1. Draw a parse tree for the following derivation:

$$S \Rightarrow C\ A\ C \Rightarrow C\ A\ b\ b \Rightarrow b\ b\ A\ b\ b \Rightarrow$$

$$b\ b\ B\ b\ b \Rightarrow b\ b\ a\ A\ a\ a\ b\ b$$

$$\Rightarrow b\ b\ a\ b\ a\ a\ b\ b$$

2. Show on your parse tree u, v, x, y, z as per the pumping theorem.

3. Prove that the language for this question is an infinite language.

Wednesday June 21:
Midterm exam in class.

Recall that you need to have at least 50%
For your assignment average.

Your lowest assignment mark will be
dropped in computing your average.

$L = \{ a^n b^p : \quad n \le p \le 3n, \quad n,p \ge 0 \}$

Start symbol S.

$S \rightarrow a\,S\,b$ $\qquad\qquad$ $S \rightarrow \varepsilon$

$S \rightarrow a\,S\,bb$

$S \rightarrow a\,S\,bbb$

This works because any integer p can be expressed as:

$p = r + 2\,(n - r)$ when $n \le p \le 2n$, and

$p = 2r + 3\,(n - r)$ when $2n \le p \le 3n$.

Prove the following languages are context-free by designing context-free grammars which generate them:

$L_1 = \{a^n b^n c^p \ : \ n, p \geq 0\}$

$L_2 = \{a^n b^p c^n \ : \ n, p \geq 0\}$

$L_3 = \{a^n b^m : n \neq m, \ n, m \geq 0\}$

Hint: $L_3 = \{a^n b^m : n < m\} \cup \{a^n b^m : n > m\}$

$L_4 = \{ c\,u\,c\,v\,c : |u| = |v|, \ u, v \in \{a, b\}^* \}$

What is $L_1 \cap L_2$?

