Pre-lecture brain teaser

In the following languages, three are decidable and three are undecidable. Which are which?

- $A_{CFG} = \left\{ \langle G, w \rangle \;\middle|\; G \text{ is a } CFG \text{ that generates string } w \right\}$.
- $E_{CFG} = \left\{ \langle G \rangle \;\middle|\; G \text{ is a } CFG \text{ and } L(G) = \emptyset \right\}$.
- $ALL_{CFG} = \left\{ \langle G \rangle \;\middle|\; G \text{ is a } CFG \text{ and } L(G) = \Sigma^* \right\}$.
- $A_{LBA} = \left\{ \langle M, w \rangle \;\middle|\; M \text{ is a } LBA \text{ that generates string } w \right\}$.
- $E_{LBA} = \left\{ \langle M \rangle \;\middle|\; M \text{ is a } LBA \text{ where } L(M) = \emptyset \right\}$.
- $ALL_{LBA} = \left\{ \langle M \rangle \;\middle|\; M \text{ is a } LBA \text{ where } L(M) = \Sigma^* \right\}$.

ECE-374-B: Lecture 25 - Midterm 3 Review

Instructor: Nickvash Kani

University of Illinois at Urbana-Champaign

Pre-lecture brain teaser

In the following languages, three are decidable and three are undecidable. Which are which?

- $A_{CFG} = \left\{ \langle G, w \rangle \;\middle|\; G \text{ is a } CFG \text{ that generates string } w \right\}$.
- $E_{CFG} = \left\{ \langle G \rangle \;\middle|\; G \text{ is a } CFG \text{ and } L(G) = \emptyset \right\}$.
- $ALL_{CFG} = \left\{ \langle G \rangle \;\middle|\; G \text{ is a } CFG \text{ and } L(G) = \Sigma^* \right\}$.
- $A_{LBA} = \left\{ \langle M, w \rangle \;\middle|\; M \text{ is a } LBA \text{ that generates string } w \right\}$.
- $E_{LBA} = \left\{ \langle M \rangle \;\middle|\; M \text{ is a } LBA \text{ where } L(M) = \emptyset \right\}$.
- $ALL_{LBA} = \left\{ \langle M \rangle \;\middle|\; M \text{ is a } LBA \text{ where } L(M) = \Sigma^* \right\}$.

A_{TM}

$\bar{E}_{TM}$

$ALL_{\bar{T}M}$

A_{CFG} decidable?

$A_{CFG} = \{\langle G, w \rangle \mid \text{if } G \text{ accepts } w\}$

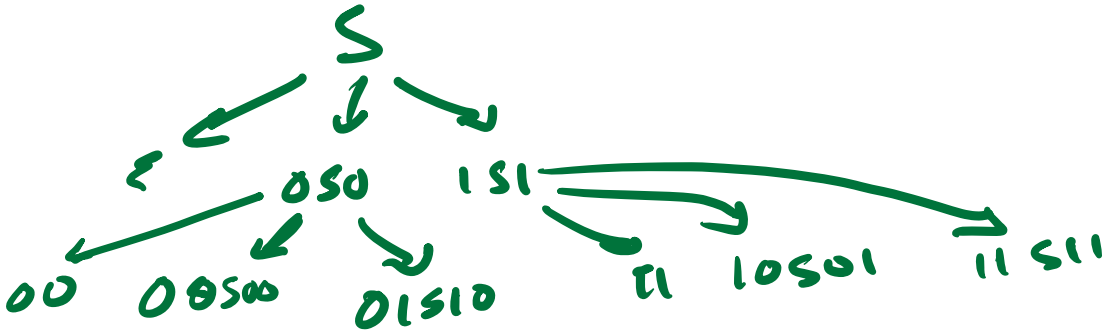
A_{CFG} decidable?

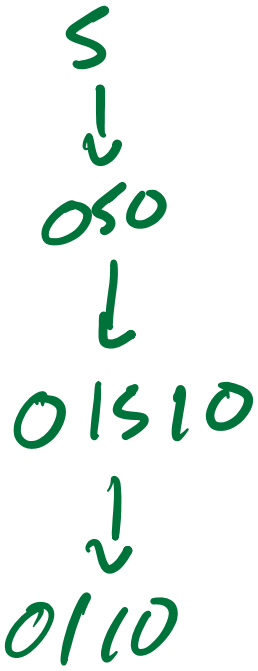
YES!

A_{CFG} decidable?

YES!

