Decidable Problems

- ○ Given a DFA M and an input w, does M accept w?
- ○ 〃 NFA 〃 〃
- ○ Given a regular expression R and a string w? does R generate w?
- ○ Given a PDA M and an input w, does M accept w?
- • Given a CFG G and a string w, does G generate w?

Undecidable Problems

- ○ Given a TM/NTM M and w, does M accept w?
- ○ Given a TM/NTM M and w, does M halts on w?

More undecidable problems

- ○ Given a TM M, does M recognize $\emptyset$?
- ○ Given TMs M_1 and M_2, do they recognize the same language?
- ○ Given a TM M, does M recognize a regular language? context-free language?

Finite automata problems can be reformulated as languages

Does DFA B accept input string w?

Consider the language

$$A_{DFA} = \{ \langle B, w \rangle \mid B \text{ is a DFA that accepts } w \}$$

The following conditions are equivalent

(a) B accepts w

(b) $\langle B, w \rangle \in A_{DFA}$

Showing that the computational problem is decidable is equivalent to showing that the language is decidable

Thm: A_{DFA} is a decidable language.

Proof: The following TM M decides A_{DFA}

M = "On input $\langle B, w \rangle$, where B is a DFA and w is a string

1. Simulate B on w
2. If the simulation ends in an accept state, accept. If it ends in a nonaccepting state, reject "

scan and check if input is correct

- B can be represented by five components $(Q, \Sigma, \delta, q_0, F)$

simulation is straightforward

Given an NFA B and a string w, does B accept w?

$A_{NFA} = \{\langle B, w\rangle \mid B$ is an NFA that accepts $w\}$

Thm: A_{NFA} is a decidable language.

Proof. The following TM decides it.

```
N= "On input <B, w>, where B is an NFA
    and ω is a string
      1. Convert NFA B into an equivalent DFA
         C (Thm 1.39)
      2. Run previous TM on <C, w>
      3. If that TM accepts, accept,
         otherwise reject
```

N.B. Use of subroutine

Given a regular expression R and a string w Does R generate ω?

$A_{REX} = \{\langle R, w\rangle \mid R$ is a regular expression that generates $w\}$

Thm: A_{REX} is a decidable language

Proof: The following TM P decides it

```
P = "On input <R, w>, where R is a regular
     expression, and w is a string,
       1. Convert R into an equivalent DFA C
       2  Run earlier TM on input <C, w>
       3  If that TM accepts, accept
          otherwise reject.
```

Given a DFA A, does A recognize ∅?

$E_{DFA} = \{\langle A\rangle \mid A$ is a DFA and $L(A) = \emptyset\}$

Thm: E_{DFA} is a decidable language

Proof: The following TM T decides it

```
T= "On input <A>, where A is a DFA
    1. mark the start state of A
    2. Repeat until no new states are
       marked.
    3.      mark any state that has a transition
            coming into it from any state that
            is already marked
    4. If no accept state is marked, accept,
       otherwise reject
```

N.B. tests whether any accept state is reachable from the start state.

Given DFAs A and B do they recognize the same language?

$EQ_{DFA} = \{\langle A, B\rangle \mid A$ and B are DFA and $L(A) = L(B).\}$

Thm: EQ_{DFA} is a decidable language

Proof: Construct a DFA C using A and B so that $L(C) = (L(A) \cap \overline{L(B)}) \cup (\overline{L(A)} \cap L(B))$

$L(C)$ is symmetric difference of $L(A)$ and $L(B)$

(recall that DFA is closed under complementation, intersection, union)

$EQ_{DFA} = \{\langle A, B\rangle \mid$ A and B are DFA and $L(A) = L(B).\}$

"On input $\langle A, B\rangle$ where A, B are DFA

1 Construct C as described.

2 Run previous TM (for E_{DFA}) on $\langle C\rangle$

3. If that TM accepts, accept otherwise, reject.