MA/CSSE 474

Theory of Computation

Computation
FSM Intro

Trying Another PDA

A PDA to accept strings of the form:

$A^nB^nC^n = \{a^nb^nc^n : n \geq 0\}$

Q1

Turing Machines

A read/write head and a tape that is infinite in both directions.

Based on current state and symbol it sees on the tape, it writes a symbol (possibly the same one) to the tape and moves one position left or right.

If it reaches an accepting state or rejecting state, it halts.

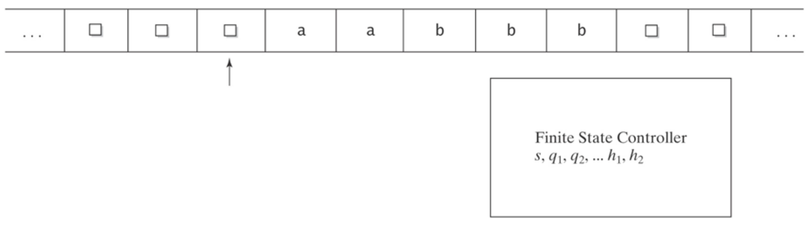
The (finite) input string is originally written consecutively on a portion of the tape; the rest is initially blank, but the read/write head can write things on any square.

At any given time, only a finite part of the tape is non-blank.

The head starts at the square to the left of the first input symbol.

Turing Machines

A Turing Machine to accept $A^nB^nC^n$:



Q2

Decidable and Semidecidable Languages

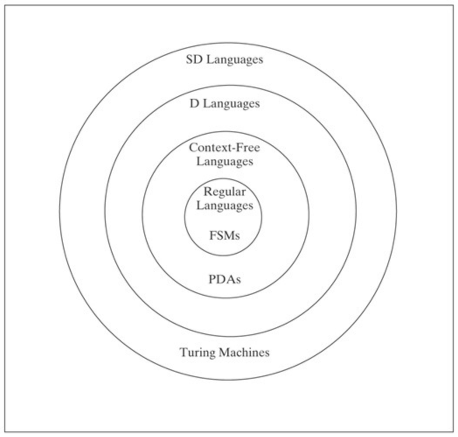
- A language L is **decidable** iff there exists a Turing machine M that halts on all inputs, accepts all strings that are in L, and rejects all strings that are not in L.
  - In other words, M can always say yes or no, as appropriate.

Decidable and Semidecidable Languages

- A language L is **semidecidable** iff there exists a Turing machine M that accepts all strings that are in L and does not accept any string that is not in L.
  - Given a string that is not in L, M may reject or it may loop forever.
  - M can always recognize a string in L and say yes,
    - but it may not know when it should give up looking for a solution and say no.
- A language L is **undecidable** iff it is not semidecidable.

Q3

Languages and Machines



Rule of Least Power: "Use the least powerful language suitable for expressing information, constraints or programs on the World-wide web."
--Tim Berners-Lee and Noah Mendelsohn(2006)

Some "Canonical" Languages

- $A^nB^n = \{a^nb^n : n >= 0\}$
- Bal = { strings of balanced parentheses}
- $WW = \{ww : w \in \Sigma^*\}$
- PalEven $\{ww^R : w \in \Sigma^*\}$
- $A^nB^nC^n = \{a^nb^nc^n : n >= 0\}$
- HP_{ALL} = {<T> : T is a Turing machine that eventually halts, no matter what input it is given}
- PRIMES = {*w* : *w* is the binary encoding of a prime integer}

Nondeterminism

- A **nondeterministic** machine in a given state, looking at a given symbol (and with a given symbol on top of the stack if it is a PDA), has a choice of multiple possible moves that it can make.
- If there is a move that leads toward acceptance, it makes that move.

Nondeterminism

- Given a string in {a, b}*, is it in
  PalEven = $\{ ww^R : w \in \{a,b\}^* \}$?
- PDA
- **Choice:** Continue pushing, or start popping?
- This language can be accepted by a nondeterministic PDA but not by any deterministic one.