- $V = \{S\}$
- $T = \{0, 1\}$
- $P = \{S \to \epsilon \mid 0S0 \mid 1S1\}$
  (abbrev. for $S \to \epsilon, S \to 0S0, S \to 1S1$)





A_{CFG} decidable?

$V \to TVT$ YES!

Lemma
A CFG in Chomsky normal form can derive a string w in at most 2^n steps!

Knowing this, we can just simulate all the possible rule combinations for 2^n steps and see if any of the resulting strings matches w.

E_{CFG} decidable?

$E_{CFG} = \{\langle G \rangle \mid G \text{ does not accept any strings}\}$

E_{CFG} decidable?

YES!

E_{CFG} decidable?

YES!

In this case, we just need to know if we can get from the start variable to a string with only terminal symbols.

1. Mark all terminal symbols in G
2. Repeat until no new variables get marked:
   2.1 Mark any variable A where G has the rule $A \to U_1 U_2 \ldots U_k$ where U_i is a marked terminal/variable
3. If start variable is ~~not~~ marked, accept. Otherwise reject.

- $V = \{S\}$
- $T = \{0, 1\}$
- $P = \{S \to \epsilon \mid 0S0 \mid 1S1\}$ (abbrev. for $S \to \epsilon, S \to 0S0, S \to 1S1$)

ALL_{CFG} decidable?

$ALL_{CFG} = \{\langle G \rangle \mid G \text{ accepts all strings}$
$G \text{ does not rejects any strings}\}$

$S \to A \mid B \mid C$

$A \to 1A$

$B \to 0 \mid 1 \mid C$

$C \to \varepsilon \mid 11 \mid 00$

ALL_{CFG} decidable?

Nope

ALL_{CFG} decidable?

Nope

Proof requires computation histories which are outside the scope of this course.

A_{LBA} decidable?

$$A_{LBA} = \{ \langle M, w \rangle \mid M \text{ accepts } w \}$$

A_{LBA} decidable?

YES!

A_{LBA} decidable?

YES!

Remember a LBA has a finite tapes. Therefore we know:

1. A tape of length n where each cell can contain g symbols, you have g^n possible configurations.
2. The tape head can be in one of n positions and has q states yielding a ~~tape~~ head that can be in qn configurations.
3. Therefore the machine can be in qng^n configurations.

A_{LBA} decidable?

YES!

Remember a LBA has a finite tapes. Therefore we know:

1. A tape of length n where each cell can contain g symbols, you have g^n possible configurations.
2. The tape head can be in one of n positions and has q states yielding a tape that can be in qn configurations.
3. Therefore the machine can be in qng^n configurations.

Lemma
If an LBA does not accept or reject in qng^n then it is stuck in a loop forever.

A_{LBA} decidable?

Decider for A_{LBA} will:

1. Simulate $\langle M \rangle$ on w for qng^n steps.
   1.1 if accepts, then accept
   1.2 if rejects, then reject
2. If neither accepts or rejects, means it's in a loop in whihc case, reject.

E_{LBA} decidable?

$E_{LBA} = \{\langle M \rangle \mid M \text{ does not accept any string}\}$

E_{LBA} decidable?

Nope

E_{LBA} decidable?

Nope

Proof requires computational history trick, a story for another time.....

ALL_{LBA} decidable?

$ALL_{LBA} = \{\langle M \rangle \mid M$ accepts all strings $\}$

(M does not reject or loop on any string)

ALL_{LBA} decidable?

