COSC1107 Computing Theory

(We will commence soon. We are just allowing a few minutes for people to join and set up. *Please mute your microphone unless you are speaking.* You can raise your hand or use the chat at any time.)

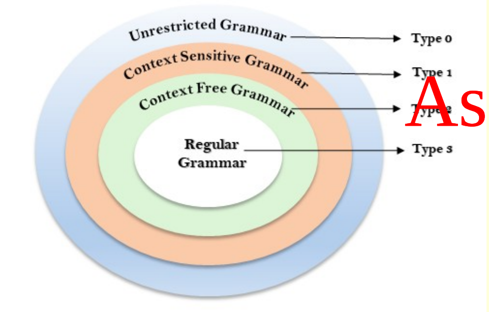
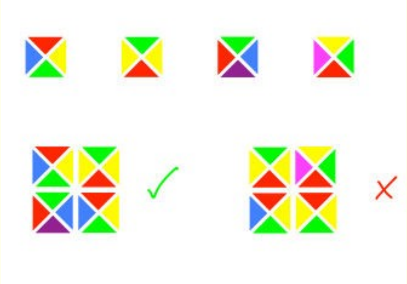
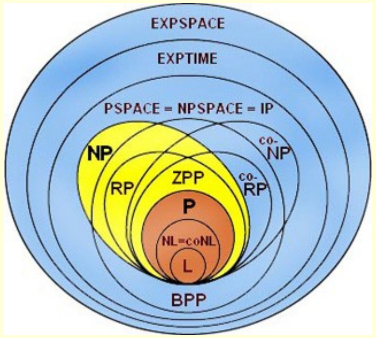
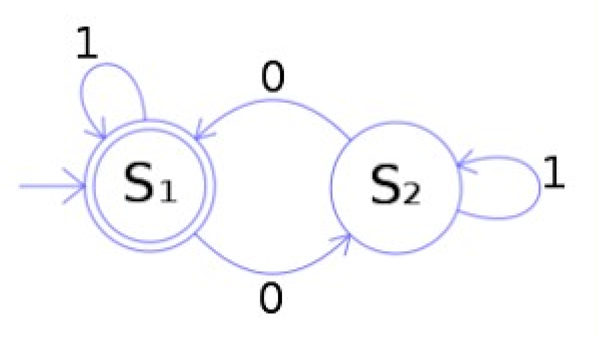
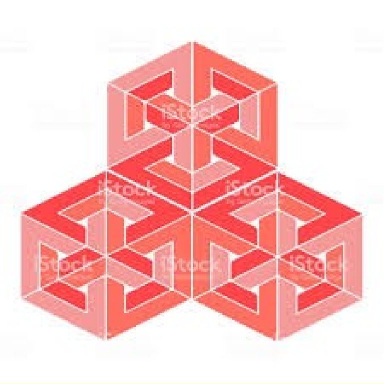


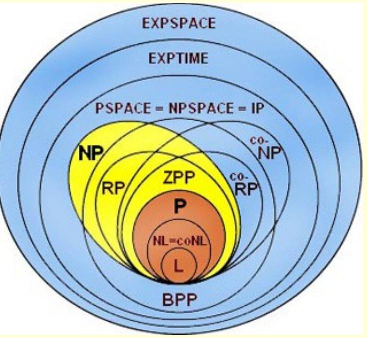
Fig: Chomsky Hierarchy

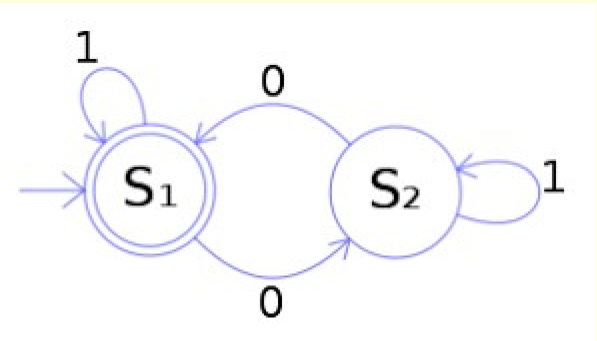










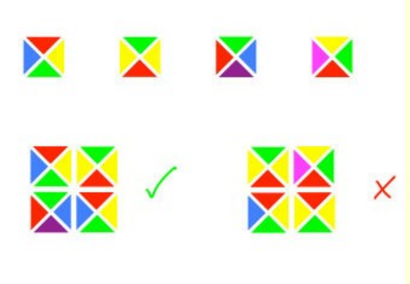


COSC1107

Computing Theory

Chomsky Hierarchy

Week 7



James Harland

james.harland@rmit.edu.au

*** With thanks to Sebastian Sardina**

Intro music 'Far Over' playing now …

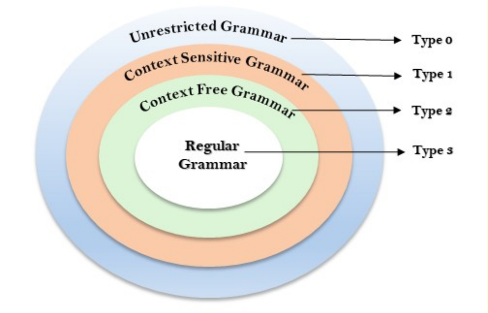


Fig: Chomsky Hierarchy

Acknowledgement

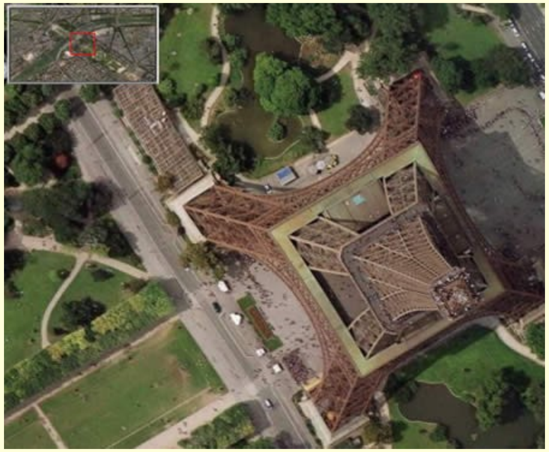
Acknowledgement of Traditional Owners
Original artwork by Amanda Power

RMIT University acknowledges the people of the Woi wurrung and Boon wurrung language groups of the eastern Kulin Nations on whose unceded lands we conduct the business of the University. RMIT University respectfully acknowledges their Ancestors and Elders, past and present.

RMIT also acknowledges the Traditional Custodians and their Ancestors of the lands and waters across Australia where we conduct our business.

(add your name here to volunteer for this or email me)

Overview



- Questions?
- Grammar normal forms
- Questions?
- Pumping Lemma
- Questions?
- Platypus Game
- Questions?

What **can** be done

What **can't** be done

Of course!

Weekly Schedule

	Lecture/Lectorial	Tutorial	Assessment
1	Formal languages, grammars	Motivations & Mathematical preliminaries	
2	Finite State Machines	Grammars	Quiz 1
3	Pushdown Automata, nondeterminism	NFAs and DFAs	Quiz 2
4	Turing machines	Pushdown automata	Quiz 3
5	Computability, universality	Turing machines	Quiz 4
6	Pumping Lemma, NFA->DFA conversion	Computability, universality	Assignment 1, Quiz 5
7	Chomsky Hierarchy	Nondeterminism, Pumping Lemma	Quiz 6
8	Unrestricted grammars		Quiz 7
9	Complexity and intractability	Unrestricted grammars	Quiz 8
10	NP-completeness	Complexity and intractability	Quiz 9
11	Zero-knowledge proofs	NP-completeness	Quiz 10
12	Research and requests	Sample exercise	Assignment 2
14-16	--	--	Final exercise

Foundations

Relationships

Analysis

Assessment

Questions?

