FINITE-STATE MACHINES LEFTOVERS

(1) NONREGULAR LANGUAGES
(2) REGULAR EXPRESSIONS

NON-REGULAR LANGUAGES

NOT EVERY LANGUAGE IS REGULAR

E.G. $\{a^n b^n \mid n \geq 0\} = \{\varepsilon, ab, aabb, aaabbb, \ldots\}$

$\{a^n b^n c^n \mid n \geq 0\}$

$\{a^m b^n \mid m \geq n \geq 0\}$

$\{w \mid \#_a(w) = \#_b(w)\}$

→ INTUITION: FINITE-STATE MACHINES CANNOT REMEMBER AN ARBITRARY NATURAL NUMBER

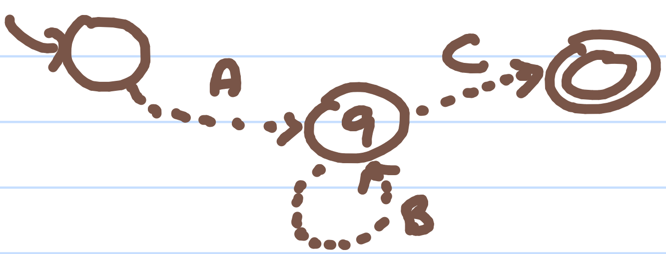
ARGUMENT: $\{a^n b^n \mid n \geq 0\}$ IS NOT REGULAR.

BY CONTRADICTION: SUPPOSE IT WERE. THEN THERE IS A DETERMINISTIC FINITE AUTOMATON M THAT ACCEPTS IT.

LET K BE THE NUMBER OF STATES IN M.

CONSIDER THE STRING $a^{k+1}b^{k+1}$ — IT IS IN THE LANGUAGE SO IT IS ACCEPTED BY M. IF YOU LOOK AT THE STATES M GOES THROUGH WHILE PROCESSING THE a^{k+1} PART OF THE STRING, IT MUST GO THROUGH SOME STATE q AT LEAST TWICE — a^{k+1} HAS $k+1$ SYMBOLS, AND THERE ARE ONLY k STATES IN TOTAL.

SO M HAS A PATH OF THE FORM



WHERE A GOES THROUGH STATES VIA SYMBOLS a^{n_1}, B GOES THROUGH STATES TO ACCEPT a^{n_2}, AND C GOES THROUGH STATES TO ACCEPT $a^{n_3}b^k$ AND $n_1+n_2+n_3=k$, AND $n_2>0$

BUT SINCE YOU CAN GO FROM q TO q FOLLOWING a^{n_2} YOU CAN ALSO GO THROUGH THE LOOP TWICE, AND THUS M MUST ALSO ACCEPT $a^{n_1}a^{n_2}a^{n_2}a^{n_3}b^k$

But that string has $n_1 + 2n_2 + n_3 > k$ a's and k b's, so is not in the language. This contradicts M accepting the language. So our initial assumption, that there is a DFA accepting the language is false, that is, the language is not regular. $\square$

Regular Expressions

I mentioned regular expressions in passing. Let's dig a bit into them.

Regular expressions are a convenient notation for regular languages. They are often use as the basis of search patterns over text.

> A search pattern describes a set of strings, all the strings <u>matching</u> the pattern. So a search pattern describes a language. You can think of checking if a string matches a pattern as the same as checking if the string is in the language described by the pattern.

A **REGULAR EXPRESSION** OVER ALPHABET $\Sigma = \{a_1, \ldots, a_k\}$ IS AN EXPRESSION WRITTEN VIA THE FOLLOWING SYNTAX:

$$R ::= 1 \mid 0 \mid a_1 \mid \ldots \mid a_k \mid R \cdot R \mid R \cup R \mid R^* \mid (R)$$

E.G. $\Sigma = \{a, b\}$

- $1 \cup a \cup b \cup ab$
- $a^* b^*$
- $a(a \cup b)^* a$

CAN DROP THE $\cdot$

SOME AUTHORS USE + INSTEAD OF $\cup$

PRECEDENCE:

$* \quad \cdot \quad \cup$

(USE PARENS TO CHANGE)

WE ASSOCIATE WITH EACH REGULAR EXPRESSION A LANGUAGE DESCRIBED BY THE REGULAR EXPRESSION, $L[\![R]\!]$

→ PER EARLIER, YOU CAN THINK OF IT AS THE SET OF ALL STRINGS THAT **MATCH** THE REGULAR EXPRESSION.

WE DEFINE THE LANGUAGE DESCRIBED BY A REGULAR EXPRESSION BY RECURSION OVER THE STRUCTURE OF REGULAR EXPRESSIONS:

$$L[\![1]\!] = \{\varepsilon\}$$

$$L[\![0]\!] = \emptyset$$

$$L[\![a_i]\!] = \{a_i\}$$

$$L[\![R_1 R_2]\!] = L[\![R_1]\!] \cdot L[\![R_2]\!]$$

$$L[\![R_1 \cup R_2]\!] = L[\![R_1]\!] \cup L[\![R_2]\!]$$

$$L[\![R_1^*]\!] = (L[\![R_1]\!])^*$$

E.G. $L[\![a(a\cup b)^* b]\!]$

$$= L[\![a]\!] \cdot L[\![(a\cup b)^*]\!] \cdot L[\![b]\!]$$

$$= L[\![a]\!] \cdot L[\![a\cup b]\!]^* \cdot L[\![b]\!]$$

$$= L[\![a]\!] \cdot (L[\![a]\!] \cup L[\![b]\!])^* \cdot L[\![b]\!]$$

$$= \{a\} \cdot (\{a\} \cup \{b\})^* \cdot \{b\}$$

$$= \{a\} \cdot \{a, b\}^* \cdot \{b\}$$

$$= \{awb \mid w \in \{a, b\}^*\}$$

THEOREM: A LANGUAGE A IS REGULAR IF AND ONLY IF THERE EXISTS A REGULAR EXPRESSION R WITH $L[\![R]\!] = A$.

PROPERTIES OF REGULAR LANGUAGES

- IF A IS FINITE, THEN A IS REGULAR
- $\emptyset$ IS REGULAR
- Σ^* IS REGULAR
- IF A AND B ARE REGULAR, THEN $A \cdot B$ AND $A \cup B$ ARE BOTH REGULAR
- IF A IS REGULAR, THEN A^* IS REGULAR

MORE INTERESTING:

- IF A IS REGULAR, $\overline{A}$ IS REGULAR (USE COMPLETE DFA)

- IF A IS REGULAR, $REV(A)$ IS REGULAR WHERE $REV(A) = \{REV(w) \mid w \in A\}$ (USE REGULAR EXPRESSIONS)