Nondeterministic value-added?

- Ability to recognize additional languages?
  - FSM: no
  - PDA : yes
  - TM: no

We will prove these later

- Ease of designing a machine for a particular language
  - Yes in all cases

Sample Decision Problems

Example: Given two strings *s* and *t*, does *s* occur anywhere as a substring of *t*?

It's easy to see that this is decidable.

Q4

Sample Decision Problems Prime Fermat Numbers

Fermat numbers: $F_n = 2^{2^n} + 1$, $n \geq 0$

$F_0 = 3$, $F_1 = 5$, $F_2 = 17$, $F_3 = 257$, $F_4 = 65{,}537$,
$F_5 = 4{,}294{,}967{,}297$

- Are there any prime Fermat numbers less than 1,000,000?
- Are there any prime Fermat numbers greater than 1,000,000?

Q5

Sample Decision Problems

Given a Java program *P*, is there some input string on which *P* halts?

Given a Java program *P*, and a value *v*, is *v* the shortest input string on which *P* halts?

Functions on Languages

Functions whose domains and ranges are languages

$maxstring(L) = \{w \in L: \forall z \in \Sigma^* (z \neq \varepsilon \rightarrow wz \notin L)\}$.

Examples:

- *maxstring*(A^nB^n)
- *maxstring*($\{a\}^*$)

Let INF be the set of infinite languages.
Let FIN be the set of finite languages.

Are the language classes FIN and INF closed under *maxstring*? Q6

Functions on Languages

$chop(L) =$
$\{w : \exists x \in L\ (x = x_1cx_2,\ x_1 \in \Sigma_L^*,\ x_2 \in \Sigma_L^*,\ c \in \Sigma_L,$
$|x_1| = |x_2|, \text{ and } w = x_1x_2)\}$.

What is *chop*(A^nB^n)?

What is *chop*($A^nB^nC^n$)?

Are FIN and INF closed under *chop*?

If we don't have much time left, we'll skip this in class

Functions on Languages

firstchars(L) =
$\{w : \exists y \in L\ (y = cx \wedge c \in \Sigma_L \wedge x \in \Sigma_L^* \wedge w \in \{c\}^*)\}$. .

What is *firstchars*(A^nB^n)?

What is *firstchars*({a, b}*)?

Are FIN and INF closed under *firstchars*?

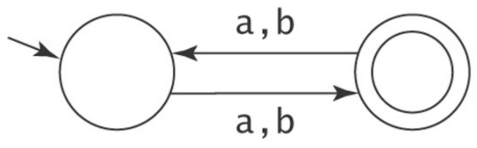
If we don't have much time left, we'll skip this in class

Representing Languages as Machines

Compute union using descriptions like:

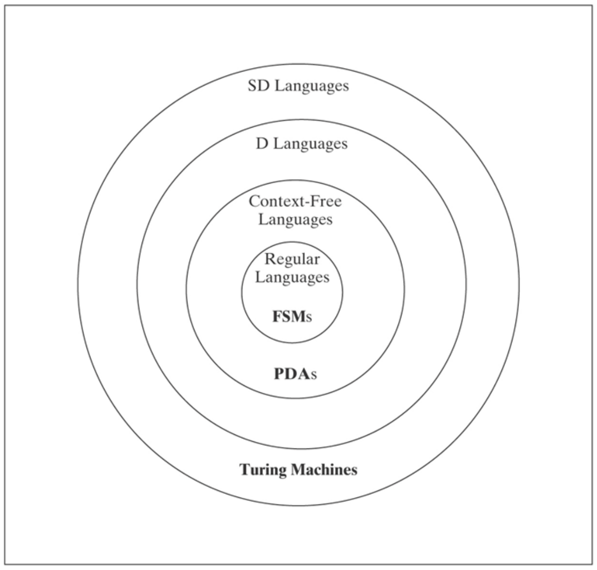
- {*w* ∈ {a, b}* : *w* has odd length}
- {*w* ∈ {a, b}* : all a's in *w* precede all b's}