Questions?

Questions?

Chomsky Hierarchy

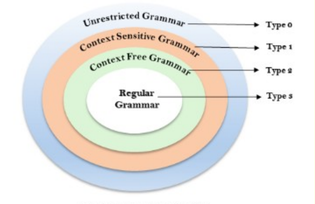
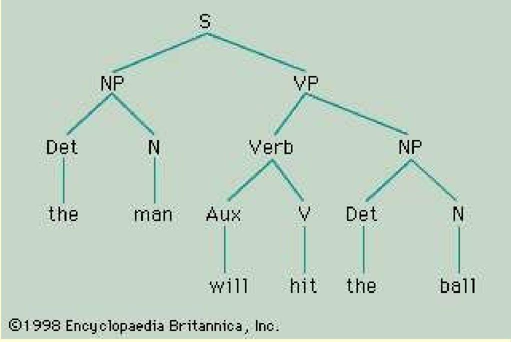
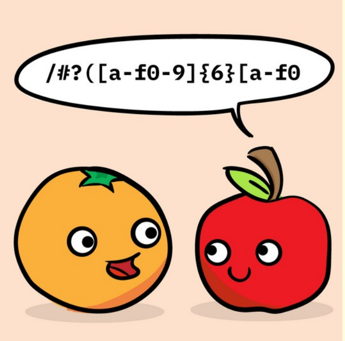


Fig: Chomsky Hierarchy

Automata		Languages		Grammars
		Week 8		
		Undecidable languages		---
Turing Machines	↔	Recursively enumerable languages	↔	Unrestricted grammars
Linear Bounded Automata	↔	Context-sensitive languages	↔	Context-sensitive grammars
(Nondeterministic) Pushdown Automata	↔	Context-free languages Week 7	↔	Context-free grammars
Deterministic Pushdown Automata		?? (Deterministic CF?) Week 7		???
Nondeterministic Finite Automata & Deterministic Finite Automata	↔	Regular languages Week 6	↔	Regular grammars & regular expressions





Grammars

What are grammars again?

“Around the survivors a perimeter create!”

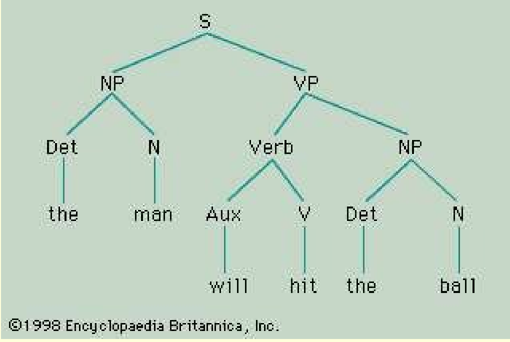
Much studied by Noam Chomsky in the 1950’s

- Similar to rules used in natural language

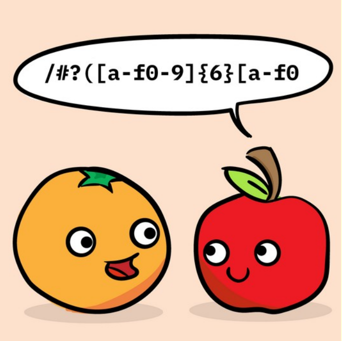
- Special start symbol S

- Set of rules of the form X Y (“whenever you see X, you can replace it with Y”)
- For any string w_1Xw_2, can obtain string w_1Yw_2
- Stop when no more rules apply

Grammars



©1998 Encyclopaedia Britannica, Inc.



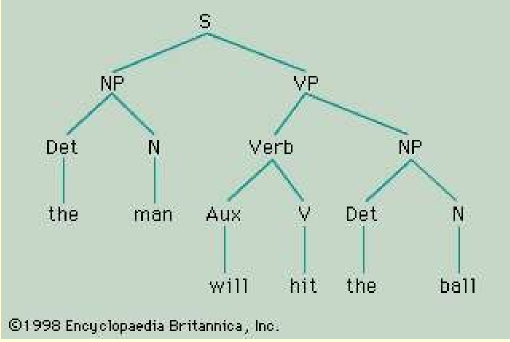
Inherently nondeterministic!

S zN@E.mad
N 1 | 1E
E DD | DDE
D 0 | 1 | 2

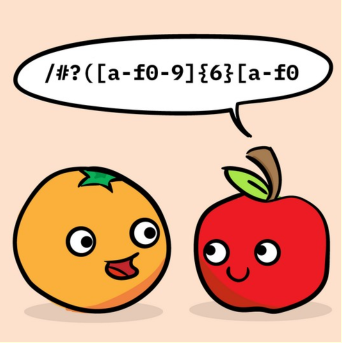
S zN@E.mad z1E@E.mad z1DD@E.mad z1DD@DD.mad

… z10D@DD.mad z101@DD.mad z101@0D.mad z101@01.mad
… z11D@DD.mad z110@DD.mad z110@1D.mad z110@10.mad
… z10D@DD.mad z100@DD.mad z100@0D.mad z100@00.mad
… z12D@DD.mad z122@DD.mad z122@2D.mad z122@22.mad
…

Grammars



©1998 Encyclopaedia Britannica, Inc.



Question: What **exactly** is a grammar?

Answer: A set of replacement rules for strings

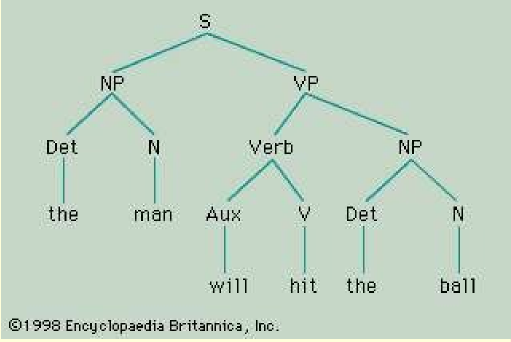
V: set of **non-terminal** symbols (or variables) (eg S, A, B, C, …)
T: set of terminal symbols (eg a, b, c, 0, 1, 2, …)

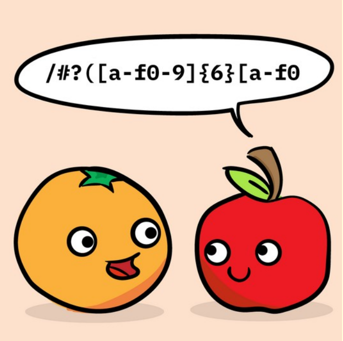
A rule is of the form $\underbrace{(V\ \ T)}_{\text{non-empty string over V T}}\ \ \underbrace{(V\ \ T)^*}_{\text{string over V T}}$

Derivation: Given rule $L\ \ R_1 \mid R_2 \mid \dots \mid R_n$ and string xLy, a permitted step is $xLy\ \ x R_i y$ for any $i = 1 \dots n$

