COMP30026 Models of Computation

Lecture 15: Pushdown Automata

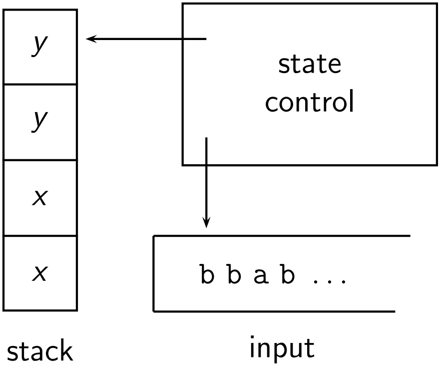
Mak Nazecic-Andrlon and William Umboh

Semester 2, 2024

Pushdown Automata

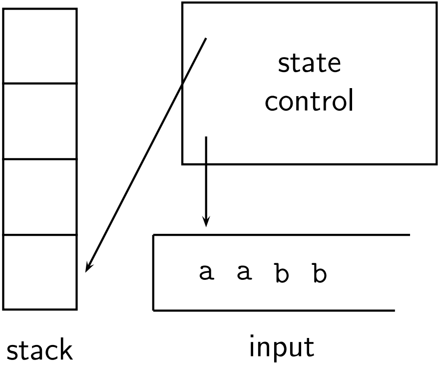
DFAs and NFAs have limited memory.

Pushdown automaton (PDA): NFA with unlimited stack.



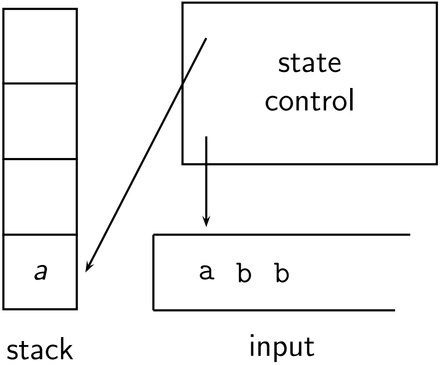
Pushdown Automata

The language $\{a^n b^n \mid n \geq 0\}$ cannot be recognised by a DFA, since the recogniser must remember how many consecutive as have been consumed from the input. But a PDA can recognise it.



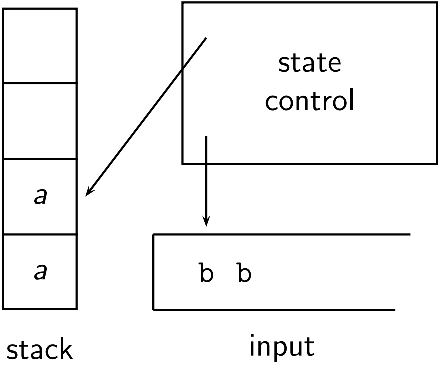
Pushdown Automata

The language $\{a^n b^n \mid n \geq 0\}$ cannot be recognised by a DFA, since the recogniser must remember how many consecutive as have been consumed from the input. But a PDA can recognise it.



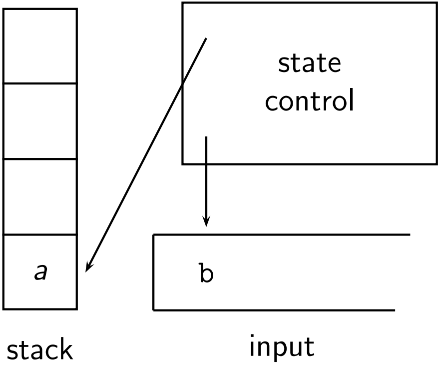
Pushdown Automata

The language $\{\mathrm{a}^n\mathrm{b}^n \mid n \geq 0\}$ cannot be recognised by a DFA, since the recogniser must remember how many consecutive as have been consumed from the input. But a PDA can recognise it.



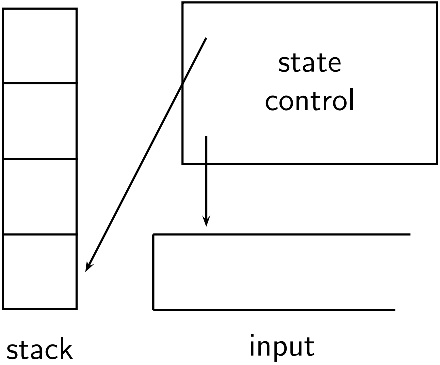
Pushdown Automata

The language $\{a^n b^n \mid n \geq 0\}$ cannot be recognised by a DFA, since the recogniser must remember how many consecutive as have been consumed from the input. But a PDA can recognise it.



