CSCI 361 Lecture 16:

Reductions Using Computation Histories

Shikha Singh

Announcements & Logistics

- Hand in **reading assignment # 11**
- No **reading assignment today**
- **HW 6** due tomorrow at 10 pm
    - Office hours today and tomorrow 2.30-4 pm

Last Time

- Practice with reductions to prove a bunch of languages are undecidable
- Introduced mapping reducibility
    - Helps to reason about Turing (un)/recognizable languages

Today

- Rice Theorem
- Using computational histories method to prove undecidability

Rice's Theorem

Any nontrivial **property of the languages** recognized by Turing machines is undecidable.

- Is the language empty? Is it finite? Is it infinite? Is it regular?
- Does the language contain strings in Σ^*
- Is the language the same as the language of another TM?

We proved many such examples in class.

HW 6 will have more practice with these.

Rice's Theorem

Any nontrivial **property of the languages** recognized by Turing machines is undecidable.

- Is the language empty? Is it finite? Is it infinite? Is it regular?
- Does the language contain strings in Σ^*
- Is the language the same as the language of another TM?

In contrast, questions about the TM's structure are decidable

- Has more than 15 states, has no transitions into its reject state, etc

Decidable or Not

Questions about behavior of TM's computation on inputs may or may not be decidable.

(HW 6 Problem): One of these is decidable, one is not:

- Does a given TM M and input w, does it ever move its head left when running on w?
- Does a given TM M and input w, does it ever move its head three times in a row when running on w?

Undecidable Languages about CFGs

The following languages about CFGs are all undecidable:

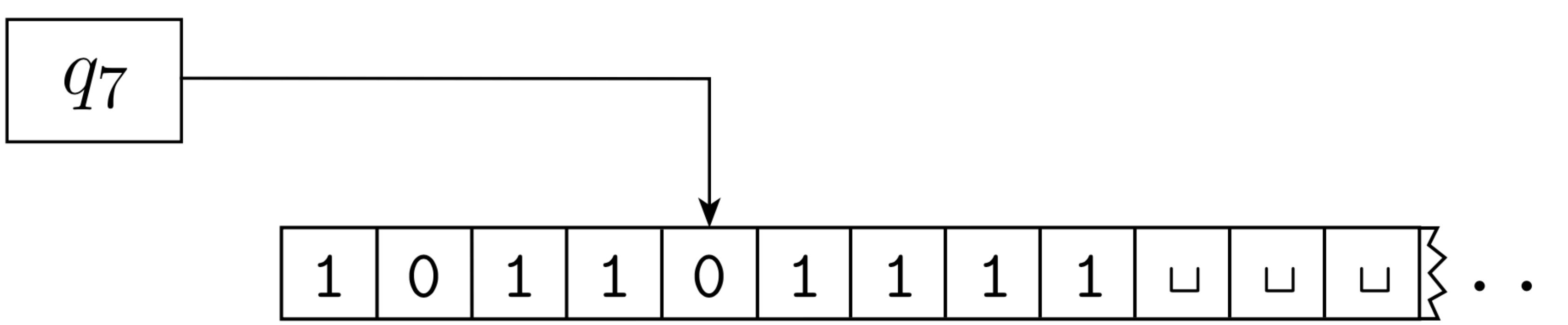
- **(All)** Given a CFG G, is $L(G) = \Sigma^*$?
- **(EQ)** Given two CFGs G_1, G_2, is $L(G_1) = L(G_2)$?
- **(Disjoint)** Given two CFGs G_1, G_2, is $L(G_1 \cap G_2) = \varnothing$?
- **(Ambiguity)** Given a CFG G, is it ambiguous?
- **(Disjoint Regular)** Given two CFGs G_1, G_2, is $L(G_1 \cap G_2)$ a regular language?
- .... etc
- To prove that these are undecidable, we need a way to encode the computation history of a Turing into "grammar" form

