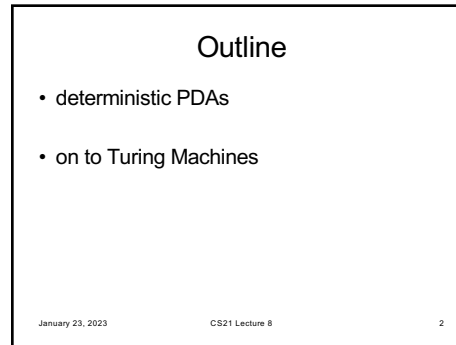
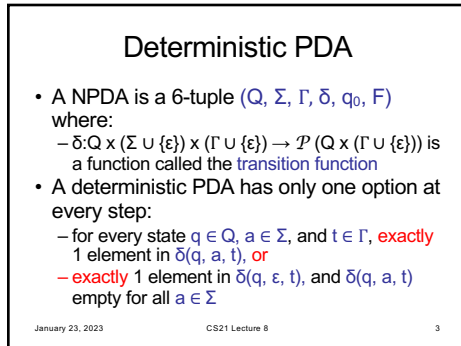




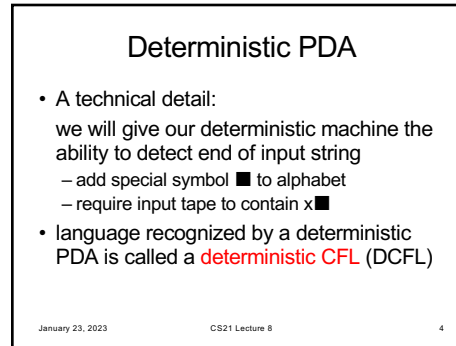
1



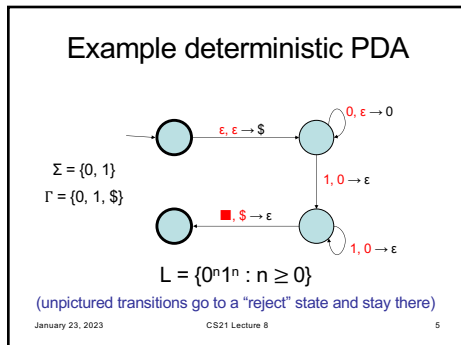
2



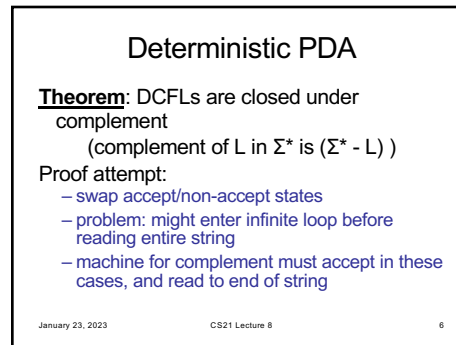
3



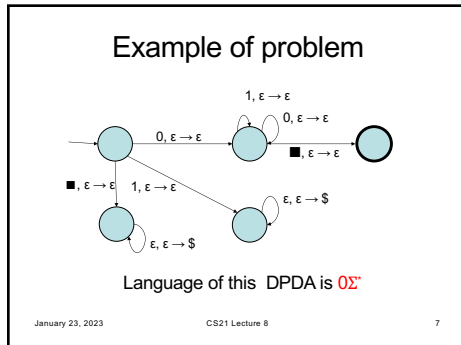
4



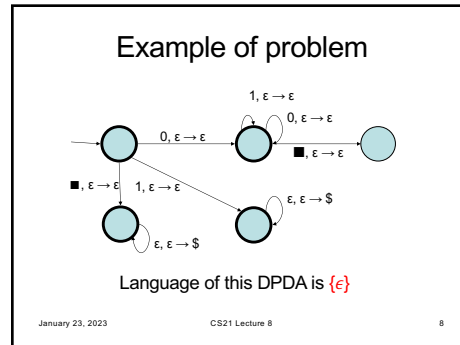
5



6



7



8

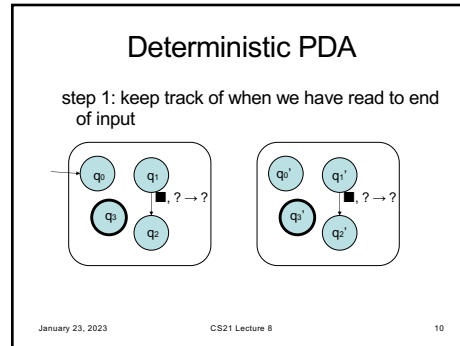
Deterministic PDA

Proof:

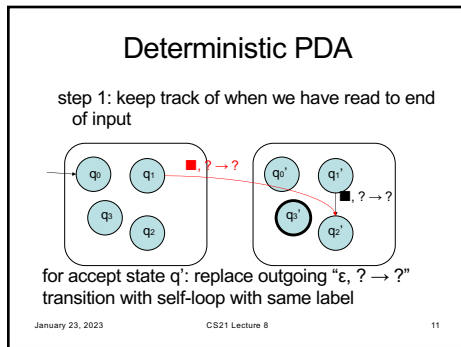
- convert machine into “normal form”
 - always reads to end of input
 - always enters either an accept state or single distinguished “reject” state
- step 1: keep track of when we have read to end of input
- step 2: eliminate infinite loops

January 23, 2023 CS21 Lecture 8 9

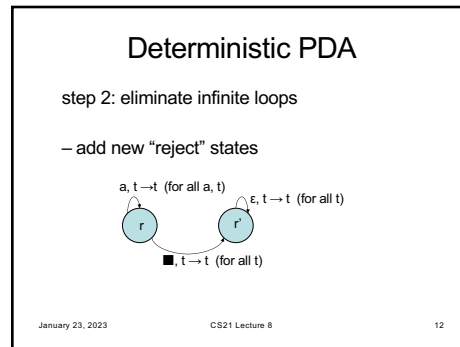
9



10



11



12

Deterministic PDA

step 2: eliminate infinite loops

– on input x , if infinite loop, then:

infinite sequence $i_0 < i_1 < i_2 < \dots$ such that for all k , stack height never decreases below $ht(i_k)$ after time i_k

January 23, 2023 CS21 Lecture 8 13

13

Deterministic PDA

step 2: eliminate infinite loops

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

- 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

January 23, 2023 CS21 Lecture 8 14

14

Deterministic PDA

step 2: eliminate infinite loops

- infinite subsequence $j_0 < j_1 < j_2 < \dots$ such that same transition is applied at each time j_k
- safe to replace:

or

January 23, 2023 CS21 Lecture 8 15

15

Deterministic PDA

- finishing up...
- have a machine M with no infinite loops
- therefore it always reads to end of input
- either enters an accept state q' , or enters "reject" state r'
- now, can swap: make r' unique accept state to get a machine recognizing complement of L

January 23, 2023 CS21 Lecture 8 16

16

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 CS21 Lecture 8 17

17

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 CS21 Lecture 8 18

18

Summary

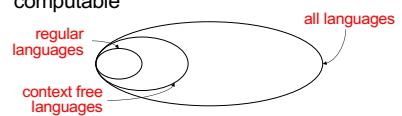
- deterministic PDAs recognize DCFLs
- DCFLs are closed under complement
- there is an efficient algorithm (based on dynamic programming) to determine if a string x is generated by a given grammar G

January 23, 2023 CS21 Lecture 8 19

19

So far...

- several **models of computation**
 - finite automata
 - pushdown automata
- fail to capture our intuitive notion of what is computable



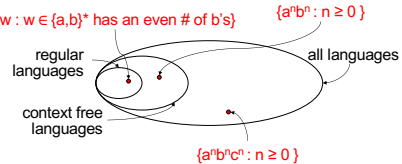
January 23, 2023 CS21 Lecture 8 20

20

So far...

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

$\{w : w \in \{a,b\}^* \text{ has an even \# of b's}\}$ $\{a^n b^n : n \geq 0\}$



January 23, 2023 CS21 Lecture 8 21

21

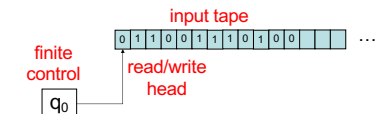
A more powerful machine

- 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., $\{a^n b^n c^n : n \geq 0\}$

January 23, 2023 CS21 Lecture 8 22

22

Turing Machines



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

January 23, 2023 CS21 Lecture 8 23

23

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
 - changing state

January 23, 2023 CS21 Lecture 8 24

24