# Pushdown Automata

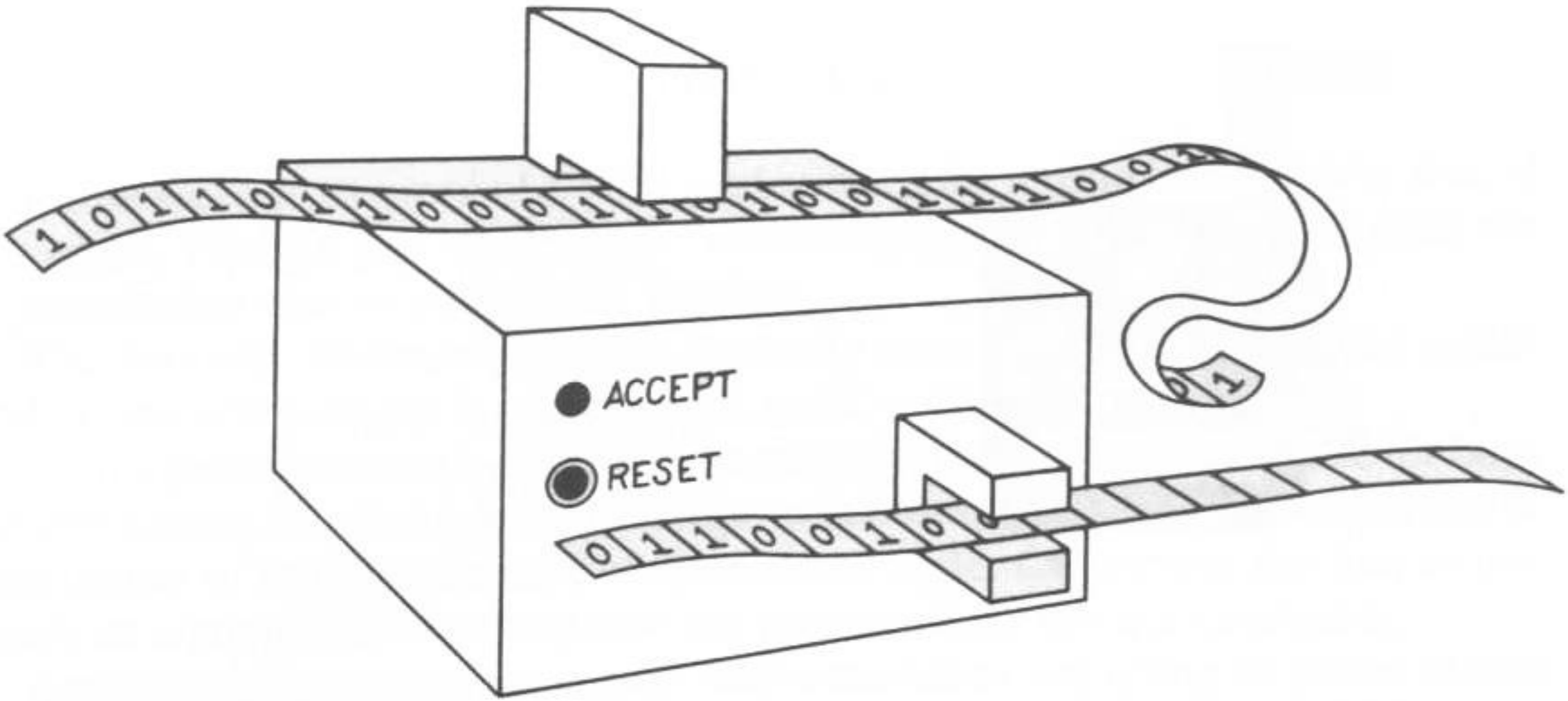



Figure 7.4 A push-down automaton

**Picture from: Torsten Schaßan**

# Pushdown Automata:

A pushdown automaton is like a NDFA which has a stack.

Every context-free language has a pushdown automaton that accepts it.

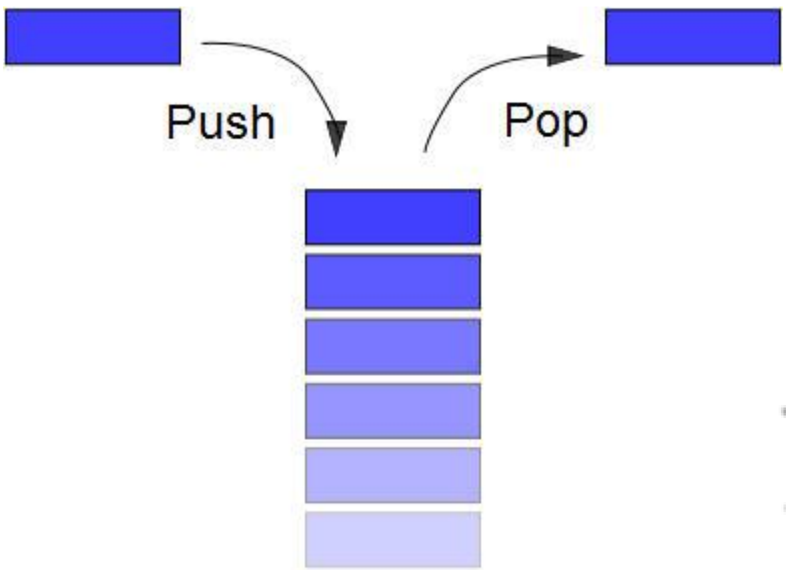
This lecture starts with some examples, gives the formal definition, then investigates PDA's further.

