COL352 Lecture 16

Contents

1 Recap

Discussion about grammars, parse trees, yields, and context free languages.

2 Definitions

Definition 1

A (non-deterministic) pushdown automaton ((N)PDA) is a 6-tuple $(Q, \Sigma, \Gamma, \Delta, q_0, A)$ where

1. Q – finite nonempty set of states
2. Σ – finite nonempty input alphabet
3. Γ – finite stack alphabet
4. $q_0 \in Q$ – initial state
5. A – set of accepting states
6. $\Delta \subseteq Q \times \Sigma_\epsilon \times \Gamma_\epsilon \times Q \times \Gamma_\epsilon$, where X_ϵ is defined as $X \cup \{\epsilon\}$

Note that in an NFA, $\Delta \subseteq Q \times (\Sigma \cup \{\epsilon\}) \times Q$.
Informally, a string is accepted if $\exists$ a run which reads the entire string and does what ?????

Definition 2

Let $P = (Q, \Sigma, \Gamma, \Delta, q_0, A)$ be a PDA. An instantaneous description (i.d.) of P is a tuple (q, x, α) where $q \in Q$, $x \in \Sigma^*$, $\alpha \in \Gamma^*$. (The set of instantaneous descriptions is $Q \times \Sigma^* \times \Gamma^*$).

Informally, it consists of the current state, the string left to be read, and the description of the current stack.

Definition 3

Let $P = (Q, \Sigma, \Gamma, \Delta, q_0, A)$ be a PDA. The relation $\vdash_P$ (read as "changes to") is defined on the set of i.d.s as follows:

If $(q, a, B, q', B') \in \Delta$, then $(q, ax, B\alpha) \vdash_P (q', x, B'\alpha)$, and no other pairs of i.d.s are related.

In other words:

$$(q, y, \beta) \vdash_P (q', y', \beta') \iff \exists a \in \Sigma_\epsilon, B \in \Gamma_\epsilon, \alpha \in \Gamma^*, B' \in \Gamma_\epsilon \text{ such that } y = ay', \beta = B\alpha, \beta' = B'\alpha, (q, a, B, q', B') \in \Delta$$

Definition 4

$\vdash_P^*$ is defined as the reflexive transitive closure of $\vdash$ (read as "changes to in finitely many steps").

Definition 5

$x \in \Sigma^*$ is said to be accepted by PDA $P = (Q, \Sigma^*, \Gamma, \Delta, q_0, A)$ iff

$$(q_0, x, \epsilon) \vdash_P^* (q, \epsilon, \alpha)$$

for some $q \in A$ and some $\alpha \in \Gamma^*$.

3 Content

We begin with an informal discussion first.

Note that we look at two kinds of objects - languages and recognizers.

Language Class	Recognizer	Generator
Regular	DFA/NFA	Regular expressions
Context-free	Non deterministic pushdown automaton	Grammar

Note that unlike DFA/NFAs (which have the same power), NPDAs are more powerful that DPDAs.

An NFA looks like a tape where we have a read head pointing to a location in the string, with some state associated to the read head, and transitions can either go to the next position (in the string) or remain at the same position.

A PDA is similar to this, except that it has a stack as well.

(q, a, B, q', B') is a transition in a PDA.

1. q is the current state.
2. a is the current input.
3. B is the top of the stack that can be popped (ϵ if not popping).
4. q' is the next state.
5. B' is the top of the stack after pushing B' on the stack (ϵ if we don't push anything on the stack).

Claim 0.1

A PDA can recognize $\{a^n b^n \mid n \in \mathbb{N} \cup \{0\}\}$.

Intuition: try to think in terms of the stack. When is it empty? Do we need it to be empty?
See definitions section for the formal definition.

Example 1

$\Sigma = \{0, 1\}$, L is the set of all palindromes. $\Gamma = \{0, 1, \perp\}, Q = \{q_0, q_1, q_2, q_3\}$. Define in the usual sense (see lecture).

Claim 0.2

$x \in L \iff (q_0, x, \epsilon) \vdash_P^* (q_3, \epsilon, \epsilon)$.

Proof. Exercise. □