Introduction to Language Theory and Compilation: Exercises

Session 5: Pushdown automata and parsing

ULB

Pushdown automata (PDA)

A pushdown automaton is described by 7 components:

$$M = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F \rangle$$

- Q is the set of states
- Σ is an *input alphabet*
- Γ is a *stack alphabet*
- δ is the transition function
- q_0 is the start state
- Z_0 is the start symbol on the stack (may be ε)
- F is the set of accepting states.
- Accept if finish at at accepting state OR have empty stack

ULB

Accepting State

A PDA defines two languages:

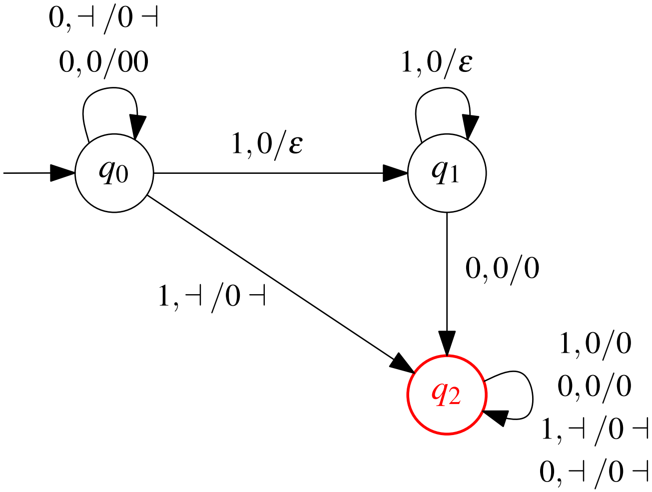
- $N(P)$ Accept by empty stack (i.e. $F = \emptyset$)
- $L(P)$ Accept by reaching a final state

Expression power of both languages is equal

PDA example

- Let's consider the language $\{0^n 1^n \mid n \geqslant 0\}$.
- Let's design a PDA that accepts it.
  - The automaton may be nondeterministic
  - The stack will help us count symbols
  - We'll choose to accept by empty stack

PDA example (accepting by empty stack)



Exercise 1

Design a pushdown automaton that accepts the language made of all words of the form ww^R where w is any given word on the alphabet $\Sigma = \{a, b\}$ and w^R is the mirror image of w.

Top-down parser

- A top-down parser builds a parse tree using a top-down approach.
- A given grammar $G = \langle V, T, P, S \rangle$ will be assimilated to the following PDA:

$$M = \langle \{q\}, T \cup \{\$\}, V \cup T \cup \{\$\}, \delta, q, S \rangle$$

- M only has a single state (the start state)
- Its input alphabet includes \$, which denotes the end of the input
- The stack is initialized with the grammar's start symbol ($S \dashv$)

ULB

Top-down parser (ctd.)

- We define the configuration of the PDA by a tuple: (q, i, S, o)
  - q is the current state
  - i is the remaining input
  - S is the current stack state
  - o is the output so far (we output production rule numbers as they are applied)
- Starting configuration: $(q, I, S \dashv, \varepsilon)$ where I is the complete input
  1. State: q (unique state)
  2. Input left to parse: whole input I
  3. Stack contents: start symbol of the grammar($S \vdash$)
  4. Output so far: empty (ε)

ULB

Top-down parser (ctd.)

There are three kinds of transitions in the transition function δ:

- Match: $(q, ax, a\gamma, y) \rightarrow (q, x, \gamma, y)$: we match the top of the stack with the next input symbol and remove both
- Produce: $(q, x, A\gamma, y) \rightarrow (q, x, \alpha\gamma, yi)$ where production rule i has the form $A \rightarrow \alpha$: we replace a variable A on top of the stack with its production α
- Accept: $(q, \$, \$ \dashv, y) \rightarrow (q, \varepsilon, \dashv, y)$: we match the "end of input" symbols and signal that we accept the given input

ULB

Exercise 2

Using the grammar given on paper,

2. Give the parse tree for the following input:

```
begin ID := ID - INTLIT + ID ; end
```

Exercise 3

Using the grammar given on paper,

3. Simulate a top-down parser on the following input:

   `begin A := BB - 314 + A ; end`

Bottom-up parser

- A bottom-up parser builds a parse tree using a bottom-up approach
- A given grammar $G = \langle V, T, P, S \rangle$ will be assimilated to the following PDA:

$$M = \langle \{q\}, T \cup \{\$\}, V \cup T \cup \{\$\}, \delta, q, \varepsilon \rangle$$

- We start with an empty stack.

Bottom-up parser (ctd.)

There are three kinds of transitions in the transition function δ:

- Shift: $(q, \alpha x, \gamma, y) \to (q, x, \gamma\alpha, y)$: push the next input symbol on the stack
- Reduce: $(q, x, \gamma\alpha, y) \to (q, x, \gamma A, yi)$ if rule i had the form $A \to \alpha$: replace the corresponding input α by the corresponding symbol A on the stack, without touching the input
- Accept: $(q, \varepsilon, \vdash S, y) \to (q, \varepsilon, \varepsilon, y)$: we accept the input if we manage to get to the end of the input with the start symbol on the stack

ULB

Exercise 4

Using the grammar given on paper,

4. Simulate a bottom-up parser on the same input.