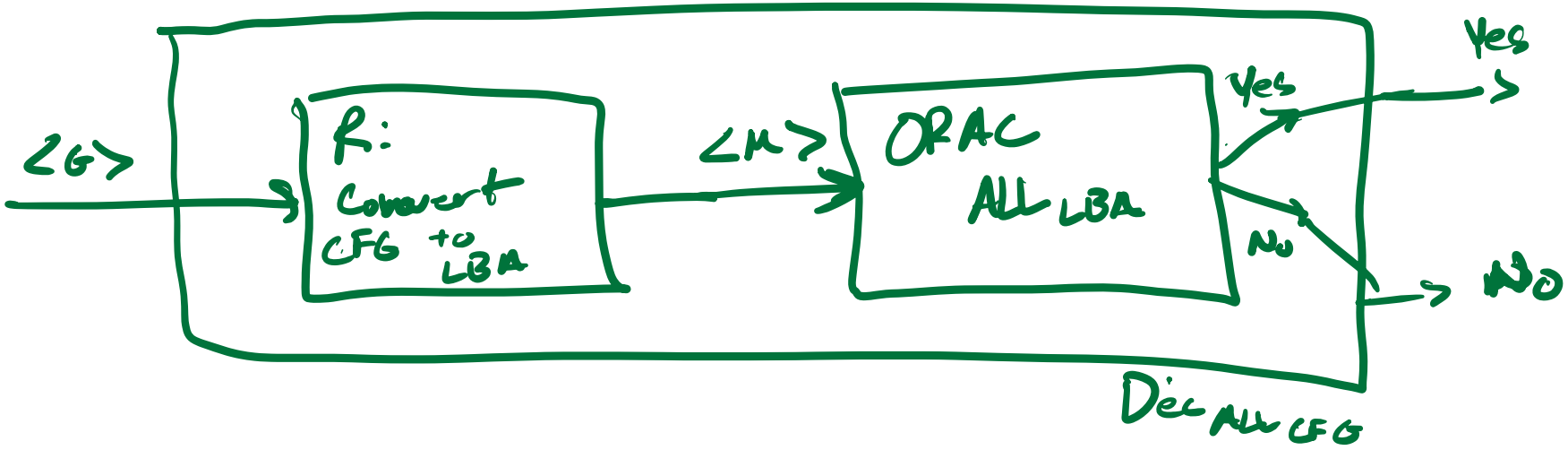
Nope

ALL_{LBA} decidable?

Nope

No standard proof for this, but let's look at a pattern:

$ALL_{CFG} \Rightarrow ALL_{LBA}$



Is 3SAT decidable? $\varphi(x\{0,\dots n-1\})$

Dec3SAT

for all assignment of x

if $\varphi(x) =$ true

return true

return false

Decidability across grammar complexities

$\{\langle M, w\rangle \mid M \text{ accepts } w \text{ in } |w||w| \text{ steps}\}$

	DFA	CFG	PDA	LBA	TM
A	D	D	D	D	U
E	D	D	D	U	U
ALL	D	U	U	U	U

Eventually problems get too tough....

$A475_{TM} = \{\langle M\rangle \mid M \text{ accepts } 475 \text{ strings}\}$

$A475_{DFA} = \{\langle M\rangle \mid M \text{ accepts atleast } 475 \text{ strings}\}$

ALL_{LBA} decidable?

Nope

No standard proof for this, but let's look at a pattern:

So we sort've know that ALL_{LBA} isn't decidable because we knew ALL_{CFG} wasn't (though intuition is never sufficient evidence).

Un-/decidability practice problems

Available Undecidable languages

- $L_{Accept} = \left\{ \langle M, w \rangle \;\middle|\; M \text{ is a } TM \text{ and accepts } w \right\}$.
- $L_{HALT} = \left\{ \langle M \rangle \;\middle|\; M \text{ is a } TM \text{ and halts on } \varepsilon \right\}$.

Practice 1: Halt on Input

Is the language:

$$L_{HaltOnInput} = \left\{ \langle M, w \rangle \;\middle|\; M \text{ is a } TM \text{ and halts on } w \right\}.$$

Practice 2: L has fooling set

Is the language:

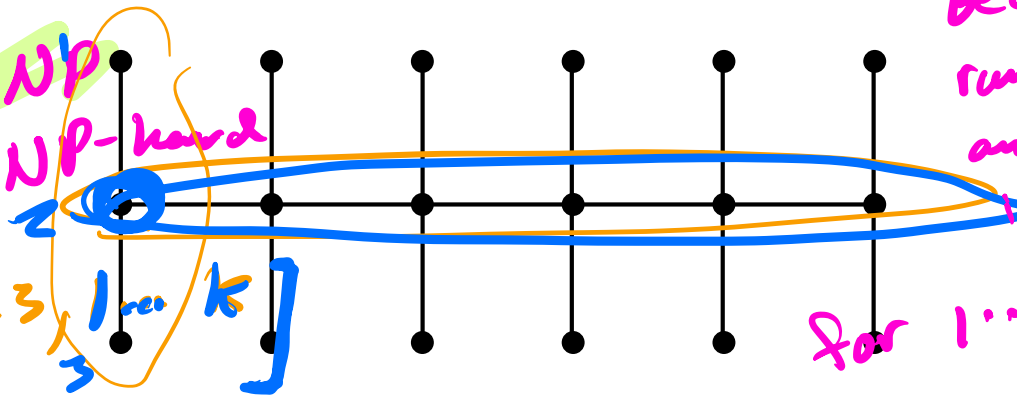
$$L_{HasFooling} = \left\{ \langle M \rangle \;\middle|\; M \text{ is a } TM \text{ and } L(M) \text{ has a fooling set} \right\}.$$

NP-Complete practice problems

Practice: NP-Complete Reduction I

A centipede is an undirected graph formed by a path of length k with two edges (legs) attached to each node on the path as shown in the below figure. Hence, the centipede graph has $3k$ vertices. The **CENTIPEDE** problem is the following: given an undirected graph $G = (V, E)$ and an integer k, does G contain a centipede of $3k$ vertices as a subgraph? Prove that **CENTIPEDE** is **NP-Complete**.