Compute union using descriptions like:

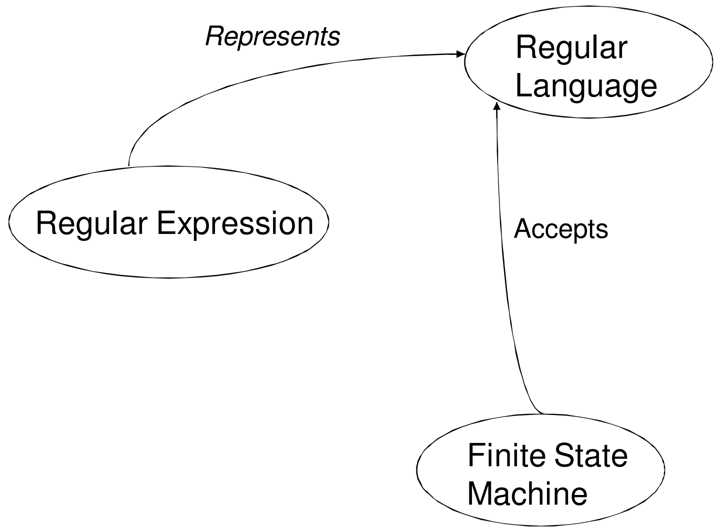


Finite State Machine Intro

Languages and Machines

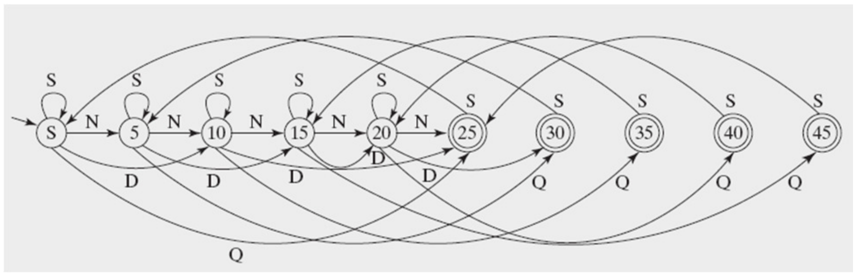


Regular Languages



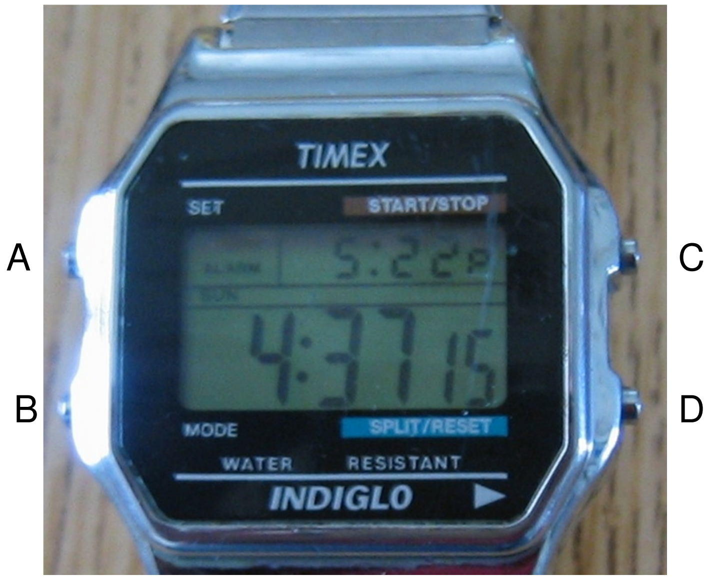
Finite State Machines

An (ancient) Soda Machine FSM to accept $.25 in change:



(Note that a couple of the dime transitions are incorrect, but you get the idea).

