18.404/6.840 Lecture 7

Last time:

- Equivalence of variants of the Turing machine model
    a. Multi-tape TMs
    b. Nondeterministic TMs
    c. Enumerators
- Church-Turing Thesis
- Notation for encodings and TMs

Today:

- Decision procedures for automata and grammars

Will have mini chat-breaks (experiment)

TMs and Encodings – review

A TM has 3 possible outcomes for each input :

1. *Accept* (enter)
2. *Reject* by halting (enter)
3. *Reject* by looping (running forever)

is T-recognizable if for some TM .

is T-decidable if for some TM decider .

halts on all inputs

encodes objects as a single string.

Notation for writing a TM is

"On input

[English description of the algorithm]"

Acceptance Problem for DFAs

Let is a DFA and accepts

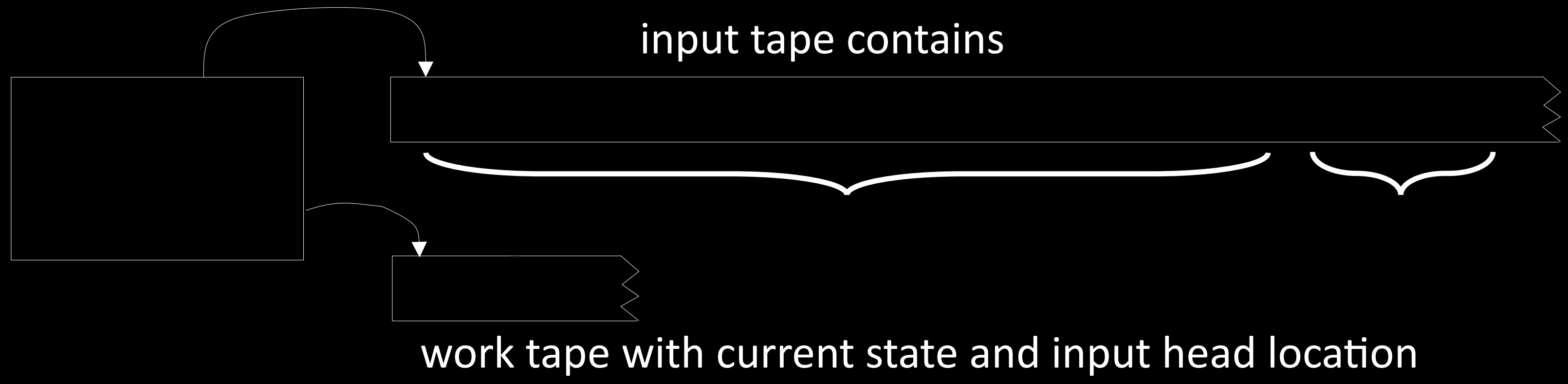
Theorem: is decidable

Proof: Give TM that decides .

"On input

1. Check that has the form where
   is a DFA and is a string; *reject* if not.
2. Simulate the computation of on .
3. If ends in an accept state then *accept*.
   If not then *reject*."

Shorthand:
On input



Acceptance Problem for NFAs

Let is a NFA and accepts

Theorem: is decidable

Proof: Give TM that decides .

“On input

1. Convert NFA to equivalent DFA .
2. Run TM on input . [Recall that decides]
3. *Accept* if accepts.
   Reject if not.”

New element: Use conversion construction and previously constructed TM as a subroutine.

