Undecidable problems about CFL’s

Deepak D’Souza

Department of Computer Science and Automation
Indian Institute of Science, Bangalore.

22 November 2019
Some Decidable/Undecidable problems about CFL’s
Problem (a)

Is it decidable whether a given CFG accepts a non-empty language?

Yes, it is. We can find out which non-terminals of $G$ can derive a terminal string: i.e. there exists a derivation $X^* \Rightarrow w$ for some terminal string $w$. Maintain a set of "marked" non-terminals. Initially $N_{\text{marked}} = \emptyset$. Mark all non-terminals $X$ such that $X \rightarrow w$ is a production in $G$. Repeat until we are unable to mark any more non-terminals: Mark $X$ if there exists a production $X \rightarrow \alpha$ such that $\alpha \in (A \cup N_{\text{marked}})^*$. Return "Non-empty" if $S \in N_{\text{marked}}$, else return "Empty."
Problem (a)

Is it decidable whether a given CFG accepts a non-empty language?

Yes, it is. We can find out which non-terminals of \( G \) can derive a terminal string: i.e. there exists a derivation \( X \xrightarrow{*} w \) for some terminal string \( w \).

- Maintain a set of “marked” non-terminals. Initially \( N_{marked} = \emptyset \).
- Mark all non-terminals \( X \) such that \( X \rightarrow w \) is a production in \( G \).
- Repeat until we are unable to mark any more non-terminals:
  - Mark \( X \) if there exists a production \( X \rightarrow \alpha \) such that \( \alpha \in (A \cup N_{marked})^* \).
- Return “Non-empty” if \( S \in N_{marked} \), else return “Empty.”
Problem (b)

Is it decidable whether a given CFG accepts a finite language?
Problem (b)

Is it decidable whether a given CFG accepts a finite language?

Yes, it is.

- Convert $G$ to CNF.
- Check if there is a parse tree within a height of $3n$, where $n$ is the number of non-terminals in $G$, that contains a pump.

$L(G)$ is infinite iff such a parse tree exists. (Essentially, since each basic pump is bounded by height $2n$.)
Problem (c)

Is it decidable whether a given CFG $G$ is universal. That is, is $L(G) = A^*$?
Problems about CFL’s

Problem (c)

Is it decidable whether a given CFG $G$ is universal. That is, is $L(G) = A^*$?

No, it is undecidable (not even r.e.).
Some Decidable/Undecidable problems about CFL’s

Undecidability of universality of a CFL

- We can reduce $\neg$HP to the problem of universality of a CFG:

  \[ \neg HP \leq \text{Universality of CFG}. \]

- Given a TM $M$ and input $x$, we can construct a CFG $G_{M,x}$ over an input alphabet $\Delta$ such that

  \[ M \text{ does not halt on } x \text{ iff } G_{M,x} \text{ is universal (i.e. } L(G_{M,x}) = \Delta^*). \]

- Hence the problem is non-r.e.
Let $M = (Q, A, \Gamma, s, \delta, \vdash, b, t, r)$ be a given TM and let $x = a_1a_2\cdots a_n$ be an input to it.

We can represent a configuration of $M$ as follows:

\[
\vdash b_1 \ b_2 \ b_3 \ \cdots \ b_m \\
- \ - \ q \ - \ -
\]

Thus a configuration is encoded over the alphabet $\Gamma \times (Q \cup \{-\})$. 
Encoding computations of $M$ on $x$

A computation of $M$ on $x$ is a string of the form

$$c_0 \# c_1 \# \cdots \# c_N \#$$

such that

1. Each $c_i$ is the encoding of a configuration of $M$.
2. $c_0$ is (encoding of) the start configuration of $M$ on $x$.

$$\vdash a_1 a_2 a_3 \cdots a_n$$

3. All $c_i$’s are of same length, and maximal (in at least one config the head is at the last position).
4. Each $c_i \xRightarrow{1} c_{i+1}$, and
5. $c_N$ is a halting configuration (i.e. state component is $t$ or $r$).
Some Decidable/Undecidable problems about CFL’s

**Describing $Valcomp_{M,x}$**

The language $Valcomp_{M,x}$ over the alphabet

$$\Delta = \Gamma \times (Q \cup \{-\}) \cup \{\#\}$$

can be described as the intersection of

- $L_1 \subseteq (C \cdot \#)^*$ where $C$ is the set of valid encodings of configurations of $M$, beginning with initial config, and containing one config with a $t$ or $r$ state.
- $L_2$ which makes sure each $c_i$ is of the same length.
- $L_3 = \{c_0\#\cdots\#c_N\# \mid N \geq 1, \; c_i \xrightarrow{1} c_{i+1}\}$.