Recall: TM Configurations

- A configuration C_1 **yields** a configuration C_2 if the TM can legally go from C_1 to C_2 using its transition function δ
- Consider symbols $a, b, c \in \Gamma$ and strings $u, v \in \Gamma^*$ then

$ua\ q_i\ bv$ yields $u\ q_j\ acv$ if $\delta(q_i, b) = (q_j, c, L)$, and

$ua\ q_i\ bv$ yields $uac\ q_j\ v$ if $\delta(q_i, b) = (q_j, c, R)$



Computation Histories

- Consider a **deterministic** TM M and an input string w
- M's computation on w can:
  - Halt and accept
  - Halt and reject
  - Or never halt (loop forever)
- For the first two cases, M's **computation history** on w is a **finite** sequence $C_1, C_2, \ldots, C_\ell$ where
  - C_1 is the start configuration
  - C_i yields C_{i+1} for each $1 \leq i \leq \ell - 1$
  - C_ℓ is an accept or reject configuration
- If M does not halt on w, no computation history exists

ALL_{CFG} is undecidable

- **Theorem.** The language

  $\text{ALL}_{\text{CFG}} = \{\langle G\rangle \mid G$ is a CFG and $L(G) = \Sigma^*\}$ is undecidable.

- **Proof Idea.**
  - Show $\overline{\text{A}_{\text{TM}}} \leq_m \text{ALL}_{\text{CFG}}$
  - Given $\langle M, w\rangle$, create a CFG G such that

$\langle M, w\rangle \in \overline{\text{A}_{\text{TM}}}$ if and only if $L(G) = \Sigma^*$, equivalently

M does not accept w if and only if $L(G) = \Sigma^*$

ALL_{CFG} is undecidable

- **Theorem.** The language $ALL_{CFG} = \{\langle G \rangle \mid G$ is a CFG and $L(G) = \Sigma^*\}$ is undecidable.
- **Proof Idea.**
  - Given M and w, construct a grammar that generates all strings except the accepting computation history of M on w
  - If M does not accept w, then $L(G) = \Sigma^*$
  - Otherwise, $L(G) \neq \Sigma^*$

ALL_{CFG} is undecidable

- **Theorem.** The language
  $\text{ALL}_{\text{CFG}} = \{\langle G \rangle \mid G$ is a CFG and $L(G) = \Sigma^*\}$ is undecidable.
- **Proof Idea.**
  - Suppose the computation history of M on w is $\#C_1\#C_2\#\ldots\#C_\ell$ then it is not an accepting history if any of these conditions hold:
    - C_1 is not the start configuration
    - some C_i does not yield C_{i+1}
    - C_ℓ is not an accepting configuration
- Create a PDA D that accepts if one of these conditions are true

ALL_{CFG} is undecidable

- **Theorem.** The language

  $\text{ALL}_{\text{CFG}} = \{\langle G \rangle \mid G \text{ is a CFG and } L(G) = \Sigma^*\}$ is undecidable.
- **Proof Idea.**
  - PDA D non-deterministically guesses which of the three conditions are true
  - To check if some C_i does not yield C_{i+1}, consider a different way to encode accepting configurations

$$\#\underbrace{\longrightarrow}_{C_1}\#\underbrace{\longleftarrow}_{C_2^{\mathcal{R}}}\#\underbrace{\longrightarrow}_{C_3}\#\underbrace{\longleftarrow}_{C_4^{\mathcal{R}}}\#\quad\cdots\quad\#\underbrace{\qquad\qquad}_{C_l}\#$$