Another FSM Example



Definition of a DFSM

$M = (K, \Sigma, \delta, s, A)$, where:

The D is for Deterministic

K is a finite set of ***states***

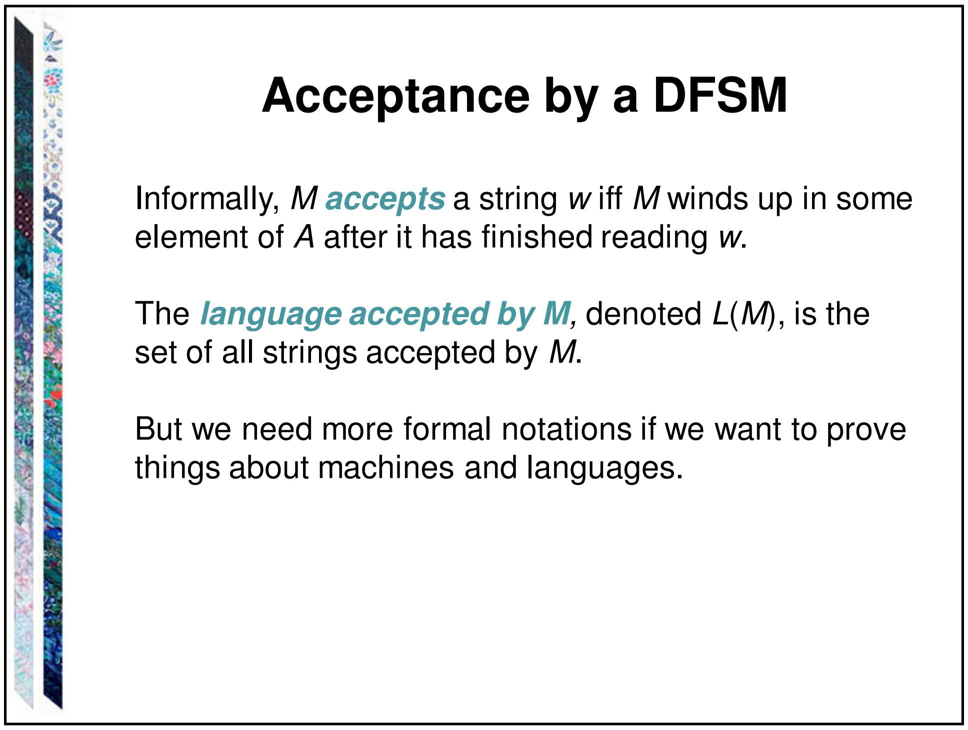
Σ is a (finite) ***alphabet***

$s \in K$ is the ***initial state*** (a.k.a. start state)

$A \subseteq K$ is the set of ***accepting states***

$\delta: (K \times \Sigma) \rightarrow K$ is the ***transition function***

Sometimes we will put an M subscript on K, Σ, δ, s, or A (for example, s_M), to indicate that this component is part of machine M.

