

COMP30026 Models of Computation

Pushdown Automata

Assignment Project Exam Help

<https://eduassistpro.github.io>

Lecture Week 9

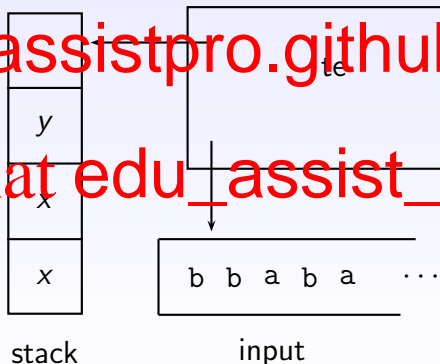
Add WeChat Semester 2, 2020 edu_assist_pro

Pushdown Automata

The automata we saw so far were limited by their **lack of memory**.

Assignment Project Exam Help
A **pushdown automaton** (PDA) is a finite-state automaton, equipped with a **s**

The language is not recognised since it requires the ability of a recogniser to remember how many consecutive **a**s have been consumed from the input.

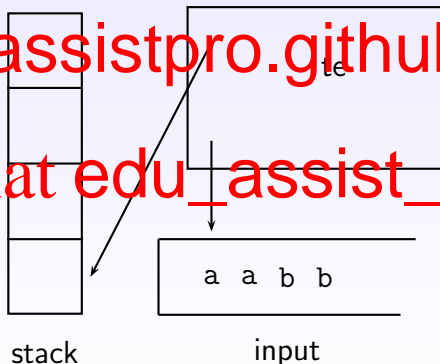


Pushdown Automata

The automata we saw so far were limited by their lack of memory.

Assignment Project Exam Help
A pushdown automaton (PDA) is a finite-state automaton, equipped with a stack.

The language is not recognised since it requires the ability of a recogniser to remember how many consecutive 'a's have been consumed from the input.

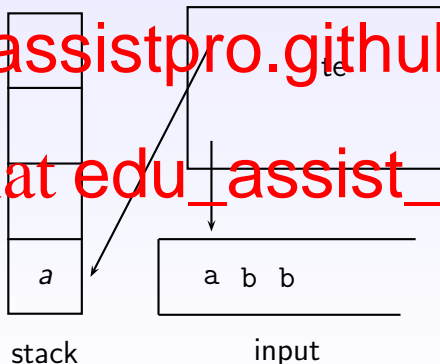


Pushdown Automata

The automata we saw so far were limited by their lack of memory.

Assignment Project Exam Help
A pushdown automaton (PDA) is a finite-state automaton, equipped with a stack.

The language is not recognised since it requires the ability of a recogniser to remember how many consecutive 'a's have been consumed from the input.

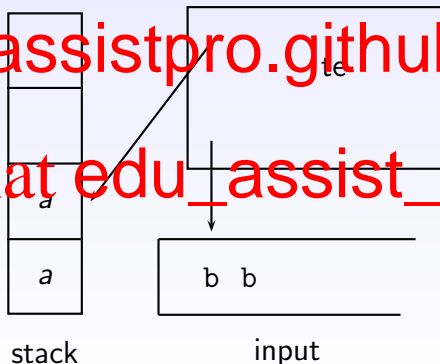


Pushdown Automata

The automata we saw so far were limited by their lack of memory.

Assignment Project Exam Help
A pushdown automaton (PDA) is a finite-state automaton, equipped with a stack.

The language is not regular since it requires the ability of a recogniser to remember how many consecutive 'a's have been consumed from the input.

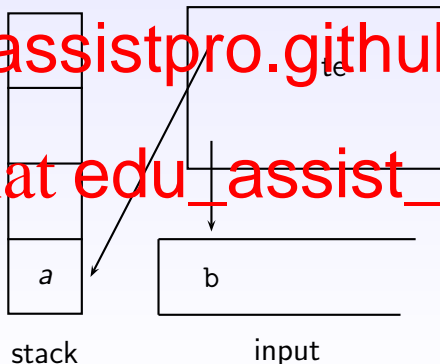


Pushdown Automata

The automata we saw so far were limited by their lack of memory.

Assignment Project Exam Help
A pushdown automaton (PDA) is a finite-state automaton, equipped with a stack.

The language is not recognised since it requires the ability of a recogniser to remember how many consecutive 'a's have been consumed from the input.

