

Outline

· deterministic PDAs

• on to Turing Machines

January 23, 2023

2

CS21 Lecture 8

L

Deterministic PDA

- A NPDA is a 6-tuple (Q, Σ, Γ, δ, q₀, F) where:
- $-\delta:Q \times (\Sigma \cup \{\epsilon\}) \times (\Gamma \cup \{\epsilon\}) \rightarrow \mathcal{P} (Q \times (\Gamma \cup \{\epsilon\}))$ is a function called the transition function
- A deterministic PDA has only one option at every step:
- for every state $q \in Q$, $a \in \Sigma$, and $t \in \Gamma$, exactly 1 element in $\delta(q, a, t)$, or
- exactly 1 element in $\delta(q, \epsilon, t)$, and $\delta(q, a, t)$ empty for all $a \in \Sigma$

January 23, 2023

CS21 Lecture 8

Deterministic PDA

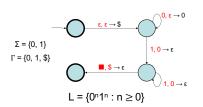
- A technical detail: we will give our deterministic machine the ability to detect end of input string
- add special symbol to alphabet
- require input tape to contain x
- language recognized by a deterministic PDA is called a deterministic CFL (DCFL)

January 23, 2023

CS21 Lecture 8

3

Example deterministic PDA



(unpictured transitions go to a "reject" state and stay there)

January 23, 2023 CS21 Lecture 8 5

Deterministic PDA

<u>Theorem</u>: DCFLs are closed under complement

(complement of L in Σ^* is $(\Sigma^* - L)$)

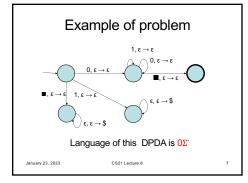
Proof attempt:

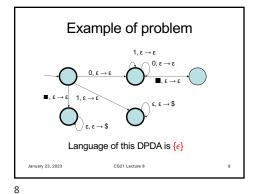
- swap accept/non-accept states
- problem: might enter infinite loop before reading entire string
- machine for complement must accept in these cases, and read to end of string

January 23, 2023

CS21 Lecture 8

5





Proof:

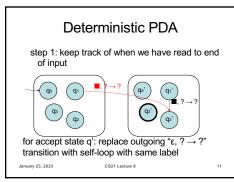
- convert machine into "normal form"

- always reads to end of input

- always enters either an accept state or single distinguished "reject" state

- step 1: keep track of when we have read to end of input

- step 2: eliminate infinite loops



Deterministic PDA

step 2: eliminate infinite loops

- add new "reject" states

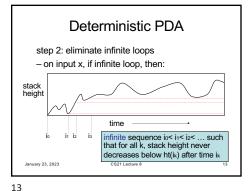
a, t \rightarrow t (for all a, t)

r

January 23, 2023

CS21 Lecture 8

12



Deterministic PDA

step 2: eliminate infinite loops

- infinite seq. $i_0 < i_1 < \dots$ such that for all k, stack height never decreases below ht(ik) after time ik
- infinite subsequence $j_0 < j_1 < j_2 < ...$ such that same transition is applied at each time jk



- finishing up...

"reject" state r'

 \bullet never see any stack symbol below $t \ from \ j_k$ on

· we are in a periodic, deterministic sequence of stack operations independent of the input

CS21 Lecture 8

Deterministic PDA

- have a machine M with no infinite loops

- therefore it always reads to end of input

- either enters an accept state q', or enters

- now, can swap: make r' unique accept state

CS21 Lecture 8

to get a machine recognizing complement of L

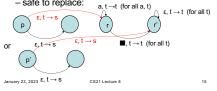
14

January 23, 2023

Deterministic PDA

step 2: eliminate infinite loops

- infinite subsequence $j_0 < j_1 < j_2 < ...$ such that
- same transition is applied at each time jk – safe to replace:



January 23, 2023

16

15

Summary

- · Nondeterministic Pushdown Automata (NPDA)
- Context-Free Grammars (CFGs) describe Context-Free Languages (CFLs)
- terminals, non-terminals
- productions
- yields, derivations
- parse trees

January 23, 2023

17

CS21 Lecture 8

Summary

- grouping determined by grammar
- Chomsky Normal Form (CNF)
- · NDPAs and CFGs are equivalent
- CFL Pumping Lemma is used to show certain languages are not CFLs

January 23, 2023

18

CS21 Lecture 8

Summary

- · deterministic PDAs recognize DCFLs
- · DCFLs are closed under complement
- there is an efficient algorithm (based on dynamic programming) to determine if a string x is generated by a given grammar G

January 23, 2023 CS21 Lecture 8

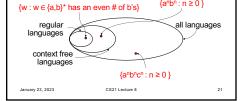
So far... · several models of computation - finite automata - pushdown automata · fail to capture our intuitive notion of what is computable regular languages context free languages January 23, 2023 CS21 Lecture 8

20

19

So far...

· We proved (using constructions of FA and NPDAs and the two pumping lemmas):



21

A more powerful machine

- · limitation of NPDA related to fact that their memory is stack-based (last in, first out)
- What is the simplest alteration that adds general-purpose "memory" to our machine?
- Should be able to recognize, e.g., {aⁿbⁿcⁿ: n ≥ 0 } CS21 Lecture 8

January 23, 2023

22

Turing Machines

input tape 0 1 1 0 0 1 1 1 0 1 0 0 ... finite read/write control head q_0

- · New capabilities:
- infinite tape
- can read OR write to tape
- read/write head can move left and right

23

CS21 Lecture 8

Turing Machine

- · Informal description:
 - input written on left-most squares of tape
 - rest of squares are blank
 - at each point, take a step determined by
 - · current symbol being read · current state of finite control
 - a step consists of
 - · writing new symbol
 - · moving read/write head left or right

CS21 Lecture 8

· changing state