Take transitive closure of steps from S …

Grammars





A rule is of the form $\underbrace{(V\ \ T)^+}_{\text{non-empty string over V T}}\ \underbrace{(V\ \ T)^*}_{\text{string over V T}}$

$L\ \ R_1 \mid R_2 \mid \ldots \mid R_n$

Different types of grammars result from different constraints on L and R_i

Name	Type	Constraint	Example
Unrestricted	0	--	AbC AC
Context-sensitive	1	|L| ≤ |Ri| or S	AbC abc
Context-free	2	|L| = 1	A AC
Regular	3	|L| = 1 (ie L V) and R_i T {} TV	A a A bB A

Chomsky Hierarchy

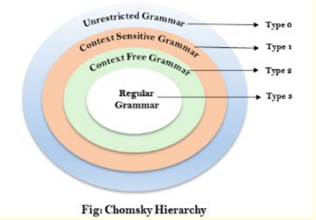


Fig: Chomsky Hierarchy

Automata	Languages	Grammars
---	*Undecidable languages*	---
Turing Machines	Recursively enumerable languages	Unrestricted grammars AbC AC
Linear Bounded Automata	Context-sensitive languages	Context-sensitive grammars AbC abc
(Nondeterministic) Pushdown Automata	Context-free languages	Context-free grammars A AC
Nondeterministic Finite Automata & Deterministic Finite Automata	Regular languages	Regular grammars & regular expressions A a | bB |

Chomsky Hierarchy

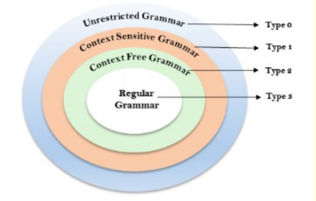


Fig: Chomsky Hierarchy

- Grammars generate languages
- Automata accept languages

Relationships

- Given a grammar G, is there an automaton M such that L(G) = L(M)? (so M accepts w iff G generates w)

- Given an automaton M, is there a grammar G, such that L(M) = L(G)? (so G generates w iff M accepts w)

Types

What of grammar is G?

Regular? Context-free? Context-sensitive? Unrestricted?

What of automaton is M?

DFA? NFA? PDA? LBA? DTM? NDTM?

Chomsky Hierarchy

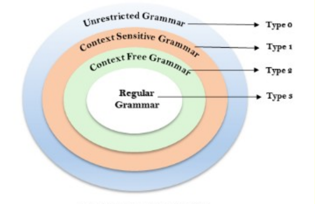


Fig: Chomsky Hierarchy

Automata	Languages	Grammars
	Undecidable languages	---
Turing Machines	Recursively enumerable languages	Unrestricted grammars
Linear Bounded Automata	Context-sensitive languages	Context-sensitive grammars
(Nondeterministic) Pushdown Automata	Context-free languages	Context-free grammars
Deterministic Pushdown Automata	?? (Deterministic CF?)	???
Nondeterministic Finite Automata & Deterministic Finite Automata	Regular languages	Regular grammars & regular expressions

Questions?

Questions?

Questions?

Normal Forms

shutterstock.com • 132311147

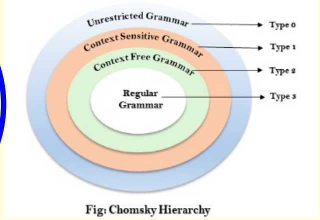


Fig: Chomsky Hierarchy

Grammars are rather free form …
The rule below is legal!

S   SUPERCALIFRAGILISTICEXPIALIDOCIOUS

For any context-free grammar G,
there are grammars G_1 and G_2 such that $L(G) = L(G_1) = L(G_2)$ and
G_1 is in Chomksy* normal form and
G_2 is in Greibach* normal form

*Noam Chomsky

*Sheila Greibach

Chomksy normal form
S
A BC
A a

Derived string only grows by 1

Greibach normal form
S
A $aA_1A_2\ldots A_n$ where $n \geq 0$

pop   input   push

Like PDA

Regular
A aB | a |

input   state

Like FSA

Week 7

Computing Theory

Chomsky Normal Form

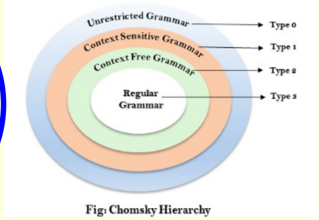


Fig: Chomsky Hierarchy

Grammar is in Chomsky normal form if every rule is of the form

S
A BC
A a

where B,C are variables (non-terminals) other than S

- S is the only rule with the empty string.
- Otherwise the right hand side has length 1 or 2
- **All context-free grammars can be (efficiently) converted into an equivalent one in CNF**
- String in derivation grows by 1 each step or stops
- Some variants exist

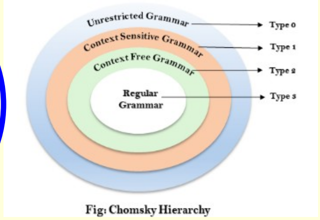


Fig: Chomsky Hierarchy

Chomsky Normal Form

S | ASA | BSB | A | B
A 0
B 1

(from week 1 slides)

S | ASA | BSB | 0 | 1
A 0
B 1

S | AC | BD | 0 | 1
A 0
B 1
C SA
D SB

S aB| bS | cS |
B aB | bC | cS |
C aB | bS |

(from week 1 slides)

S aB| bS | cS |
B aB | b | bC | cS |
C aB | bS

S a | aB | bS | cS |
B a | aB | b | bC | cS
C a | aB | bS

S a | AB | DS | ES |
B a | AB | b | DC | ES
C a | AB | DS
A a
D b
E c

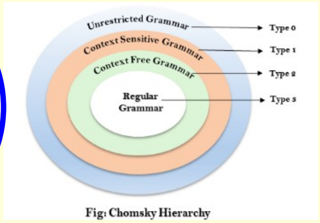


Fig: Chomsky Hierarchy

Chomsky Normal Form

Conversion Process:

1. Make start state non-recursive
2. Eliminate A rules (other than S)
3. Eliminate 'chain' rules (eg A B, B C, C DE becomes A DE)
4. Eliminate variables that do not derive strings (eg A B, B A)
5. Eliminate variables not reachable from S
6. Replace terminals with variables (eg A bCb to A BCB, B b)
7. Reduce variables in each rule (eg A BCB to A BD, D CB)
   (rule with n variables becomes n-1 rules with 2 variables)

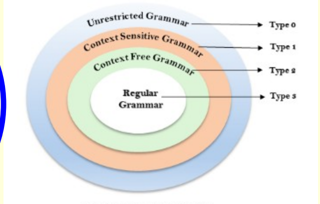


Fig: Chomsky Hierarchy

Chomsky Normal Form

Derivations are much simpler to manage in with CNF

S : delete S from the current string
(only time derived string gets shorter)

A BC: replace A with BC
(adds one variable to the string, which grows by 1 in length)

A a: replace A with a
(one less variable, one more terminal and string stays the same length)

So if we have S T, then |T| grows with every application of a rule like A BC

Precise relationship between depth of parse tree and |T|

Greibach Normal Form

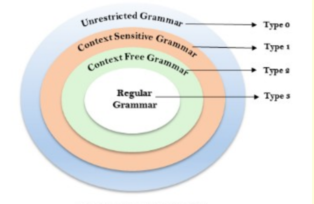


Fig: Chomsky Hierarchy

Grammar is in Greibach normal form if every rule is of the form

S
A a
A $aA_1A_2 \ldots A_n$

- No rules like A → Ba or A → BC

- Generalisation of regular grammars (for which $n \leq 1$)
- **All context-free grammars can be (efficiently) converted into an equivalent one in GNF**
- Conversion is more complex than for Chomsky normal form
- Needs to eliminate 'left' recursions, so that terminals can be made to appear 'predictably from the left'

Questions?

