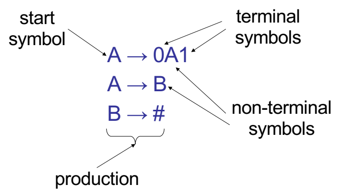
CS21 Decidability and Tractability

Lecture 6
January 17, 2025

1

Context-Free Grammars



$A \rightarrow 0A1$

$A \rightarrow B$

$B \rightarrow \#$

2

CFG example

- Balanced parentheses:
  - ( )
  - (( ) ((( ) ( ))))
- a string w in Σ* = { (,) }* is balanced iff:
  - # “(”s equals # “)”s, and
  - for any prefix of w, # “(”s ≥ # “)”s

Exercise: design a CFG for balanced parentheses.

3

CFG example

$S \rightarrow (S) \mid SS \mid \epsilon$

- Proof that w ∈ L(G) implies w is balanced
  - induction on length of derivation
  - base case: length 1: $S \Rightarrow \epsilon$
  - general case: length n
    - $S \Rightarrow (S) \Rightarrow^{n-1} (w') = w$
    - $S \Rightarrow SS \Rightarrow^{n-1} w'w'' = w$

4

CFG example

$S \rightarrow (S) \mid SS \mid \epsilon$

- Proof that w is balanced implies w ∈ L(G)
  - induction on length of w
  - base case: length 0: $w = \epsilon$
  - general case: length n
  - consider shortest prefix in language
  - if whole string then w = (w’) and w’ balanced
  - if proper prefix then w = w’w’’ with w’ and w’’ balanced

5

CFG example

- Arithmetic expressions over {+,*,(,),a}
  - (a + a) * a
  - a * a + a + a + a + a
- A CFG generating this language:

  <expr> → <expr> * <expr>
  <expr> → <expr> + <expr>
  <expr> → (<expr>) | a

6

CFG example

<expr> → <expr> * <expr>
<expr> → <expr> + <expr>
<expr> → (<expr>) | a

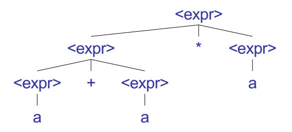
- A derivation of the string: a+a*a

  <expr> ⇒ <expr> * <expr>
  ⇒ <expr> + <expr> * <expr>
  ⇒ a + <expr> * <expr>
  ⇒ a + a * <expr>
  ⇒ a + a * a

7

Parse Trees

- Easier way to picture derivation: parse tree



- grammar encodes grouping information; this is captured in the parse tree.

8

CFGs and parse trees

<expr> → <expr> * <expr>
<expr> → <expr> + <expr>
<expr> → (<expr>) | a

- Is this a good grammar for arithmetic expressions?
  - can group wrong way (+ precedence over *)
  - different grammar for same language can force correct precedence/grouping

9

Some facts about CFLs

- CFLs are closed under
  - union (proof?)
  - concatenation (proof?)
  - star (proof?)
- Every regular language is a CFL
  - proof?

10

NPDA, CFG equivalence

Theorem: a language L is recognized by a NPDA iff L is described by a CFG.

Must prove *two* directions: **(proof next lecture!)**

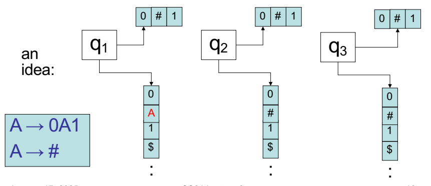
(⇒) L is recognized by a NPDA implies L is described by a CFG.

(⇐) L is described by a CFG implies L is recognized by a NPDA.

11

NPDA, CFG equivalence

Proof of (⇐): L is described by a CFG implies L is recognized by a NPDA.



A → 0A1
A → #

12

NPDA, CFG equivalence

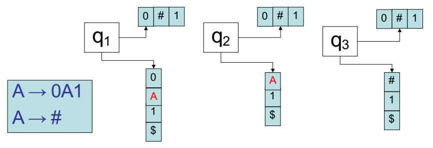
1. we’d like to non-deterministically guess the derivation, forming it on the stack
2. then scan the input, popping matching symbol off the stack at each step
3. accept if we get to the bottom of the stack at the end of the input.

what is wrong with this approach?

13

NPDA, CFG equivalence

– only have access to top of stack
– combine steps 1 and 2:
  - allow to match stack terminals with tape *during* the process of producing the derivation on the stack



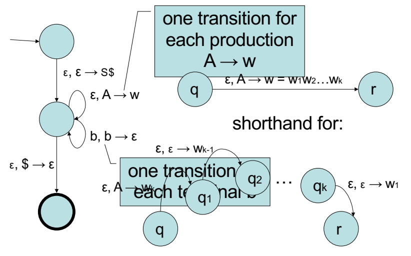
14

NPDA, CFG equivalence

- informal description of construction:
  – place $ and start symbol S on the stack
  – repeat:
    - if the top of the stack is a non-terminal A, pick a production with A on the lhs and substitute the rhs for A on the stack
    - if the top of the stack is a terminal b, read b from the tape, and pop b from the stack.
    - if the top of the stack is $, enter the accept state.

15

NPDA, CFG equivalence



16