Pushdown automata

March 27, 2023

A pushdown automata is a type of automation that use a stack structure, with the remaining part the same as the NFA (non-definite finte state automata). With the repect to the stack the pushdown automata has gained a more greater ability to describe the language. We will in the further section prove that pushdown automata has just the same ability describing as Context free grammar.

1 The definition of PDA

What is new in the pushdown automata is that there is a stack that we can operate when we receive a character. When it receive a character of Σ, the automata is going to next state based on three object: 1. the current state; 2. the received character of Σ; 2. the character on the top of the stack.

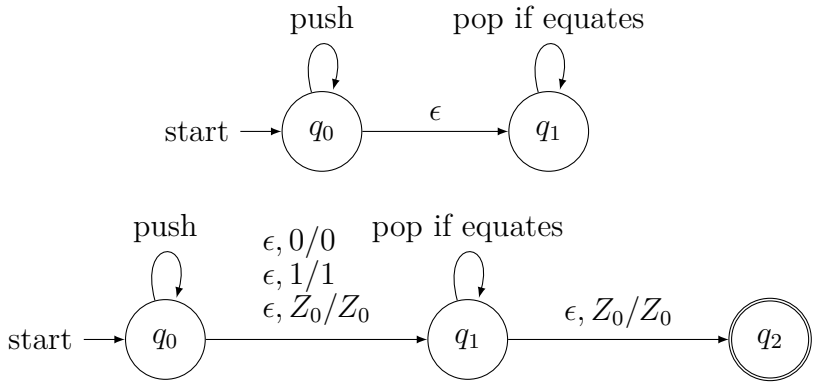
If you wrote out the defintion of state-transition function, it would be something like: $\delta\colon Q \times \Sigma \times Z \to Q$, where Z denote the character in the stack (and of course that Z can equate Q).

Let's take the pushdown automata that describe the language $L = \{ww^R\}$ as an example.

Example 1.1. *It is clear that we should push a into the stack if we receive a, before we finish going through the string w. And it is the same clear that we should get the top of the stack and check if it is the same as the received character after we go through the string w.*

The tricky one is that we have to check these two kinds of situation at the same time, for that every time we get a character, it could be that the w is done or not.

We can use NFA with two state to construct the pushdown automaton we need.



We say that the automata is at q_0 saying that we are at w, and the state q_1 is saying that we are at w^R. And before we receive a character we use a ϵ transition from q_0 to q_1, after figuring out which, all is clear.

1.1 The precise definition of the pushdown automata

A pushdown automaton is a seven tuple: $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$.

> Γ is the set of the character that appears in the stack
>
> Z_0 is the initial character that located in the stack.
>
> Moreover, the state-transition function should be like:
>
> $$\delta_0 \colon Q \times \Sigma \times \Gamma \to \mathfrak{P}(Q)$$
>
> for that the the pushdown automata is built on a NFA.

Also, the element on the top of the stack is always popped. If you want the stack to remain un-changed, you can define something like $\delta(q_i, a, \alpha) = \{(q_j, \alpha)\}$

Example 1.2. Let us just skip the blahblah and draw a diagram that shows how a pushdown automaton work.

1.2 Instantaneous Descriptions of a Pushdown automata

We introduce the instantaneous descriptions of PDA to show how a PDA accept a string.

A string is given and the string is the input of the PDA. What will happen in the PDA is that part of the string is received and

state changes and stack is pushed into character or the other way. Then the information in the middle of the procedure contains three parts: 1. the state; 2. the string in the stack; 3. the remaining string.

1. the state
2. the string in stack
3. the remaining string

So we can describe the middle state of the whole PDA with a three tuple—(q, w, γ), where $q \in Q$, $w \in \Sigma^*$, $\gamma \in \Gamma^*$. We use $\vdash$ to indicate that an ID can transit to the other. Then we know that if $\delta(q, a, X)$ contains (p, α) we will know that

$$(q, aw, X\beta) \vdash (p, w, \alpha\beta)$$

It is very similar to the function $\hat{\delta}$ in the finite state automata. And it is very similar to the symbol $\Rightarrow$ in the context free grammar. Similarly we use $\overset{*}{\vdash}$ to indicate that one ID can transit to the other after zero or one or more than one character are received.

If $(q_1, w_1, \gamma_1) \vdash^* (q_2, w_2, \gamma_2)$ then $(q_1, w_1, \gamma_1) \sim (q_2, w_2, \gamma_2)$

Theorem 1.3 (Deduction principle)**.** $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ is a PDA. If $(q, x, \alpha) \vdash^* (p, y, \beta)$, then

$$(q, xw, \alpha\gamma) \vdash^* (p, yw, \beta\gamma)$$

Proof. The proof is trivial. □

However, it is worth noting that the converse of the thm is not true.