Questions?

Questions?

Quiz time!

Go to Canvas and find the quiz Lectorial 7 Question set

- Not worth any marks
- You can consult other students if you wish
- Time limit will be 10 minutes

Go!

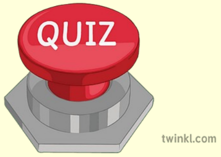
The pictures will take 10 minutes to disappear!

Thomas music means 1 minute left!

Questions?

Questions?

Questions?



How did you go?

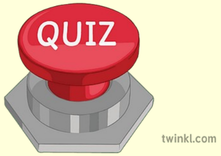
Question 1: Convert the grammar below to Chomsky normal form.

S aaSB |
B bB | b

Is the above grammar in Greibach normal form?

S CD |
B AB | b
A b
C EE
E a
D SB

S aaTB |
T aaTB |
B AB | b
A b
C EE
E a
D SB



How did you go?

Question 2: Convert the grammar below to Chomsky normal form.

S aSbb | A
A cA | c

Is the above grammar in Greibach normal form?

S EB | CA | c
E FS
B DD
D b
A CA | c
C c
F a

Grammars and PDAs

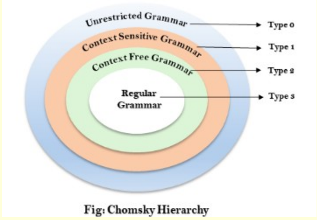


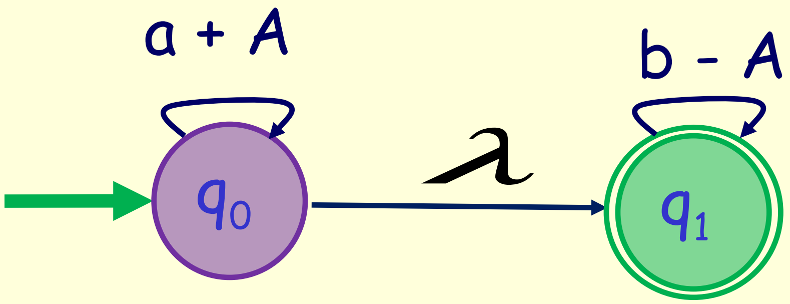
Fig: Chomsky Hierarchy

(Nondeterministic) Pushdown Automata	Context-free languages	Context-free grammars A AC

- For every context-free grammar G, there is an PDA M such that L(G) = L(M)
- For every PDA M, there is a context-free grammar G such that L(M) = L(G)

Consider $L = \{a^n b^n \mid n \geq 0\}$

S aSb |



Grammars and PDAs

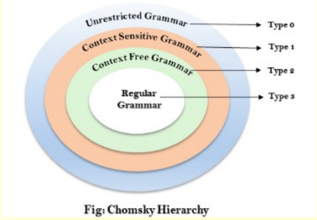
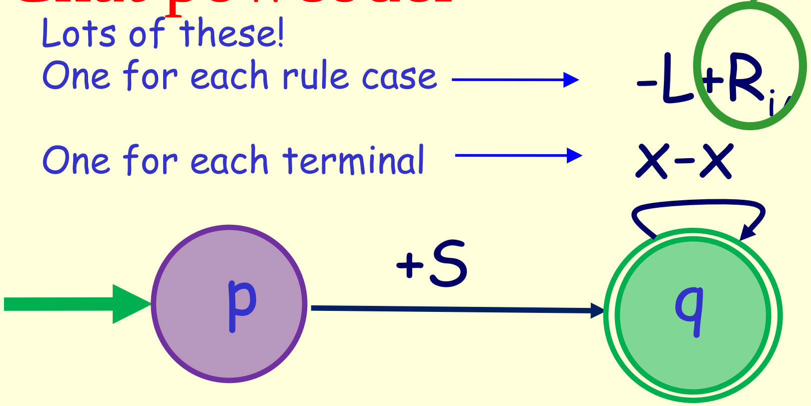


Fig: Chomsky Hierarchy

Given any grammar, construct an equivalent PDA as follows

- PDA has two states (say p and q)
- Initially push S onto the stack
- For each rule L R_i in G, add a transition (q, , L) = (q, R_i)
  (replaces L on the stack with R_i)
- For each terminal x, add a transition (q, x, x) = (q,)
  (remove non-terminal from top of the stack)



(Also shows only ever "need" two states in a PDA)

"Use extra states if need be …"

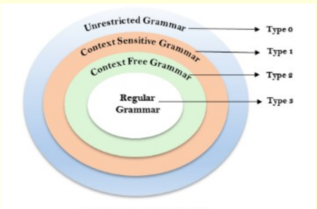
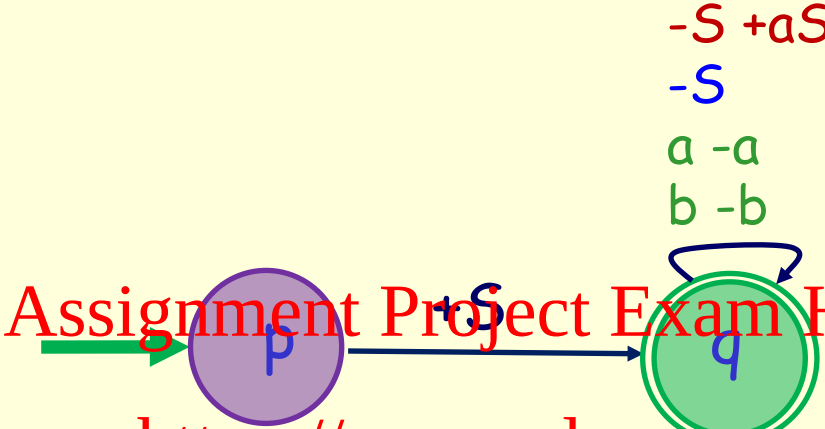


Fig: Chomsky Hierarchy

Grammars and PDAs

S aSb |



-S +aSb
-S
a -a
b -b

S aSb aaSbb aaaSbbb aaabbb

(p, aaabbb,) (q, aaabbb, S) (q, aaabbb, aSb) (q, aabbb, Sb)

(q, aabbb, aSbb) (q, abbb, Sbb) (q, abbb, aSbbb)

(q, bbb, Sbbb) (q, bbb, bbb) (q, bb, bb) (q, b, b) (q, ,)

(Greibach normal form simplifies this process, but is not required)

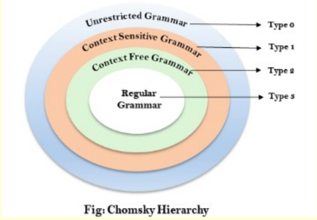


Fig: Chomsky Hierarchy

Grammars and PDAs

Given any PDA, construct an equivalent grammar as follows
(this is a little trickier than the other way around!)