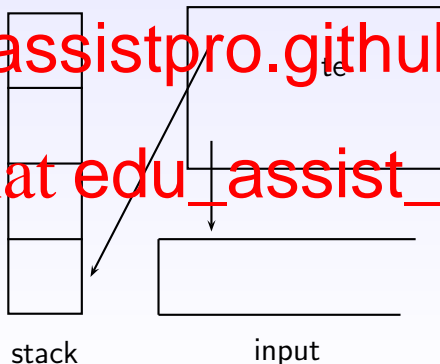


Pushdown Automata

The automata we saw so far were limited by their lack of memory.

Assignment Project Exam Help
A pushdown automaton (PDA) is a finite-state automaton, equipped with a stack.

The language is not recognised since it requires the ability of a recogniser to remember how many consecutive 'a's have been consumed from the input.



Fine but Important Points

Based on (1) input symbol, (2) top stack symbol and (3) the current state, PDA will decide which state to go to next, as well as, what operation

In one transition (the top stack symbol, the top stack symbol).

We shall consider the non-deterministic

It may also ignore the input.

Assignment Project Exam Help

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

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

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

Assignment Project Exam Help

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

If in state
stack holds

$\delta(q_5, \epsilon,$ 6 7

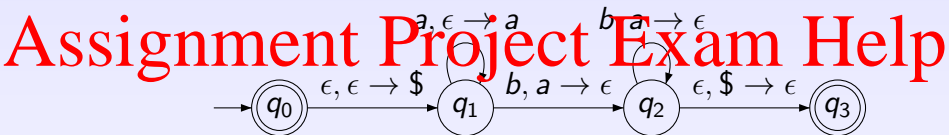
If in state q_5 , and if the top of the stack holds 'b' by a and go to state q_6 , or leave the stack unchanged that q_7 .
In either case do not consume an input symbol.

<https://eduassistpro.github.io>

Add WeChat [edu_assist_pro](https://eduassistpro.github.io)

PDA Example 1

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



• Q

• $\Sigma = \{a, b\};$

• $\Gamma = \{a, \$\};$

• $\delta(q_0, \epsilon, \epsilon) = \{(q_1, \$)\}, \delta(q_1, a, \epsilon) =$

$\delta(q_1, b, a) = \{(q_2, \epsilon)\}, \delta(q_2, b, a) = \{(q_2, \epsilon)\},$

$\delta(q_2, \epsilon, \$) = \{(q_3, \epsilon)\},$ for other inputs δ returns \emptyset ;

• $q_0 = q_0;$

• $F = \{q_0, q_3\}.$

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

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 $r_0, r_1, \dots, r_n \in Q$ and strings $s_0, s_1, \dots, s_n \in \Gamma^*$ such that

- 1 r_0
- 2 $(r_i, t) \in \delta_\epsilon$ and $t \in \Sigma_\epsilon$
- 3 $r_n \in F$.

Note 1: There is no requirement that non-empty when the machine stops (even when it accepts).

Note 2: Trying to pop an empty stack leads to rejection of input, rather than “runtime error”.

PDA Example 2

Let w^R denote the string w reversed.

Assignment Project Exam Help

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

<https://eduassistpro.github.io>

Add WeChat edu_assist_pr



Assignment Project Exam Help

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

<https://eduassistpro.github.io>



CFLs Have PDAs as Recognisers

Given a context-free language L (in the form of a grammar), we can find a PDA which recognises L .

And, every PDA recognises a context-free language.

We won't

seen to

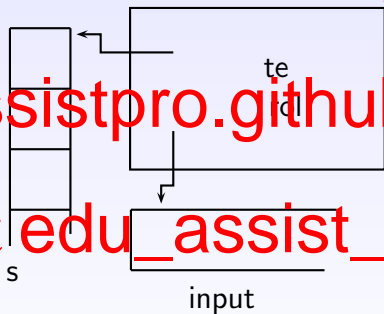
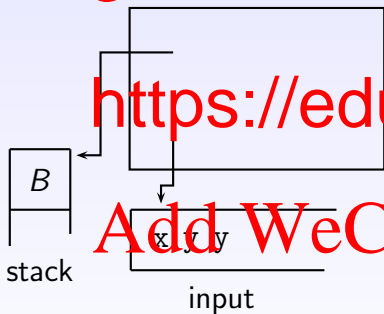
Namely, given a CFG G , we show how to construct
such that $L(P) = L(G)$.

The idea is to let the PDA use its stack to store a list of “pending”
recogniser tasks.

The construction does not give the cleverest PDA, but it always works.