Need to show
Centipede is $\in NP$
it is $\in NP$-hard



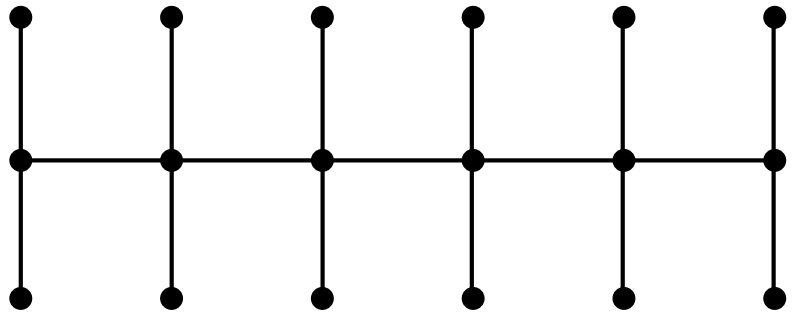
Because this algorithm runs in polynomial time and check the YES instance of Centipede then CENTIPEDE $\in NP$

Certificate: $C[1..3, 1..k]$

Certifier: for $i = 1..k$
$\{(C[2,i], C[1,i]), (C[2,i], C[3,i])\} \in E$, $(C[2,i], C[2,i+1]) \in E$ for $1..k-1$

Practice: NP-Complete Reduction

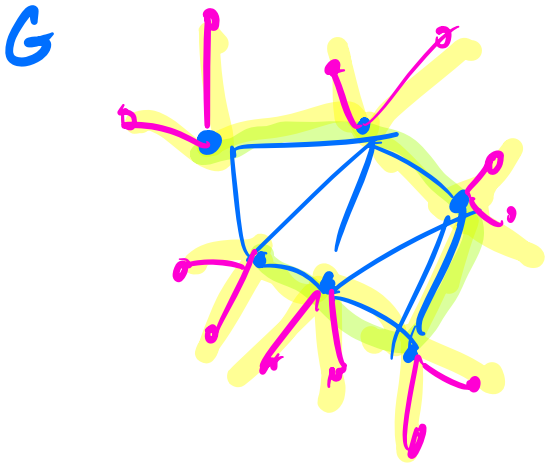
What do we need to do to prove Centipede is NP-Complete?

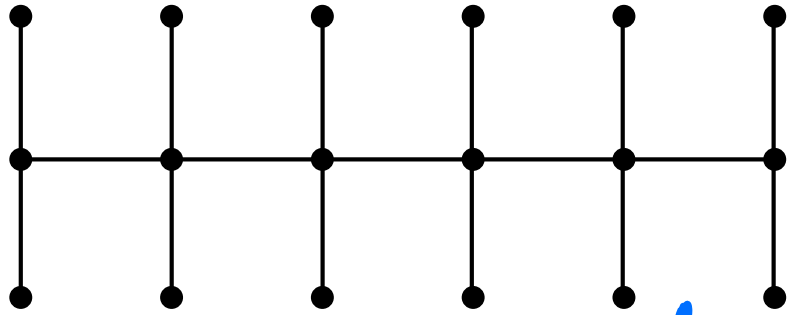


Practice: NP-Complete Reduction

Prove Centipede is in **NP**:

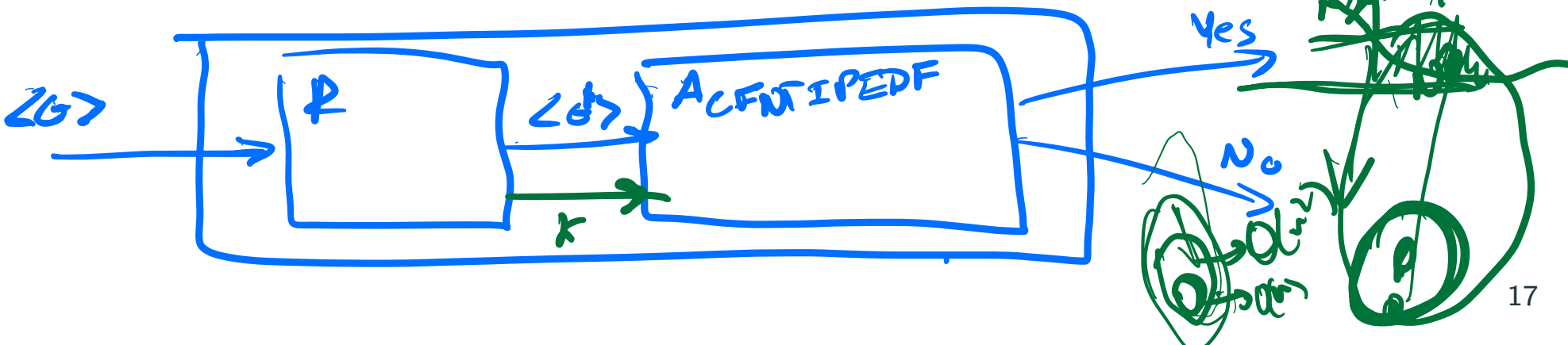
Hamiltonian Path $\leq_p$ CENTIPEDE





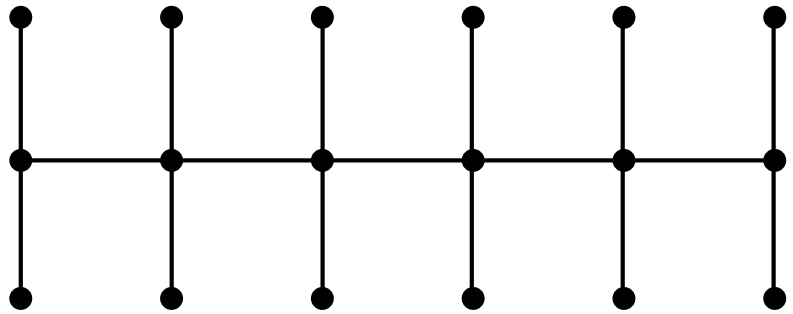
A_{HP}

G' is G with 2 vertices added and attached to every V



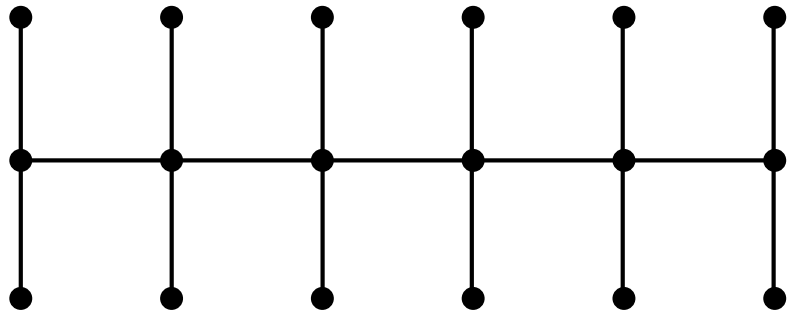
Practice: NP-Complete Reduction

Prove Centipede is in **NP-hard**:



Practice: NP-Complete Reduction

Prove Centipede is in **NP-hard**:



Hamiltonian Path: Given a graph G (either directed or undirected), is there a path that visits every vertex exactly once

$HC \leq_P$ *Centipede*

Practice: NP-Complete Reduction I

A quasi-satisfying assignment for a 3CNF boolean formula Φ is an assignment of truth values to the variables such that at most one clause in Φ does not contain a true literal. Prove that it is NP-complete to determine whether a given 3CNF boolean formula has a quasi-satisfying assignment.

Practice: NP-Complete Reduction I

A quasi-satisfying assignment for a 3CNF boolean formula Φ is an assignment of truth values to the variables such that at most one clause in Φ does not contain a true literal. Prove that it is NP-complete to determine whether a given 3CNF boolean formula has a quasi-satisfying assignment.

Prove quasiSAT is in NP

Practice: NP-Complete Reduction I

A quasi-satisfying assignment for a 3CNF boolean formula Φ is an assignment of truth values to the variables such that at most one clause in Φ does not contain a true literal. Prove that it is NP-complete to determine whether a given 3CNF boolean formula has a quasi-satisfying assignment.

Prove quasiSAT is NP-hard

Practice: NP-Complete Reduction II

Prove quasiSAT is NP-hard

Practice: NP-Complete Reduction II

Prove quasiSAT is NP-hard

3SAT: Given a boolean formula in conjunctive normal form, with exactly three distinct literals per clause, does the formula have a satisfying assignment.

Good luck on the exam