each step in a derivation deletes a symbol …

- Transform the PDA so that every transition pops the stack (!!)
  - (p, x,) = (q,) (p, x, A) = (q, A) for each A
  - (p, x,) = (q, B) (p, x, A) = (q, AB) for each A

- Construct grammar rules which mimic the PDA execution
- Variable $(q_i\ A\ q_j)$ represents execution starting in q_i, ending in q_j and popping A

- S (s Z f) where s, f are start and final states of the PDA
- For each $(q_1,x,B) = (q_2,A)$, add rule $(q_1\ B\ p)\ x\ (q_2\ A\ p)$ for all states p
- For each $(q_1,x,A) = (q_2,AB)$, add rule $(q_1\ A\ p)\ x\ (q_2\ A\ r)\ (r\ B\ p)$ for all states p,r
- (q q) for all states q

same

same

(q_1Aq_2) * w iff (q_1,w,A) $(q_2,)$

CFGs and PDAs

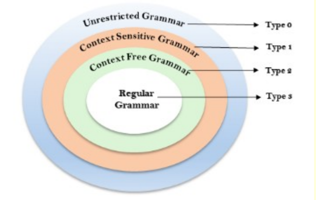


Fig: Chomsky Hierarchy

What **can** PDAs do?

- For every PDA there is an equivalent CFG
- Recognise regular languages
- Recognise context-free languages
- DPDAs are weaker than PDAs (!!)

What **can't** PDAs do?

- Recognise context-sensitive languages
- Recognise recursive languages

Questions?

Questions?

Questions?

Chomsky Hierarchy

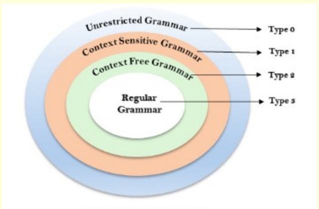


Fig: Chomsky Hierarchy

Automata	Languages	Grammars
	Undecidable languages	---
Turing Machines	Recursively enumerable languages	Unrestricted grammars
Linear Bounded Automata	Context-sensitive languages	Context-sensitive grammars
(Nondeterministic) Pushdown Automata	Context-free languages	Context-free grammars
Deterministic Pushdown Automata	?? (Deterministic CF?)	???
Nondeterministic Finite Automata & Deterministic Finite Automata	Regular languages	Regular grammars & regular expressions

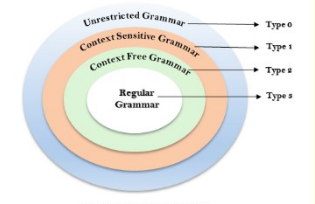


Fig: Chomsky Hierarchy

Limitation of CFGs

Type	Memory Size	Memory access
DFAs	Bounded*	Defined by machine
PDAs	Unbounded	Top of stack only
(Linear Bounded Automata)	*?? (see Week 8)*	*?? (see Week 8)*
Turing Machines	Infinite	Random (ie unlimited)

* bounded means the number is fixed and known in advance

What happens when a CFG in Chomsky normal form processes a string longer than the number of variables?

"Now you are getting somewhere …"

Some variable must be repeated!

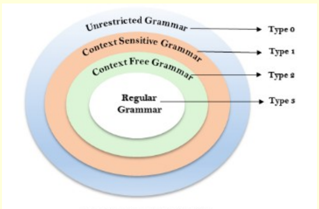


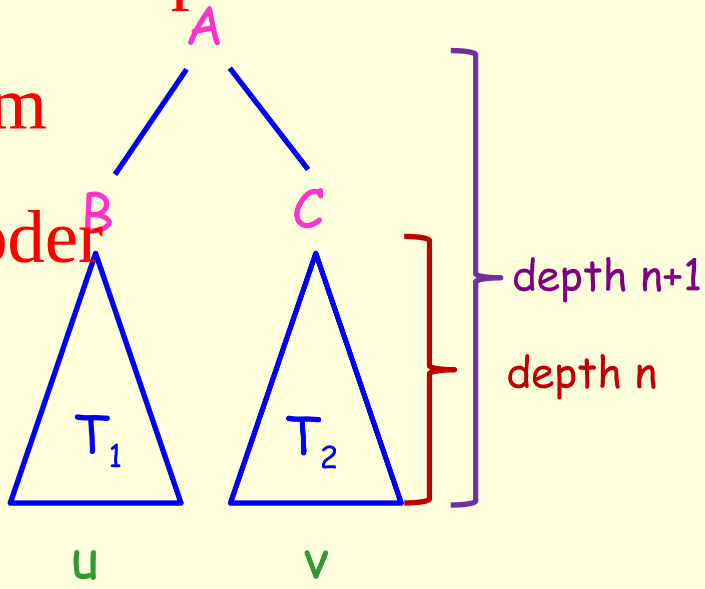
Fig: Chomsky Hierarchy

Limitation of CFGs

Let G be a context-free grammar in Chomsky normal form, and A * w with (binary) derivation tree T.
If depth(T) = n, length(w) ≤ 2^{n-1}

Chomksy normal form
S
A BC
A a

Proof: By induction on the depth of T.
Base case: depth(T) = 1, so the derivation is either S or A a
Inductive case: Assume it holds for all derivations of depth n. Let A * w be a derivation of depth n+1. As G is in **Chomsky normal form**, we must have
A BC * uv where B * u and C * v
By the hypothesis length(u) ≤ 2^{n-1} and length(v) ≤ 2^{n-1} so length(uv) ≤ 2^{n-1} + 2^{n-1} = 2^{n}



Limitation of CFGs

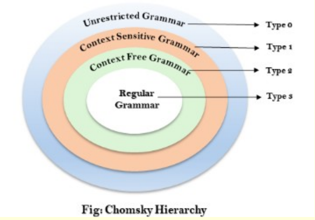


Fig: Chomsky Hierarchy

Let G be a context-free grammar in Chomsky normal form, and S * w a derivation of w. If length(w) ≥ 2^{n}, then the derivation tree has depth at least n+1.

Pumping Lemma for Context-free Languages

For any context-free language L, there is n ≥ 1 such that for any w L with |w| ≥ n, there exist x, y, z, u, v such that w = xyzuv and