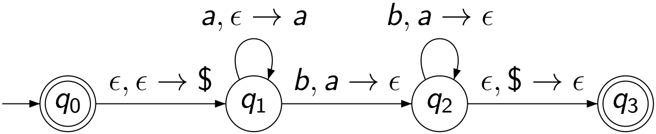
Pushdown Automata

The language $\{a^n b^n \mid n \geq 0\}$ cannot be recognised by a DFA, since the recogniser must remember how many consecutive as have been consumed from the input. But a PDA can recognise it.



PDA Example 1

This PDA recognises $\{a^n b^n \mid n \geq 0\}$:



Transition notation:

$$\text{input}, \text{top of stack} \to \text{new top of stack}$$

How PDAs Work

Transitions based on 3 things:

1. Current state
2. Input symbol
3. Top of stack

Operation of PDA:

1. Start in start state.
2. While input remains or not in a final state:
    1. Nondeterministically pick a valid transition $a, b \to c$ from current to new state q. Reject if unable.
    2. Consume input a.
    3. Replace b on top of stack with c.
    4. Move to state q.
3. Accept.

Pushdown Automata Formally

A pushdown automaton is a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$ where

- Q is a finite set of states,
- Σ is the finite input alphabet,
- Γ is the finite stack alphabet,
- $\delta : Q \times \Sigma_\epsilon \times \Gamma_\epsilon \to \mathcal{P}(Q \times \Gamma_\epsilon)$ is the transition function,
- $q_0 \in Q$ is the start state, and
- $F \subseteq Q$ are the accept states.

Example Transitions

$\delta(q_5, \mathtt{a}, \mathtt{b}) = \{(q_7, \epsilon)\}$ means:

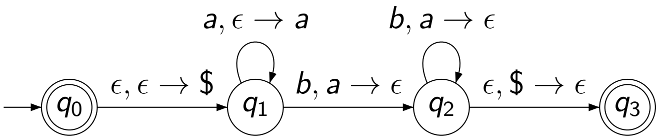
If in state q_5, when reading input symbol a, provided the top of the stack holds 'b', consume the a, pop the b, and go to state q_7.

$\delta(q_5, \epsilon, \mathtt{b}) = \{(q_6, \mathtt{a}), (q_7, \mathtt{b})\}$ means:

If in state q_5, and if the top of the stack holds 'b', either replace that b by a and go to state q_6, or leave the stack as is and go to state q_7. In either case do not consume an input symbol.

PDA Example 1

This PDA recognises $\{a^n b^n \mid n \geq 0\}$:



- $Q = \{q_0, q_1, q_2, q_3\}$;
- $\Sigma = \{a, b\}$;
- $\Gamma = \{a, \$\}$;
- $\delta(q_0, \epsilon, \epsilon) = \{(q_1, \$)\}$,$\delta(q_1, a, \epsilon) = \{(q_1, a)\}$,
  $\delta(q_1, b, a) = \{(q_2, \epsilon)\}$,$\delta(q_2, b, a) = \{(q_2, \epsilon)\}$,
  $\delta(q_2, \epsilon, \$) = \{(q_3, \epsilon)\}$, for other inputs δ returns $\emptyset$;
- $q_0 = q_0$;
- $F = \{q_0, q_3\}$.

Acceptance Precisely

The PDA $(Q, \Sigma, \Gamma, \delta, q_0, F)$ accepts input w iff $w = v_1 v_2 \cdots v_n$ with each $v_i \in \Sigma_\epsilon$, and there are states $r_0, r_1, \ldots, r_n \in Q$ and strings $s_0, s_1, \ldots, s_n \in \Gamma^*$ such that

1. $r_0 = q_0$ and $s_0 = \epsilon$.
2. $(r_{i+1}, b) \in \delta(r_i, v_{i+1}, a)$, $s_i = at$, $s_{i+1} = bt$ with $a, b \in \Gamma_\epsilon$ and $t \in \Gamma_\epsilon^*$.
3. $r_n \in F$.

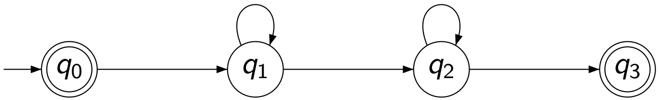
Note 1: There is no requirement that $s_n = \epsilon$, so the stack may be non-empty when the machine stops (even when it accepts).

Note 2: Empty stack cannot be popped.

PDA Example 2

Let $w^{\mathcal{R}}$ denote the string w reversed.

Let us design a PDA to recognise $\{ww^{\mathcal{R}} \mid w \in \{0,1\}^*\}$, the set of even-length binary palindromes:



PDA Example 2

This PDA recognises $\{ww^{\mathcal{R}} \mid w \in \{0,1\}^*\}$: