Lecture 5

Thursday, March 2, 2023 11:50 AM

- Context free lang, pushdown automata, context-free and non langs
- ## Pushdown Automata

  1. Definitions and Examples
  2. Deterministic Pushdown Automata
  3. A PDA from a Given CFG
  4. A CFG from a Given PDA
  5. Parsing

  - FA has 5 tuple, grammar has 4 tuple, Push down has 7 tuple
- Pushdown Automata
  - Def and ex
  - Deterministic pushdown Automata
  - PDA < - > CFG (from accepting device to generating device, and vice versa)
  - Parsing
- Def and Ex
  - Lang can be generated by a CFG if & only if can be accepted by PDA
  - Pushdown automaton is similar to a FA bu has aux mem in form of a stack
  - Pushdown are nondet
  - Unlike FA's, nondet cant always be removed
  - Simple ex:
    - Lang: AnBn = {a(^n)b(^n) | n>=0}
    - Not reg lang, but its context free
  - In processing first pt of input string that might be in AnBn, all we need to remember is numb of a's
    - Saving actual a's is simple way to do this
    - So PDA start by reading a's and pushing them onto the stack
  - As soon as the PDA reads a b, two things happen
    - Enters new state which only b's are legal input
    - Pop one a off stack to cancel this b
  - In new state, correct move on input symbol b is to pop an a off the stack to cancel it
    - One enough b's read ti cancel the a's on the stack,
    -
  - Stack has no limit in size (depend on machine technically), so PDA can handle anything in AnBn
  - In single move of PDA, it depends on current state & next input & symbol currently on top of stack (cus that's all PDA can see)
  - PDA lets change states and mod top of the stack
    - Many times, only legal move is push symbol on and pop one off
  - 2 special cases:
    - Pushing symbol Y (replacing X by YX)
    - And popping X (replacing X by lambda)
  -
  - **Definition 5.1:** A *pushdown automaton* is a 7-tuple $M = (Q, \Sigma, \Gamma, q_0, Z_0, A, \delta)$, where:
    - Q is a finite set of states;
    - The *input* and *stack* alphabets Σ and Γ are finite sets;
    - $q_0 \in Q$ is the initial state;
    - $Z_0 \in \Gamma$ is the initial stack symbol;
    - $A \subseteq Q$ is the set of accepting states;
    - The transition function is

      $\delta : Q \times (\Sigma \cup \{\Lambda\}) \times \Gamma \rightarrow$ the set of finite subsets of $Q \times \Gamma^*$. ■
  -
  - **Definition 5.2:** If $M = (Q, \Sigma, \Gamma, q_0, Z_0, A, \delta)$ and $x \in \Sigma^*$, the string x is *accepted* by M if $(q_0, x, Z_0) \vdash_M^* (q, \Lambda, \alpha)$ for some $\alpha \in \Gamma^*$ and some $q \in A$. ■
    - A lang L is said to be accepted by M if L is precisely the set of strings accepted by M
    - Sometimes a string accepted by M said to be accepted by final state bc acceptance doesn't depend on final stack contents at all
  -
  - A PDA for *AnBn* is $M = (Q, \Sigma, \Gamma, q_0, Z_0, A, \delta)$ where $Q = \{q_0, q_1, q_2, q_3\}$, $A = \{q_0, q_3\}$, and the transitions are shown in this table:

Move #	State	Input	Stack top	Move(s)
1	q_0	a	Z_0	(q_1, aZ_0)

2	q_1	a	a	(q_1, aa)
3	q_1	b	a	(q_2, Λ)
4	q_2	b	a	(q_2, Λ)
5	q_2	Λ	Z_0	(q_3, Z_0)
all other combinations				none

The moves that M makes as it processes the string $aabb$ are shown below:

$$\begin{aligned}(q_0, aabb, Z_0) &\vdash^1 (q_1, abb, aZ_0)\\ &\vdash^2 (q_1, bb, aaZ_0)\\ &\vdash^3 (q_2, b, aZ_0)\\ &\vdash^4 (q_2, \Lambda, Z_0)\\ &\vdash^5 (q_3, \Lambda, Z_0).\end{aligned}$$

Since $q_3 \in A$, it follows that the string $aabb$ is accepted. M is deterministic, because it never has a choice of moves.

Get a star: Prove that $(q_0, a^nb^n, Z_0) \vdash^{2n+1 \text{ times}} (q_3, \Lambda, Z_0)$.