1. |yzu| ≤ n
2. y or u
3. x y^i z v^i u L for all i ≥ 0

Limitation of CFGs

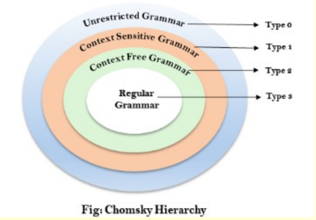
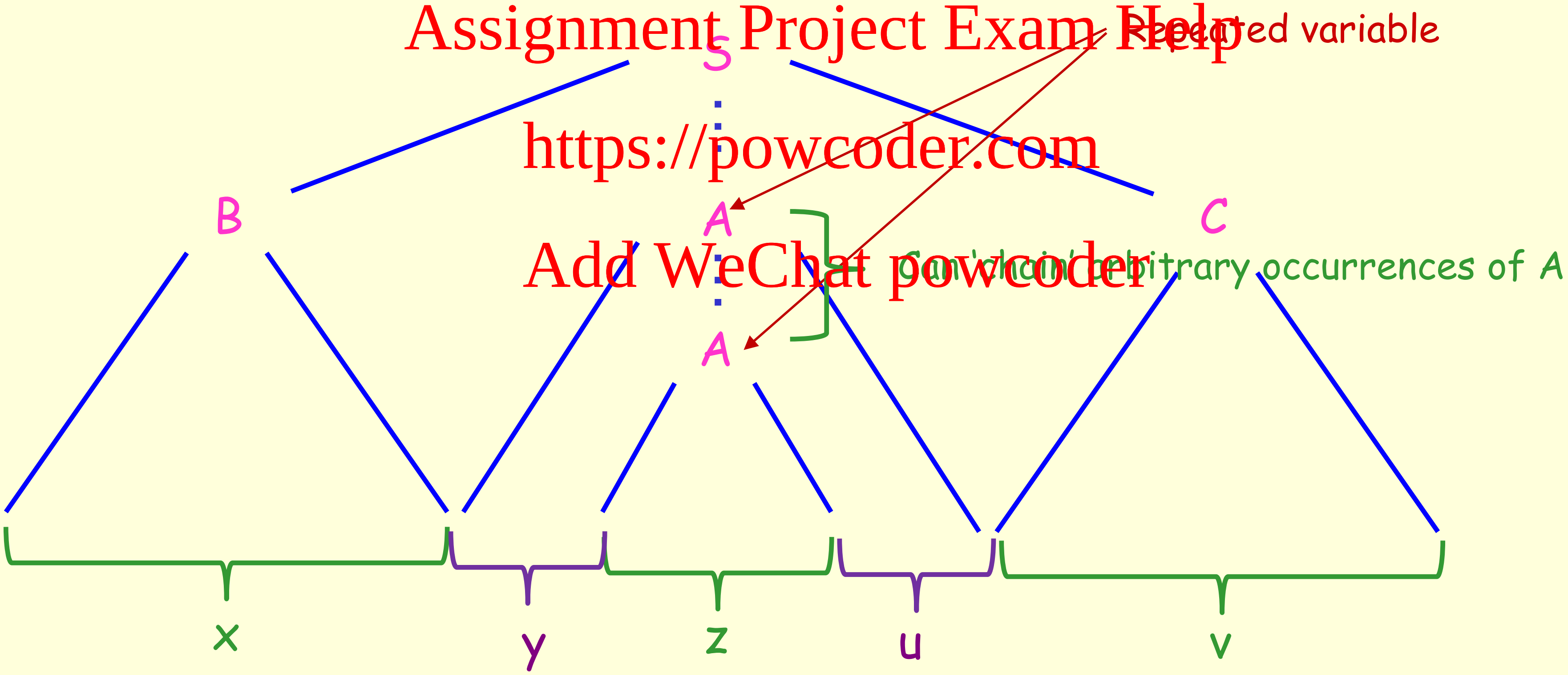


Fig: Chomsky Hierarchy

Let S * w where |w| = n = $2^{|V|}$ where V is the variables in G.
So the derivation tree has depth at least n+1 = |V| + 1.
So a path from S contains at least 2 occurrences of some variable …



Repeated variable

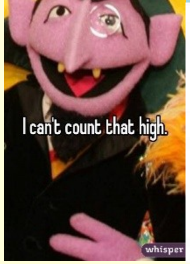
Can 'chain' arbitrary occurrences of A

Questions?

Questions?

Questions?

Pumping Lemma



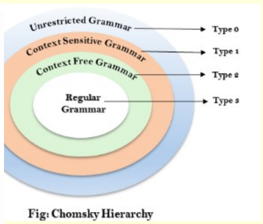


Fig: Chomsky Hierarchy

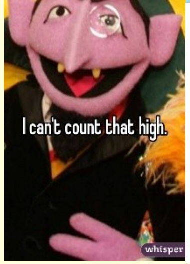
Usual use is to show languages not *context-free* by contradiction

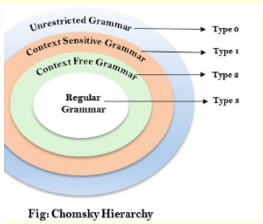
1. Assume L is *context-free*
2. Apply Pumping Lemma
3. Choose string w ∈ L
4. Use $|yzu| \leq n$ to get information about y and u
5. Choose i such that $x y^i z u^i v \notin L$ (i = 2 usually works)
6. Contradiction!

Conclude that L is not *context-free*

All such proofs the same except steps 3 & 5

Pumping Lemma





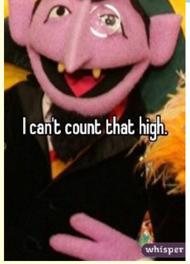
The language $L = \{a^ib^ic^i \mid i \geq 0\}$ is not context-free

Proof: Assume L is context-free. Then the Pumping Lemma applies and so there is an n ≥ 1 such that for all w L such that |w| ≥ n, w = xyzuv where

1. |yzu| ≤ n
2. y or u
3. $x\, y^i\, z\, u^i\, v$ L for all i ≥ 0

Choose $w = a^nb^nc^n$ and so w L and |w| ≥ n. So by the Pumping Lemma
$w = xyzuv = a^nb^nc^n$ and |yzu| ≤ n.
If y or u contains ab or bc, then xyyzuuv L
So both y and u contain only one of a or b or c.
But then also xyyzuuv L as this will contain unequal numbers of a's, b's and c's.
Hence L is not context-free.

Pumping Lemma



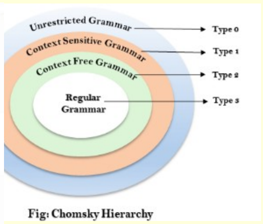


Fig: Chomsky Hierarchy

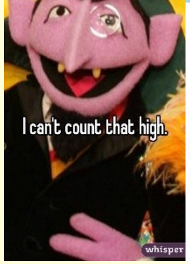
The language $L = \{a^i b^j a^i b^j \mid i,j \geq 0\}$ is not context-free

Proof: Assume L is context-free. Then the Pumping Lemma applies and so there is an $n \geq 1$ such that for all w L such that $|w| \geq n$, $w = xyz$
where

1. $|yzu| \leq n$
2. y or u
3. $x y^i z u^i v$ L for all $i \geq 0$

Choose $w = a^n b^n a^n b^n$ and so w L and $|w| \geq n$. So by the Pumping Len
$w = xyzuv = a^n b^n a^n b^n$ and $|yzu| \leq n$.
If y or u contains ab or ba, then $xyyzuuv$ L
So both y and u contain only one of a or b.
But then also $xyyzuuv$ L as this will contain unequal numbers of a's or b's.
Hence L is not context-free.