Post Correspondence Problem

- Simple problem about strings, defined by Emil Post (1946)
- Let Σ be any alphabet with at least two letters
- An instance of the Post correspondence problem (PCP) is given by a two sequences $A = (a_1, a_2, \ldots, a_m)$ and $B = (b_1, b_2, \ldots, b_m)$ where $a_i, b_i \in \Sigma^*$
- **Problem.** Does there exist a finite sequence $i_1, i_2, \ldots, i_k$ where each i_j is an index from $1, \ldots, m$ such that

$$a_{i_1} a_{i_2} \ldots a_{i_k} = b_{i_1} b_{i_2} \ldots b_{i_k}$$

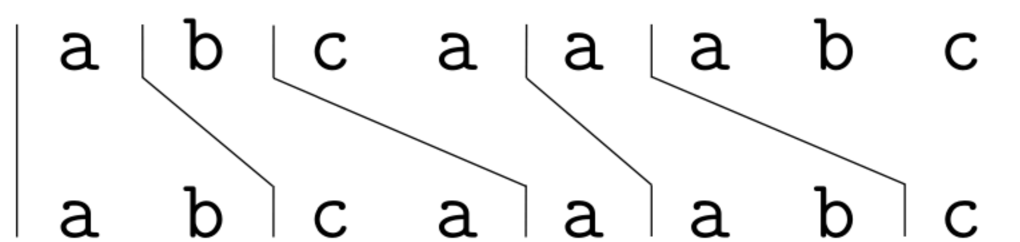
Post Correspondence Problem

- **Alternate Formulation:** An input is a collection of dominos with two sides: each containing two strings $\left[\frac{a_1}{b_1}\right], \left[\frac{a_2}{b_b}\right], \ldots, \left[\frac{a_m}{b_m}\right]$
- Problem is to find a sequence of these dominoes (*repetitions are allowed*) such that the string formed by concatenating the top is the same as the string formed by concatenating the bottom
- E.g. Consider $\left\{\left[\frac{\mathtt{b}}{\mathtt{ca}}\right], \left[\frac{\mathtt{a}}{\mathtt{ab}}\right], \left[\frac{\mathtt{ca}}{\mathtt{a}}\right], \left[\frac{\mathtt{abc}}{\mathtt{c}}\right]\right\}.$

Post Correspondence Problem

- E.g. Consider $$\left\{ \left[\frac{\texttt{b}}{\texttt{ca}}\right], \left[\frac{\texttt{a}}{\texttt{ab}}\right], \left[\frac{\texttt{ca}}{\texttt{a}}\right], \left[\frac{\texttt{abc}}{\texttt{c}}\right] \right\}.$$

- A possible solution $$\left[\frac{\texttt{a}}{\texttt{ab}}\right]\left[\frac{\texttt{b}}{\texttt{ca}}\right]\left[\frac{\texttt{ca}}{\texttt{a}}\right]\left[\frac{\texttt{a}}{\texttt{ab}}\right]\left[\frac{\texttt{abc}}{\texttt{c}}\right]$$



Post Correspondence Problem

- E.g. Consider $\left\{ \left[\frac{\texttt{abc}}{\texttt{ab}}\right], \left[\frac{\texttt{ca}}{\texttt{a}}\right], \left[\frac{\texttt{acc}}{\texttt{ba}}\right] \right\}$

- No possible solution, why?

- Given a general instance, the PCP problem is to determine if it has a solution.
- **Theorem.** PCP is undecidable.