Emptiness Problem for DFAs

Let is a DFA and

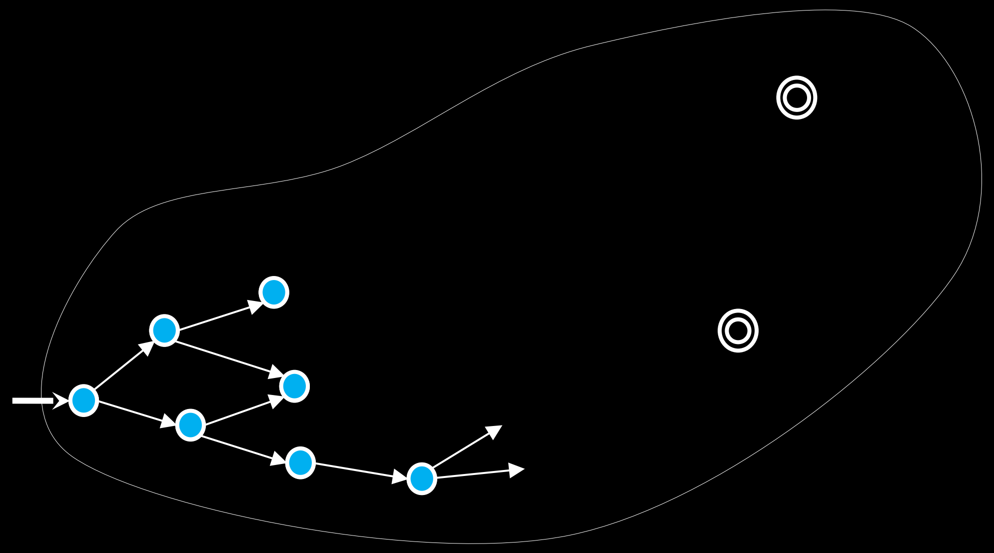
Theorem: is decidable

Proof: Give TM that decides .

"On input [IDEA: Check for a path from start to accept.]

1. Mark start state.
2. Repeat until no new state is marked:

   Mark every state that has an incoming arrow from a previously marked state.
3. *Accept* if no accept state is marked.
   Reject if some accept state is marked."



Equivalence problem for DFAs

Let and are DFAs and

Theorem: is decidable

Proof: Give TM that decides .

Check-in 7.1

Let and are regular expressions and

Can we now conclude that is decidable?

a) Yes, it follows immediately from things we've already shown.

b) Yes, but it would take significant additional work.

c) No, intersection is not a regular operation.

Check-in 7.1


Coffee
Break

Teach at Splash!

esp.mit.edu/splash20

Splash is an annual teaching and learning extravaganza, brought to you by MIT ESP!

When?	November 14-15
Where?	Virtual
What?	Teach anything! Any topic, length, or class size!
Who?	Teach thousands of curious and motivated high schoolers

Acceptance Problem for CFGs

Let

Theorem: is decidable

Proof: Give TM that decides .

"On input

1. Convert into CNF.
2. Try all derivations of length .
3. *Accept* if any generate .
   Reject if not.

Check-in 7.2

Can we conclude that is decidable?

a) Yes.

b) No, PDAs may be nondeterministic.

c) No, PDAs may not halt.

Recall Chomsky Normal Form (CNF) only allows rules:

A BC

B b

Lemma 1: Can convert every CFG into CNF.
Proof and construction in book.

Lemma 2: If is in CNF and then every derivation of has steps.
Proof: exercise.

Check-in 7.2

Emptiness Problem for CFGs

Let is a CFG and

Theorem: is decidable

Proof:

"On input [IDEA: work backwards from terminals]

1. Mark all occurrences of terminals in .
2. Repeat until no new variables are marked
   Mark all occurrences of variable A if
   A is a rule and all were already marked.
3. *Reject* if the start variable is marked.
   Accept if not."

S RTa
R Tb
T a

Equivalence Problem for CFGs

Let are CFGs and

Theorem: is NOT decidable

Proof: Next week.

Let is an ambiguous CFG

Check-in 7.3

Why can't we use the same technique we used to show is decidable to show that is decidable?

a) Because CFGs are generators and DFAs are recognizers.

b) Because CFLs are closed under union.

c) Because CFLs are not closed under complementation and intersection.

Check-in 7.3

Acceptance Problem for TMs

Let is a TM and accepts

Theorem: is not decidable
Proof: Thursday.

Theorem: is T-recognizable
Proof: The following TM recognizes

"On input

1. Simulate on input .
2. *Accept* if halts and accepts.
3. *Reject* if halts and rejects.
4. ~~*Reject* if never halts."~~ Not a legal TM action.

Turing's original "Universal Computing Machine"

Von Neumann said inspired the concept of a stored program computer.

Quick review of today

1. We showed the decidability of various problems about automata and grammars:

   , , , , ,

2. We showed that is T-recognizable.