Hence $\neg Valcomp_{M,x} = \overline{L_1} \cup \overline{L_2} \cup \overline{L_3}$.

**Claim**

$\neg Valcomp_{M,x}$ is a CFL (in fact regular) and given $M$ and $x$, we can construct a PDA/CFG $G_{M,x}$ that accepts it.
Proof of claim

Claim

Given $M, x$, we can construct a PDA/CFG $G_{M,x}$ for $\neg \text{Valcomp}_{M,x}$.

- We know $\neg \text{Valcomp}_{M,x} = \overline{L_1} \cup \overline{L_2} \cup \overline{L_3}$.
- $L_1$ is regular, and $\overline{L_2}$ is a CFL ($L_2 = L_2^o \cap L_2^e$, and each is DCFL).
- $\overline{L_3}$ is a CFL
  - Claim: $c \Rightarrow d$ iff at every position $i$ the 3 symbols $c(i), c(i + 1), c(i + 2)$ in $c$ and $d(i), d(i + 1), d(i + 2)$ in $d$, are “valid” pairs of triples.
  - Example: if $(s, \vdash), (p, \vdash, R)$ is a move of $M$ then foll pair of triples is valid:

$$\langle \vdash a_1 a_2 \vdash a_1 a_2 \rangle$$

$$\langle s \quad \_ \_ \_ , \quad \_ \quad p \quad \_ \rangle$$

- So is

$$\langle a \quad b \quad c \quad a \quad b \quad c \rangle$$

$$\langle \_ \quad \_ \quad \_ , \quad \_ \quad \_ \quad \_ \rangle$$
Proof of claim

Example: if \((p, a) \rightarrow (q, b, R)\) is a move of \(M\) then foll is invalid:
\[
\left\langle a \ b \ c \quad b \ b \ c \right\rangle
\left\langle p \quad - \quad - \quad - \quad - \quad - \quad - \right\rangle
\]

So is
\[
\left\langle a \ b \ c \quad a \ b \ c \right\rangle
\left\langle - \quad - \quad - \quad - \quad - \quad - \quad - \right\rangle
\]

Thus there is a finite table of valid triples that we can compute based on \(M\).

Now use a (non-det) PDA to guess a config \(c_k\) and a position \(i\) in it, and accept if the triple at \(c_k(i)\) and \(c_{k+1}(i)\) are not valid.

So \(\overline{L_3}\) is a CFL.

Construct a PDA/CFG \(G_{M,x}\) that accepts the union of \(\overline{L_1}, \overline{L_2},\) and \(\overline{L_3}\).
Problems about CFL’s

Problem (d)

Is it decidable whether the intersection of two given CFG’s is non-empty?

No, it is undecidable. Given $M$ and $x$, describe 2 PDA’s that accept computations of the form:

$$c_0\#c_2\#\#\#c_Nc_3c_1\#$$

Here each shaded configuration is in reversed form.

PDA $M_1$ checks that each even-numbered configuration is correctly followed by the next configuration.

PDA $M_2$ checks that each odd-numbered configuration is correctly followed by the next configuration.

In fact, a DPDA can check correct consecution of consecutive even-odd (respectively odd-even) configurations.
Problem (d)

Is it decidable whether the intersection of two given CFG’s is non-empty?

No, it is undecidable. Given $M$ and $x$, describe 2 PDA’s that accept computations of the form:

```
  c_0  #  c_1  #  c_2  #  c_3  #  ...  #  c_N  #
```

Here each shaded configuration is in reversed form.

- PDA $M_1$ checks that each even-numbered configuration is correctly followed by the next configuration.
- PDA $M_2$ checks that each odd-numbered configuration is correctly followed by the next configuration.
- In fact, a DPDA can check correct consecution of consecutive even-odd (respectively odd-even) configurations.
Some Decidable/Undecidable problems about CFL’s

Other undecidable problems about CFL’s

Problem (e)
Is it decidable whether the intersection of two given CFL’s is a CFL?

Problem (f)
Is it decidable whether the complement of a given CFL is a CFL?

Problem (g)
Is it decidable whether a given CFL is a DCFL?
### Problem (e)

Is it decidable whether the intersection of two given CFL’s is a CFL?

### Problem (f)

Is it decidable whether the complement of a given CFL is a CFL?

### Problem (g)

Is it decidable whether a given CFL is a DCFL?

All undecidable. Exercise!