Pumping Lemma



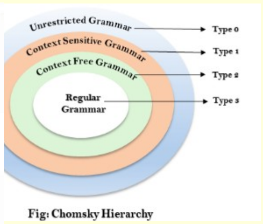


Fig: Chomsky Hierarchy

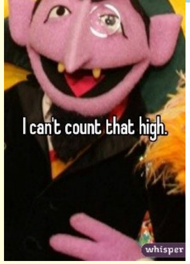
The language L = { xx | x {a,b}* } is not context-free

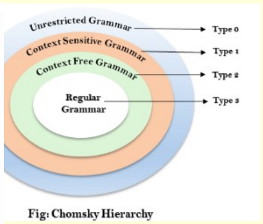
Proof: Assume L is context-free. Then the Pumping Lemma applies and so there is an $n \geq 1$ such that for all w L such that $|w| \geq n$, w = xyzuv where

1. $|yzu| \leq n$
2. y or u
3. $x y^i z u^i v$ L for all $i \geq 0$

Choose $w = a^n b^n a^n b^n$ and so w L and $|w| \geq n$. So by the Pumping Lem
$w = xyzuv = a^n b^n a^n b^n$ and $|yzu| \leq n$.
If y or u contains ab or ba, then xyyzuuv L
So both y and u contain only one of a or b.
But then also xyyzuuv L as this will contain unequal numbers of a's or b's.
Hence L is not context-free.

CFGs





Languages known to be context-free include

$L = \{a^ib^i \mid i \geq 0\}$
$L = \{ ww^R \mid w \;\; \{a,b\}^* \}$
$L = \{ a^i b^j c^k \mid i \; j \text{ or } j \; k \}$
$L = \{a^ib^ja^jb^i \mid i,j \geq 0\}$
$L = \{ w \mid w \;\; \{a,b\}^*, n_a(w) = n_b(w)\}$

Languages known not to be context-free include

$L = \{a^ib^ic^i \mid i \geq 0\}$
$L = \{a^ib^ja^ib^j \mid i,j \geq 0\}$
$L = \{ xx \mid x \;\; \{a,b\}^* \}$
$L = \{ a^m \mid m \text{ is prime}\}$
$L = \{ a^m \mid m = n^2 \text{ for some } n \geq 0\}$
$L = \{ w \mid w \;\; \{a,b,c\}^*, n_a(w) = n_b(w) = n_c(w)\}$

Questions?

Questions?

Questions?

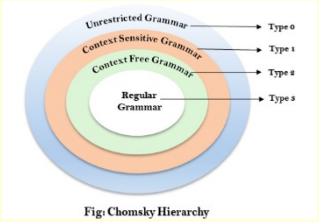


Fig: Chomsky Hierarchy

Properties of Languages

If L_1 and L_2 are regular languages, then so are

1. L_1 L_2
2. L_1L_2
3. L_1* (and L_2*)
4. L
5. L_1 L_2
6. L_1 L_2

1,2,3: Consider R_1, R_2 such that $L(R_1) = L_1$, $L(R_2) = L_1$.

Then consider R_1 R_2, R_1R_2, R_1* and R_2*

4: Get a DFA for L. Swap the final and non-final states to get a DFA for L

5: Follows from 1 and 4, as L_1 L_2 = L_1 L_2

6: Follows from 1 and 5, as L_1 L_2 = L_1 L_2

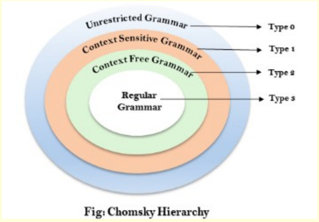


Fig: Chomsky Hierarchy

Properties of Languages

If L_1 and L_2 are context-free languages, then so are

$L_1 \ L_2$, L_1L_2, L_1* and L_2*

Let S_i be the start symbol for a context-free grammar for L_i.

Language	Grammar
$L_1 \ L_2$	$S \ S_1 \mid S_2$
L_1L_2	$S \ S_1S_2$
L_1*	$S \ S_1S \mid$

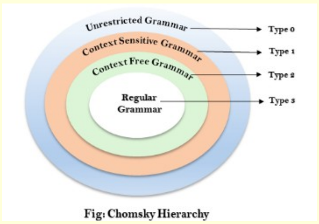


Fig: Chomsky Hierarchy

Example Language

$L = \{ a^i b^j c^k \mid i \; j$ or $j \; k\}$ Is L context-free?

$L = L_1 \; L_2$ where $L_1 = \{ a^i b^j c^k \mid i \; j\}$ and $L_2 = \{ a^i b^j c^k \mid j \; k \}$

So L is context-free if L_1 and L_2 are context-free

One of i and j must be the maximum

So L is context-free

L_1:

maximum is i

S TC
T aTb | A | B
A aA | a
B bB | b
C cC |

maximum is j

L_2:

S AT
T bTc | B | C
A aA |
B bB | b
C cC | c

Properties of Languages

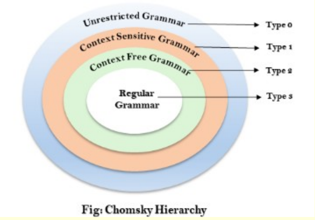


Fig: Chomsky Hierarchy

If R is a regular language and L is context-free, L R is context-free
(Think: still only have to count one thing in L R …)

Build 'composite' PDA as follows.
Let the PDA that recognises L be $M_1 = (Q_1, , , , p_0, F_1)$
Let the DFA that recognises R be $M_2 = (Q_2, , {}_2, q_0, F_2)$
Construct new PDA $M = (Q_1 \times Q_2, , , , [p_0,q_0], \underbrace{F_1 \times F_2}_{\text{composite state of new PDA}})$

So what is ??

$([p,q], a, A) = ([p',q'], B)$ where (p',B) ${}_1(p,a,A)$ and ${}_2(q,a) = q'$
$([p,q], , A) = ([p',q], B)$ where (p',B) ${}_1(p,,A)$

Then show that $([p_0,q_0], w,)$ * $([p_i,q_i], u,)$ iff $(p_0, w,)$ * $(p_i, u,)$ and (q_0, w) * (q_i, u)

Properties of Languages

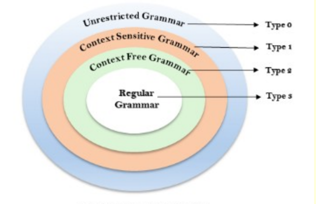


Fig: Chomsky Hierarchy

If L_1 and L_2 are context-free languages, then L_1 L_2, and $\overline{L}$ may not be!

Consider $L_1 = \{ a^i b^j c^k \mid i = j \}$ and $L_2 = \{ a^i b^j c^k \mid j = k \}$

L_1 $L_2 = \{ a^i b^j c^k \mid i = j = k \} = \{ a^i b^i c^i \}$ which is not context-free

It is true though that if R is a regular language and L is context-free, L R is context-free (!!)
(Think: still only have to count one thing in L R ...)

Consider $L = \{ a^i b^j c^k \mid i \; j \text{ or } j \; k \}$

Is $\overline{L}$ context-free? $\overline{L} = L_3$ L_4 where $L_3 = \{w \mid w$ contains ba, ca or $cb\}$ and $L_4 = \{ a^i b^j c^k \mid i = j = k \}$

Now $\overline{L}$ $a^*b^*c^* = L_4$ which is not context-free. So $\overline{L}$ cannot be context-free!