From Context-Free Grammars to PDAs

Say $B \rightarrow xAy$ is a rule in G , and the PDA finds the symbol B on top of its stack, it may pop B and push y , A , and x , in that order

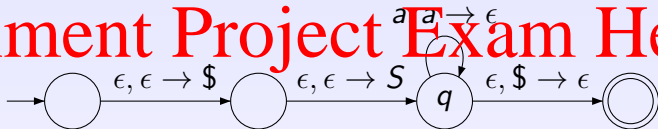


If it finds the terminal x on top of the stack, and x is the next input symbol, it may consume the input and pop x .

From Context-Free Grammars to PDAs

Construct the PDA like this:

Assignment Project Exam Help

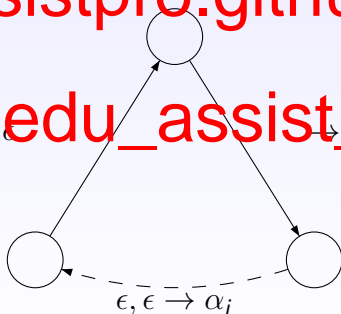


with a self-loop
(S is the g

<https://eduassistpro.github.io>

Add WeChat: edu_assist_pro

For each rule $A \rightarrow \alpha_1 \dots \alpha_n$,
add this loop from q to q :



Example Recogniser

For the grammar

$$S \rightarrow a S b b \mid b \mid \epsilon$$

$$a, a \rightarrow \epsilon$$

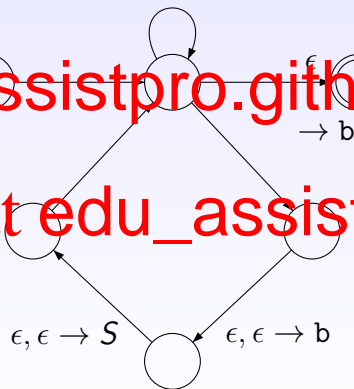
$$b, b \rightarrow \epsilon$$

$$\epsilon, S \rightarrow b$$

$$\epsilon, S \rightarrow \epsilon$$

<https://eduassistpro.github.io>

Add WeChat edu_assist_pr



Pumping Lemma for CFLs

There are languages that are not context-free, and again there is a pumping lemma that can be used to show (some) languages non-context-free:

If A is context-free, then there exists a pumping length p such that for any string $s \in A$ with $|s| \geq p$, there exists a decomposition $s = uvxyz$ such that:

1 $uv^i xy^i z \in A$ for all $i \geq 0$

2 $|vy| \geq 1$

3 $|vxy| \leq p$

We won't prove this lemma, but we give two examples of its use.

Pumping Example 1

$A = \{ww \mid w \in \{0,1\}^*\}$ is not context-free.

Assume it is, let p be the pumping length, take $0^p 1^p 0^p 1^p$.

By the pumping lemma, $0^p 1^p 0^p 1^p = uvxyz$, with $uv^i xy^i z$ in A for all $i \geq 0$, and

There are

$00\dots 0011\dots 1100\dots$

If it straddles the midpoint, it has form $0^i 1^j 0^i 1^j$, with $i < p$, or j

If it is in the first half, $uv^2 xy^2 z$ will have pushed a 1 into the first position of the second half.

Similarly if vxy is in the second half.

Pumping Example 2

$B = \{a^n b^n c^n \mid n \in \mathbb{N}\}$ is not context-free.

Assignment Project Exam Help

By the pu

B for all i .

Either v

<https://eduassistpro.github.io>

If one of them contains two different symbols from B , then uv^2xy^2z has symbols in the wrong order, and so $uv^2xy^2z \notin B$.

Add WeChat edu_assist_pro

So both v and y must contain only one kind of symbol. uv^2xy^2z can't have the same number of a s, b s, and c s.

In all cases we have a contradiction.

Assignment Project Exam Help

The class of context-free languages is closed under

- union
- concatenation
- Kleene star,
- reversal

<https://eduassistpro.github.io>

Add WeChat edu_assist_pr

Closure Properties for CFLs

The class of context-free languages is not closed under intersection!

Hence it is not closed under complement either (why?)

Assignment Project Exam Help

Consider

<https://eduassistpro.github.io>

Exercise: Prove that they are context-free!

But $C \cap D$ is the language $B = \{a^n b^n c\}$
showed is **not** context-free.

Add WeChat edu_assist_pro

However, we do have: If A is context-free and R is regular then
 $A \cap R$ is context-free.

Deterministic PDAs

Is a **deterministic** PDA (a **DPDA**) as powerful as a PDA?

Assignment Project Exam Help
No. A DPDA can recognise the context-free

but not th

<https://eduassistpro.github.io>

Intuitively a deterministic machine cannot know the end of the input has been reached. Suppose it gets input

000011000000000110000

A deterministic machine won't know when to start popping the stack.