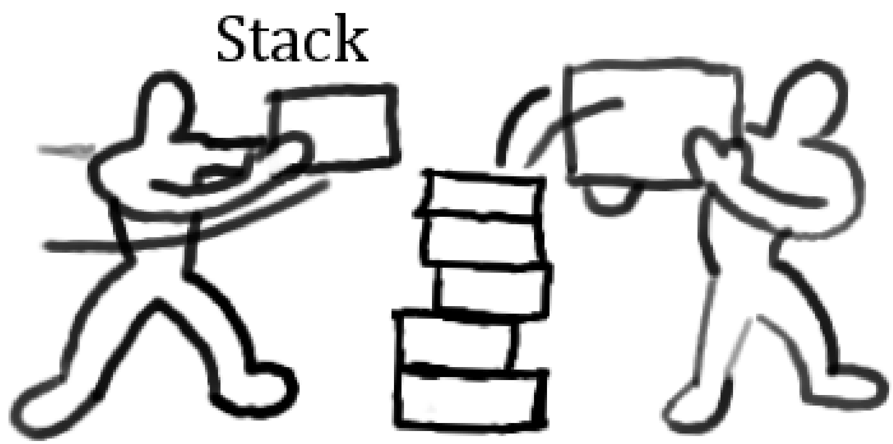
# Stacks


corbis


SO LONG, AND THANKS FOR ALL THE FISH
DOUGLAS ADAMS
Thoreau
WALDEN
BARNES & NOBLE
The Gormenghast Novels
TITUS GROAN
GORMENGHAST
TITUS ALONE
Mervyn Peake
TEXTPATTERN SOLUTIONS
Bulletproof Web Design
THE PROUD HIGHWAY
HUNTER S. THOMPSON

# Stack Data Structure: permits push and pop at the top of the stack.



$L = \{ w\, c\, w^R : w \in \{a, b\}^* \}$

Start state: s, Final State: {t}

| State | Input | Pop | Next state | Push |
|---|---|---|---|---|
| s | a | ε | s | A |
| s | b | ε | s | B |
| s | c | ε | t | ε |
| t | a | A | t | ε |
| t | b | B | t | ε |

To accept, there must exist a computation which:

1. Starts in the start state with an empty stack.
2. Applies transitions of the PDA which are applicable at each step.
3. Consumes all the input.
4. Terminates in a final state with an empty stack.

$w = a\ b\ b\ c\ b\ b\ a$

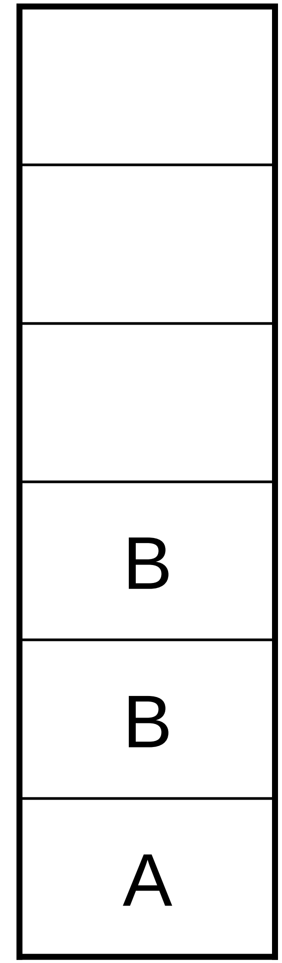
$(s, abbcbba, \varepsilon) \vdash^*$

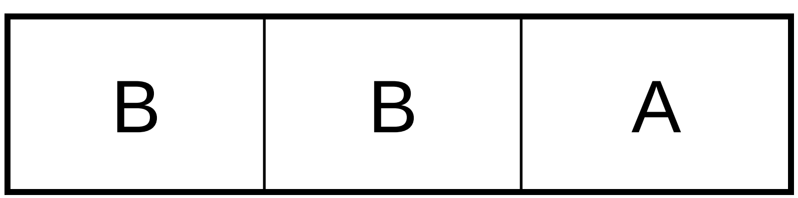
$(s, cbba, BBA) \vdash$

$(t, bba, BBA) \vdash^*$

$(t, \varepsilon, \varepsilon)$

Stack is knocked over like this:



A pushdown automaton is a sextuple

$M = (K, \Sigma, \Gamma, \Delta, s, F)$ where

K is a finite set of states,

$\Sigma$ is an alphabet (the input symbols)

$\Gamma$ is an alphabet (the stack symbols)