PCP is Undecidable

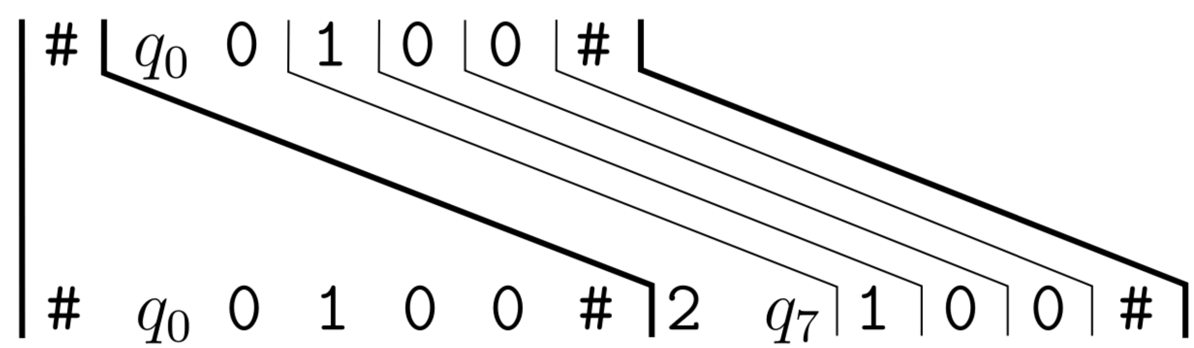
- **Reduce A_{TM} to PCP:** Given $\langle M, w \rangle$, create an instance of PCP such that it has a solution iff M accepts w
- **Key idea:** create dominoes such that a match between top and bottom strings forces M's simulation on w
- Technicalities (ignore for now):
  - Assume M never moves its head off the left-hand end of the tape
  - If $w = \varepsilon$, assume $w = \sqcup$ in the construction
  - Modify PCP to **require starting with the first domino** $\left[\frac{t_1}{b_1}\right]$
- Can remove these restrictions at the end

PCP is Undecidable

- **Part 1.** Let the first domino be $\left[\dfrac{\#}{\#w_1w_2\ldots w_n\#}\right]$
- **Part 2.** For each transition of the type $\delta(q,a)=(r,b,R)$, create a domino $\left[\dfrac{qa}{br}\right]$
- **Part 3.** For each transition of the type $\delta(q,a)=(r,b,L)$ create a domino $\left[\dfrac{cqa}{rcb}\right]$ for every $c \in \Gamma$
- **Part 4.** To "copy over" symbols that are not adjacent to head position on either side, create domino $\left[\dfrac{a}{a}\right]$ for each $a \in \Gamma$
- **Part 5.** To match the # symbols in the middle and at end add $\left[\dfrac{\#}{\#}\right]$ and $\left[\dfrac{\#}{\sqcup\#}\right]$