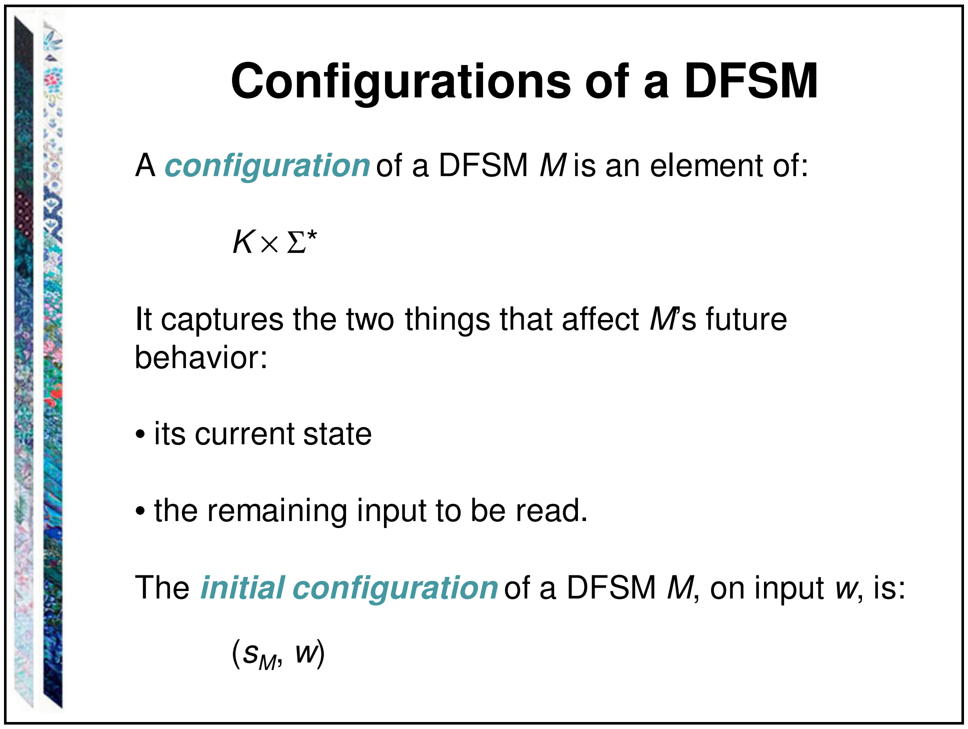


Acceptance by a DFSM

Informally, *M* ***accepts*** a string *w* iff *M* winds up in some element of *A* after it has finished reading *w*.

The ***language accepted by M,*** denoted *L*(*M*), is the set of all strings accepted by *M*.

But we need more formal notations if we want to prove things about machines and languages.



Configurations of a DFSM

A ***configuration*** of a DFSM *M* is an element of:

$$K \times \Sigma^*$$

It captures the two things that affect *M*'s future behavior:

• its current state

• the remaining input to be read.

The ***initial configuration*** of a DFSM *M*, on input *w*, is:

$$(s_M, w)$$

The "Yields" Relations

The *yields-in-one-step* relation: $|\!-_M$:

$(q, w)\ |\!-_M\ (q', w')$ iff

- $w = a\,w'$ for some symbol $a \in \Sigma$, and
- $\delta\,(q, a) = q'$

The *yields-in-zero-or-more-steps* relation: $|\!-_M{}^*$

$|\!-_M{}^*$ is the reflexive, transitive closure of $|\!-_M$.

Q7

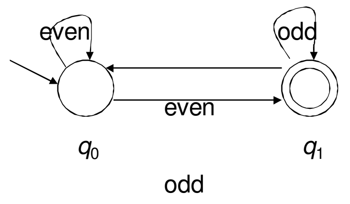
Computations Using FSMs

A ***computation*** by M is a finite sequence of configurations $C_0, C_1, \ldots, C_n$ for some $n \geq 0$ such that:

- C_0 is an initial configuration,
- C_n is of the form (q, ε), for some state $q \in K_M$,
- $\forall i \in \{0, 1, \ldots, n-1\}\ (C_i\ |\!-_M\ C_{i+1})$

An Example Computation

An FSM M that accepts decimal representations of odd integers:



On input 235, the configurations are:

$(q_0, 235)$ $\vdash_M$ $(q_0, 35)$
$\vdash_M$
$\vdash_M$

Thus $(q_0, 235) \vdash_M^* (q_1, \varepsilon)$

Accepting and Rejecting

A DFSM M ***accepts*** a string w iff:

$(s_M, w) \vdash_M^* (q, \varepsilon)$, for some $q \in A_M$

A DFSM M ***rejects*** a string w iff:

$(s_M, w) \vdash_M^* (q, \varepsilon)$, for some $q \notin A_M$

The ***language accepted by*** M, denoted $L(M)$, is the set of all strings accepted by M.

Theorem: Every DFSM M, in configuration (q, w), halts in $|w|$ steps.

Q8-10