$\Delta$, the transition relation,

is a finite subset of

$$\underbrace{(\,K}_{\text{state}} \times \underbrace{(\Sigma \cup \{\varepsilon\})}_{\text{input}} \times \underbrace{\Gamma^*}_{\text{pop}}) \times \underbrace{(K}_{\text{next state}} \times \underbrace{\Gamma^*)}_{\text{push}}$$

A configuration of a PDA is a member of

$$K \times \Sigma^* \times \Gamma^*$$

current state    input remaining    stack

A configuration $(q, \sigma w, \alpha x) \vdash (r, w, \beta x)$ if $((q, \sigma, \alpha), (r, \beta)) \in \Delta$.

For $M = (K, \Sigma, \Gamma, \Delta, s, F)$,

$L(M)$ (the language accepted by $M$) =

$\{ w \in \Sigma^* : (s, w, \varepsilon) \vdash^* (f, \varepsilon, \varepsilon)$ for some final state $f$ in $F\}$.

$L(M) = \{ w \in \Sigma^* : (s, w, \varepsilon) \vdash^* (f, \varepsilon, \varepsilon)$ for some final state $f$ in $F\}$.

To accept, there must exist a computation which:

1. Starts in the start state with an empty stack.
2. Applies transitions of the PDA which are applicable at each step.
3. Consumes all the input.
4. Terminates in a final state with an empty stack.

# PDA's are non-deterministic:

$L = \{ w w^R : w \in \{a, b\}^* \}$

Start state: s, Final State: {t}

| State | Input | Pop | Next state | Push |
|---|---|---|---|---|
| s | a | ε | s | A |
| s | b | ε | s | B |
| s | ε | ε | t | ε |
| t | a | A | t | ε |
| t | b | B | t | ε |

Guessing wrong time to switch from s to t gives non-accepting computations.

Some non-accepting computations on aaaa:

1. Transfer to state t too early:

$(s, aaaa, \varepsilon) \vdash (s, aaa, A) \vdash (t, aaa, A)$
$\vdash (t, aa, \varepsilon)$

Cannot finish reading input because stack is empty.

2. Transfer to state t too late:

$(s, aaaa, \varepsilon) \vdash (s, aaa, A) \vdash (s, aa, AA)$
$\vdash (s, a, AAA) \vdash (t, a, AAA) \vdash (t, \varepsilon, AA)$

Cannot empty stack.

## Accepting computation on $aaaa$:

$$(s, aaaa, \varepsilon) \vdash (s, aaa, A) \vdash (s, aa, AA)$$

$$\vdash (t, aa, AA) \vdash (t, a, A) \vdash (t, \varepsilon, \varepsilon)$$

The computation started in the start state with the input string and an empty stack.

It terminated in a final state with all the input consumed and an empty stack.

L= {w in {a, b}* : w has the same number of a's and b's}

Start state: s

Final states: {s}

| State | Input | Pop | Next state | Push |
| --- | --- | --- | --- | --- |
| s | a | ε | s | B |
| s | a | A | s | ε |
| s | b | ε | s | A |
| s | b | B | s | ε |

L= {w in {a, b}* : w has the same number of a's and b's}

State state: s, Final states: {f}

| State | Input | Pop | Next state | Push |
|---|---|---|---|---|
| s | ε | ε | t | X |
| t | a | X | t | BX |
| t | a | A | t | ε |
| t | a | B | t | BB |
| t | b | X | t | AX |
| t | b | A | t | AA |
| t | b | B | t | ε |
| t | ε | X | f | ε |

A more deterministic solution:

Stack will never contain both A's and B's.

X- bottom of stack marker.

Design contex-free grammars that generate:

$L_1 = \{ u\,v : u \in \{a,b\}^* ,\ v \in \{a, c\}^*,$

$\text{and } |u| \leq |v| \leq 3\,|u| \}.$

$L_2 = \{ a^p\, b^q\, c^p\, a^r\, b^{2r} : p, q, r \geq 0 \}$