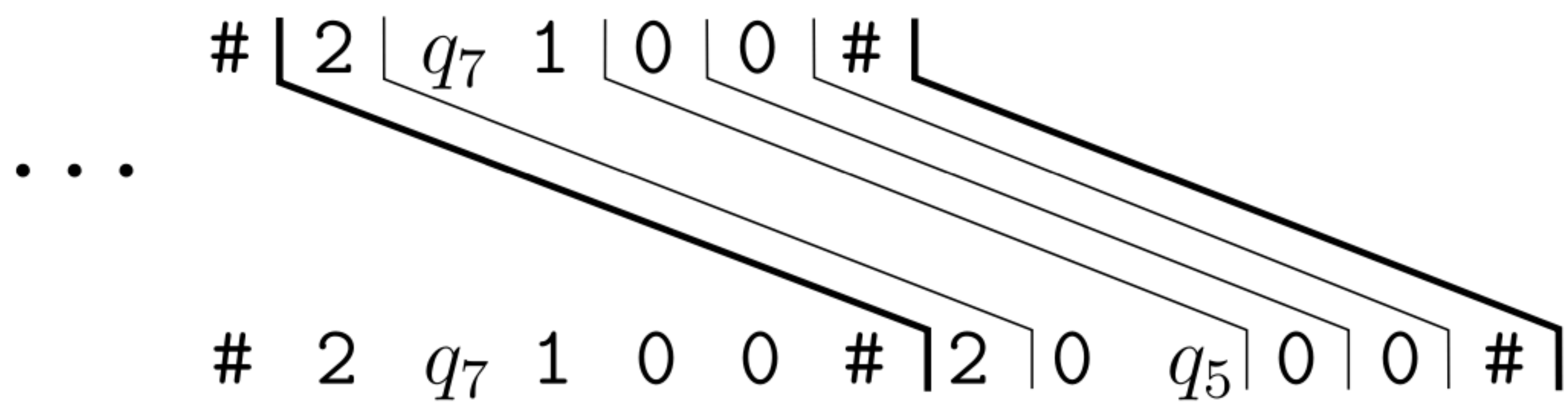
Construction Example

- Consider an M that starts in q_0 on input 0100 and q_00100 yields $2q_7100$ by following $\delta(q_0,0) = (q_7,2,R)$
- Part 1 adds the first domino $\left[\dfrac{\#}{\#q_00100\#}\right]$
- Part 2 adds the domino $\left[\dfrac{q_00}{2q_7}\right]$
- Using Part 5 can force the match:



Construction Example

- Recall that $q_0 0100$ yields $2q_7 100$
- Now suppose $\delta(q_7,1) = (q_5,0,R)$
- That is, $2q_7 100$ yields $20q_5 00$, then we add the domino $\left[\dfrac{q_7 1}{0 q_5}\right]$
- Thus, the partial match looks like this:



Construction Example

- Finally, to handle the last transition that includes a q_{accept} add

$$\left[\frac{a\ q_{\text{accept}}}{q_{\text{accept}}}\right] \text{ and } \left[\frac{q_{\text{accept}}a}{q_{\text{accept}}}\right] \text{for each } a \in \Gamma$$

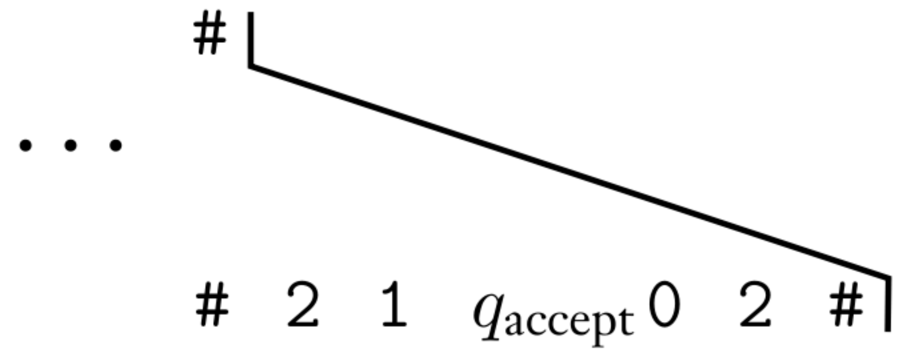
Suppose the last configuration of M is

$$\begin{array}{lccccccc} & \# & & & & & & \\ \cdots & & & & & & & \\ & \# & 2 & 1 & q_{\text{accept}} & 0 & 2 & \# \end{array}$$

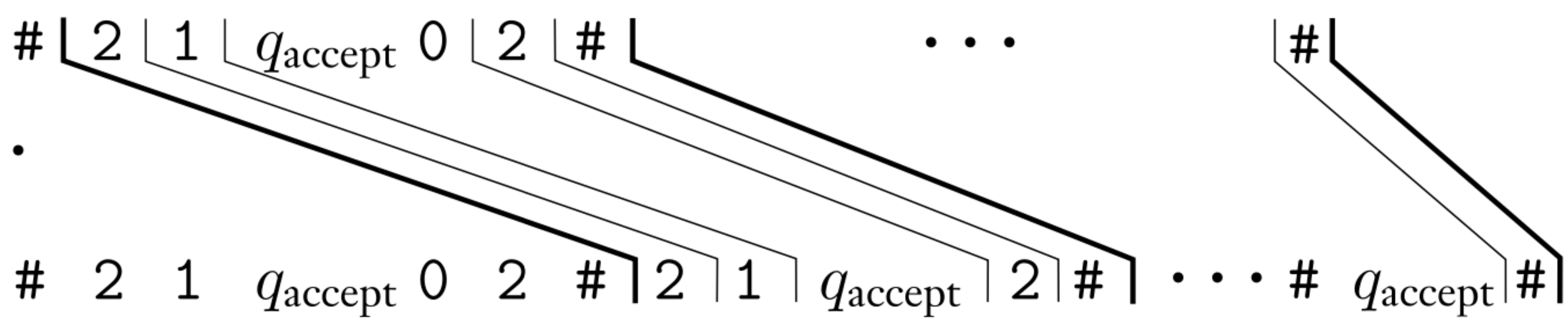
Then to allow the top to catch up we need these extra dominoes

