Decidable Problems

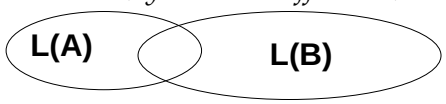
EQ_{DFA} = {<A,B> | A,B are DFA, and L(A) = L(B) }

Th: EQ_{DFA} is decidable.

Proof: We construct a DFA C with

$L(C) = (L(A) \cap \overline{L(B)}) \cup (\overline{L(A)} \cap L(B))$

(symmetric difference)



C can be constructed by TM M using algorithms we developed for closure under $\cup$, $\cap$, $\overline{\ }$

Since L(C) is empty iff. L(A) = L(B), we can use previous theorem for deciding L(C) nonempty.

1

Decidable Problems for CFG's

A_{CFG} = {<G,w> | G is a CFG, G generates w}

Th. A_{CFG} is decidable.

Proof idea: TM could start with start var. S and try all derivations. If w is not in L(G), there will be none!
So trying all derivations to see if get w would not yield decider.
To define a decider, we first put G in Chomsky normal form.
Then we can bound the number of steps in a derivation TM needs to try to $2|w|-1$. QU: WHY?
The decider D will then list all derivations of length 0 to $2|w|-1$. If any derivation generates w, D accepts; ow, reject.

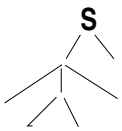
2

Decidable Problems on CFG's

E_{CFG} = {<G> | G is a CFG, L(B) is nonempty}

Th. E_{CFG} is decidable.

Proof idea: Can't try all strings (won't be decidable). Consider any parse tree for a string w.



If parse tree has path of length > $|V|$, then there is a repetition of variables along that path, and we can produce a shorter parse tree. We can repeat this process so that there is a parse tree with no path of length > $|V|$. Therefore, if G generates any string, it generates a string whose parse tree has no path > $|V|$.

Decidable Problems on CFG's

Th. E_{CFG} is decidable. ***(continued)***

A decider D for this problem will generate a collection C of parse trees on its tape. C will initially contain **S** *. D repeatedly adds to C any tree that can be obtained from one already in C by applying a single rule, such that*

1. *the new tree is not in C*
2. *the new tree does not have any path of len. > $|V|$*

Since there are only a finite number of trees of fixed length, the TM D will eventually complete C. L(G) is nonempty iff at least one tree in C has only terminals as leaves.

Decidable Problems for CFG's

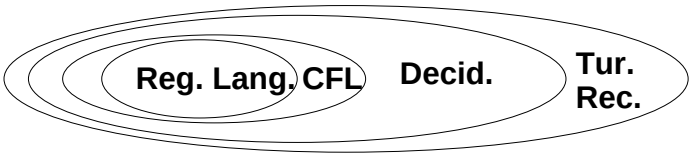
EQ_{CFG} = {<G,H> | G,H are CFG, and L(G) = L(H) }

EQ_{CFG} decidable??

Can't use same idea we used for DFA's (symmetric diff.) because CFL's are NOT closed under complement.

Turns out EQ_{CFG} is NOT decidable. We can't prove it yet.

Th. Every CFL is decidable. (HW 3)



Undecidable Problems

Showed these problems for FA and CFG's decidable:
Acceptance, Emptiness, Equivalence
Qu: What about for TM?

A_{TM} = {<M,w> | M is a TM that accepts w}

Will show that A_{TM} is not decidable (ie, undecidable)

What's wrong with:

Define TM U (Universal TM) on input <M,w>

1. *Simulate M on input w*
2. *If M accepts, U accepts; if M rejects, U rejects.*

U can act like any TM on any input

Is U a decider????

Universal TM U is not a decider

U has no way to determine if M halts on input w.

Lemma: A_{TM} is Turing-recognizable.

Proof: U is the recognizer.

Halting Problem: A_{HALT} = {<B,w> | B halts on input w}

A_{HALT} is decidable ⟷ A_{TM} is decidable

We show A_{TM} is undecidable (then A_{HALT})

Doesn't mean that we CAN'T determine for a particular <B,w> whether B halts (accepts) on input w.

Does mean that there is no algorithm that works for all input <B,w>. Gives fundamental limitation of Algorithms/TM.

Diagonalization for TM

Suppose there was a super-diagonalizer TM D

D on input <M>:

1. **Simulate M on input <M>**
2. **If M accepts, D rejects; if M rejects, D accepts.**
   If M never halts, D accepts.

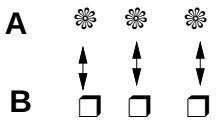
Note D has power to decide if M halts or not on <M>.

QU: Can D be in the list (ie = Mi for some i)?

	M1	***M2***	***M3***	***.....***
<M1>				
<M2>				
<M3>				
:				

Diagonalization in Mathematics

Fundamental idea of counting: **1-1 correspondence**



(function f:A→B, which is

1. **1-1 (f(a) =\ f(b) if a =\ b)**
2. **onto B (every element in b included)**

Works for finite sets! How count infinite sets??

N	**1**	**2**	**3**	**4**	**....**	*Take f(n) = 2n*
Even	**2**	**4**	**6**	**8**	**....**	*Its 1-1 and onto.*

So we say N and Even have the same number of elements (ie have the same size) even though N includes Even!

Def: A set A is countable if it is finite or it is the same size as *N*.

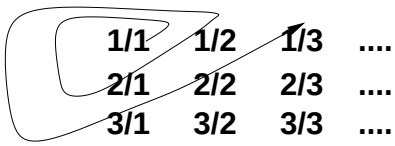
9

Another Example

Q = { m/n | m,n are in N } *Rational Numbers*

Lemma: Q is countable.

Proof: Define f as follows:



Don't worry about factoring (e.g., 2/2, 3/3 are taken as diff.)

Need to show that f so defined is 1-1 and onto.

Note that defining f via rows or columns won't work! Why?

What about real numbers R?

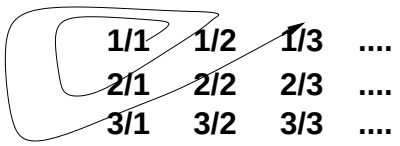
10

Another Example

Q = { m/n | m,n are in N } *Rational Numbers*

Lemma: Q is countable.

Proof: Define f as follows:



Don't worry about factoring (e.g., 2/2, 3/3 are taken as diff.)

Need to show that f so defined is 1-1 and onto.

Note that defining f via rows or columns won't work! Why?

What about real numbers R?

Another Example, continued

We could write an algorithm to pick the ith digit of x so that its always different from f(i)'s ith digit.

x differs from f(i) on ith digit (after decimal point).

Therefore, x ≠ f(i) for any i.

(Slight problem: .19999.... and .20000.... the same; never pick 0 or 9)

x is a real number, but its not in the list.

Therefore, f is not onto R .

Contradiction, proving that R is not countable.