possible to try all possible derivations that lead to strings of length n or shorter, and if none of these give s, then s is not in the language. However, no single derivation shows that s is not in the language. Compare this to regular languages. The NFA can be made a DFA, and after n steps, arriving at a non-final state is proof that s is not in the language. However, the role of a DFA is crucial, and there is no such device for CGL's.

11. The Pumping Lemma for Context Free Languages

Theorem 11.1. Given a CFL $\mathcal{L}$, there exists a p such that for all $s \in \mathcal{L}$, $|s| \geq p$, then $s = uvxyz$ such that,

(1) $|vxy| < p$,
(2) $vy \neq \varepsilon$
(3) $\forall i, uv^ixy^iz \in \mathcal{L}$.

The proof is by considering the parse tree. Any path of depth exceeding the number of variable in a grammar for the language will have a repeated variable on the path. The section between occurrences of this variable can be removed or repeated, giving the pumped versions of the string.

Therefore, it is possible to show that

$$\mathcal{A} = \{\, a^ib^ic^i \mid i \geq 0 \,\}$$

is not context free. However both

$$\mathcal{B} = \{\, a^ib^ic^j \mid i, j \geq 0 \,\}$$

and

$$\mathcal{C} = \{\, a^jb^jc^j \mid i, j \geq 0 \,\}$$

are context free. Since $\mathcal{A} = \mathcal{B} \cap \mathcal{C}$, this demonstrates that intersections of context free languages are not necessarily context free.

The language,

$$\mathcal{D} = \{ww \mid w \in \{0,1\}^* \}$$

is not context free. The string to demonstrate this is $w = 0^p 1^p 0^p 1^p$. Any way this is pumped, the result will not be of the form ww.

However, the language,

$$\mathcal{E} = \{wv \mid w, v \in \{0,1\}^* \, |w| = |v|, w \neq v, \}$$

is context free. It is generated by,

$$\begin{aligned} S &\rightarrow AB \mid BA \\ A &\rightarrow 0A0 \mid 1A0 \mid 0A1 \mid 1A1 \mid 0 \\ B &\rightarrow 0B0 \mid 1B0 \mid 0B1 \mid 1B1 \mid 1 \end{aligned}$$

The proof that this is the correct language depends on this trick,

$$\sigma^i 0 \sigma^i \sigma^j 1 \sigma^j = \sigma^i 0 \sigma^{i+j} 1 \sigma^j = \sigma^i 0 \sigma^j \; \sigma^i 1 \sigma^j$$

The complement of $\mathcal{D}$ is,

$$\mathcal{E} \cup \{ \, s \mid \text{ the length of } s \text{ is odd} \}$$

the union of to CFL's hence a CFL.