Construction Example

- Suppose the last configuration of M is



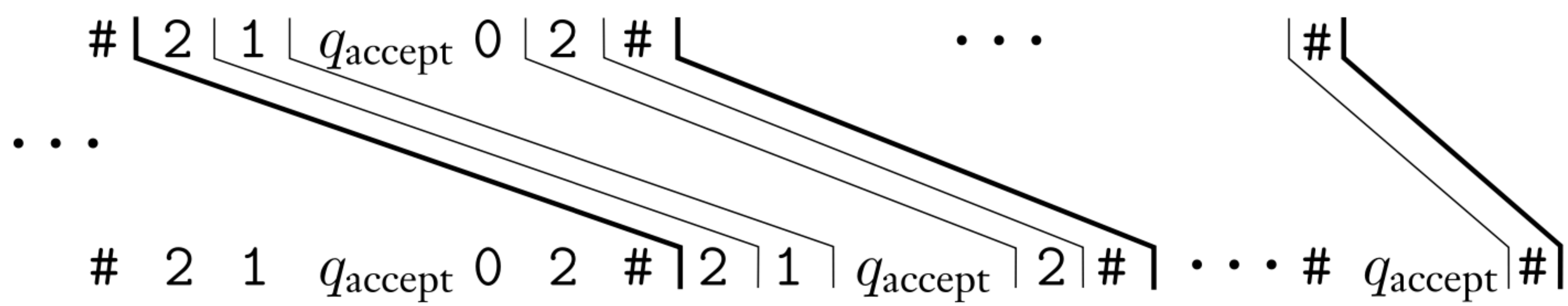
Then to allow the top to catch up we need these extra dominoes



Construction Example

- Finally, for the last step to match the extra $q_{\text{accept}}\#$ at the bottom

  add the domino $\left[\dfrac{q_{\text{accept}}\#\#}{\#}\right]$



PCP Undecidable Proof

- To complete the proof, some details remain
  - Can reduce "modified PCP" to PCP
- Consider an instance of the modified PCP:

$$\left\{ \left[\frac{t_1}{b_1}\right], \left[\frac{t_2}{b_2}\right], \left[\frac{t_3}{b_3}\right], \ldots, \left[\frac{t_k}{b_k}\right] \right\}$$

- Create an instance of PCP:

$$\left\{ \left[\frac{\star t_1}{\star b_1 \star}\right], \left[\frac{\star t_1}{b_1 \star}\right], \left[\frac{\star t_2}{b_2 \star}\right], \left[\frac{\star t_3}{b_3 \star}\right], \ldots, \left[\frac{\star t_k}{b_k \star}\right], \left[\frac{* \diamond}{\diamond}\right] \right\}.$$

where $\star\, u = * u_1 * u_2 * \cdots u_k$, $u \star = u_1 * u_2 * \cdots u_k *$ and

$\star\, u \star = * u_1 * u_2 \cdots * u_k *$

PCP to CFL Reductions

- Using PCP, we can show a bunch of questions about CFGs are undecidable:
  - **(Ambiguity)** Given a CFG G, is it ambiguous?
  - **(Disjoint)** Given two CFGs G_1, G_2, is $L(G_1 \cap G_2) = \varnothing$?
  - **(Disjoint Regular)** Given two CFGs G_1, G_2, is $L(G_1 \cap G_2)$ a regular language?
  - ... etc.