- Deterministic pushdown automata

Definition 5.10: A pushdown automaton $M = (Q, \Sigma, \Gamma, q_0, Z_0, A, \delta)$ is *deterministic* if it satisfies both of the following conditions:

1. For every $q \in Q$, every σ in $\Sigma \cup \{\Lambda\}$, and every $X \in \Gamma$, the set $\delta(q, \sigma, X)$ has at most one element.
2. For every $q \in Q$, every $\sigma \in \Sigma$, and every $X \in \Gamma$, the two sets $\delta(q, \sigma, X)$ and $\delta(q, \Lambda, X)$ cannot both be nonempty. ■

Note: A language L is a *deterministic context-free language* (DCFL) if there is a deterministic PDA (DPDA) accepting L. ■

- Transition table for a PDA accepting the lang of balanced strings of brackets is (A= {q0}):
  - Table, lots of pushdown stuff, all move based
  - Q0 is the intial and the accepting state
- The lang Pal of palindromes over {a,b} can be accepted by a PDA M that saves symbols on the stack until it "guesses" that it has reached the middle of the string, then cancels stack symbols with input symbols

The transition table for a PDA accepting the language of balanced strings of brackets is ($A=\{q_0\}$):

Move #	State	Input	Stack top	Move
1	q_0	[	Z_0	$(q_1, [Z_0)$
2	q_1	[	[	$(q_1, [[)$
3	q_1	]	[	(q_1, Λ)
4	q_1	Λ	Z_0	(q_0, Z_0)
(all other combinations)				none

One example is the previous PDA accepting $AnBn$. ■

Another example is the language of balanced strings of brackets described by the below three conditions:

1. Two states q_0 and q_1, where q_0 is the accepting state
2. Input symbols are '[' and ']'
3. Stack symbols are the input symbols plus Z_0

- 2 typical lines from the transition table are
    - (q0, a, Z0) = {(q1, aZ0), (q1, Z0)}
    - (q0, a,b) = {(q0, ab), (q1, b)}
  - For both lines, first move is the right one, if input symbol a is still in the first half of the string
  - 2nd move is right one if a is in middle symbol in an odd length palindrome
  - M decides which to make by guessing
  - Other guess it can make is that it has reached middle of an even-length palindrome
    - W/ this, M enter q1 by a lambda transition
    - EX:

- Pushdown Automaton accepting Pal
  - … delta is
  - This stuff
  - More stuff
  - $M_{pal} = (\{q_0, q_1, q_2\}, \{a, b\}, \{a, b, Z_0\}, q_0, Z_0, \{q_2\}, \delta)$, where δ is:
    1. $\delta(q_0, a, Z_0) = \{(q_0, aZ_0), (q_1, Z_0)\}$
    2. $\delta(q_0, a, a) = \{(q_0, aa), (q_1, a)\}$
    3. $\delta(q_0, a, b) = \{(q_0, ab), (q_1, b)\}$
    4. $\delta(q_0, b, Z_0) = \{(q_0, bZ_0), (q_1, Z_0)\}$
    5. $\delta(q_0, b, a) = \{(q_0, ba), (q_1, a)\}$
    6. $\delta(q_0, b, b) = \{(q_0, bb), (q_1, b)\}$
    7. $\delta(q_0, \Lambda, Z_0) = \{(q_1, Z_0)\}$
    8. $\delta(q_0, \Lambda, a) = \{(q_1, a)\}$
    9. $\delta(q_0, \Lambda, b) = \{(q_1, b)\}$
    10. $\delta(q_1, \Lambda, Z_0) = \{(q_2, Z_0)\}$
    11. $\delta(q_1, a, a) = \{(q_1, \Lambda)\}$
    12. $\delta(q_1, b, b) = \{(q_1, \Lambda)\}$
  - Let us see whether *abba* is accepted by *M* (even length).
    - $(q_0, abba, Z_0) \vdash_{1a} \quad (q_0, bba, aZ_0) \vdash$
    - $\vdash_{5a}(q_0, ba, baZ_0) \vdash_{9}(q_1, ba, baZ_0) \vdash$
    - $\vdash_{12}(q_1, a, aZ_0) \quad \vdash_{11}(q_1, \Lambda, Z_0) \vdash$
    - $\vdash_{10}(q_2, \Lambda, Z_0)$.
    - Since q_2 is an accepting state, it follows that the string *abba* is accepted.
  - Pal, somethings are nondet
  - Check If abba is accepted by M (even length)
    - Reduce it using the rules for pal
  - Check is ababa is accepted by M
    - Odd length, guess the middle, then reduce
  - M even pal =
  - Check abba
    - So q1 deals with first half, q2 deals with 2nd half, q3 is end
  - M odd pal =
  - M even_odd_pal =
  - CFG making Pal
    - Even: Pal: S->aSa|bSb|lambda
    - Odd: Pal: S-> aSa|bSb|a|b
    - Both: Pal: S-> aSa|bSb|lambda|a|b
    - Now construction of equivalent NPDA for Pal is:
    - Where delta is:
    - Go to slide 40
- A CFG from a Given PDA

        -
        - Ex:
- Parsing
    - Def
    - Consider the CFG w/ rules S -> [S]|SS|lambda
        - Top down PDA, get non-det
        - Each time var appear on top of stack, machine replaces it by right side of a production
    - First few moves made for the input string [[][]]
        - Snake
    - Try diff lang, S->[S]S | lambda
        - Got same lang, and unambiguous
    -