PDAs vs DPDAs

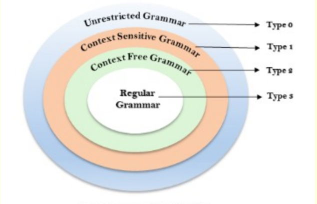


Fig: Chomsky Hierarchy

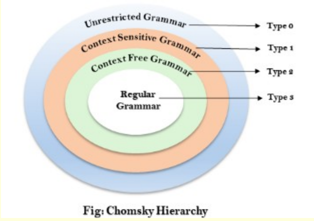
If L can be recognised by a deterministic PDA, so can $\overline{L}$

- Take DPDA for L
- Swap accepting and non-accepting states
- New machine is a DPDA for $\overline{L}$

This means that DPDAs are less powerful than PDAs (!!!)
Assume the opposite, ie every PDA has an equivalent DPDA

1. Let L be a context-free language
2. Then there is a PDA for L
3. By our assumption, there must be a DPDA for L
4. This means that there is a DPDA for $\overline{L}$
5. Then there is a PDA for $\overline{L}$
6. Then $\overline{L}$ is context-free for any language L

So our assumption is wrong, ie **some PDAs have no equivalent DPDA (!!!)**



Chomsky Hierarchy

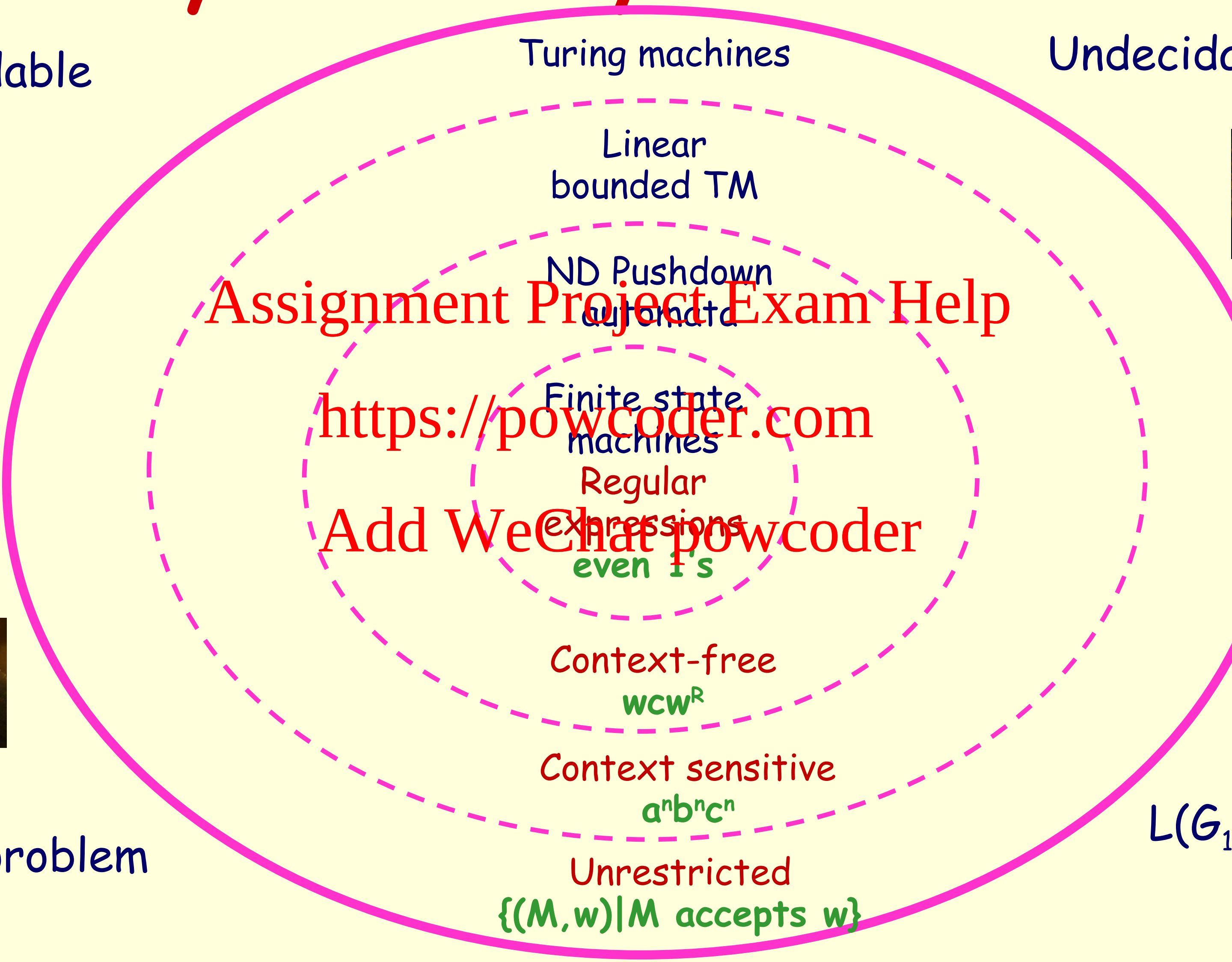
Automata	Languages	Grammars
	Undecidable languages	---
Turing Machines	Recursively enumerable languages	Unrestricted grammars
Linear Bounded Automata	Context-sensitive languages	Context-sensitive grammars
(Nondeterministic) Pushdown Automata	Context-free languages	Context-free grammars
Deterministic Pushdown Automata	?? (Deterministic CF?)	???
Nondeterministic Finite Automata & Deterministic Finite Automata	Regular languages	Regular grammars & regular expressions

Chomsky Hierarchy

Undecidable

Undecidable

L(G) =



Turing machines

Linear bounded TM

ND Pushdown automata

Finite state machines

Regular expressions

even 1's

Context-free

wcw^R

Context sensitive

$a^nb^nc^n$

Unrestricted

{(M,w)|M accepts w}

Halting problem

$L(G_1) = L(G_2)$

Chomsky Hierarchy

Problem Reduction

Automata	Languages	Grammars
---	*Undecidable languages*	---
Turing Machines	Recursively enumerable languages	Unrestricted grammars
Linear Bounded Automata	Context-sensitive languages	Context-sensitive grammars
(Nondeterministic) Pushdown Automata	Context-free languages	Context-free grammars
Deterministic Pushdown Automata	?? (Deterministic CF?)	???
Nondeterministic Finite Automata & Deterministic Finite Automata	Regular languages	Regular grammars & regular expressions

Closure properties

Pumping Lemma

The Platypus Game

https://www.youtube.com/watch?reload=9&v=0gM5TjSOQ48
https://zoneringtones.com

The Platypus Game

Assignment 2

Variations

- Standard (as previously)
- Variable length (50, 200 vs 100 steps)
- Green – score 2 points for changing green to yellow
- Tree – score 5 points for reaching either tree
- Tiebreak – plays an extra match on a random tape for up to 200 steps

Report on your results with 2,500 machines (!!)

OneDrive folder will be shared with you (find the file matching your student number)

Still coming!

Top 10 from each of you will go into the 'knockout' phase

The Platypus Game

3 player tournament

1 vs 1 vs 1
1 vs 1 vs 2
1 vs 1 vs 3
1 vs 2 vs 2
1 vs 2 vs 3
…
1 vs n vs n
2 vs 2 vs 2
2 vs 2 vs 3
…
3 vs 3 vs 3
…
(n-1) vs (n-1) vs (n-1)
(n-1) vs (n-1) vs n
n vs n vs n

$$\sum_{i=1}^{n} i(i+1)/2 = \left(\sum_{i=1}^{n} i^2 + \sum_{i=1}^{n} i\right)/2$$

$$= n(n+1)(2n+1)/12 \quad + n(n+1)/4$$

$$= n(n+1)(n+2)/6$$

When n = 268,
this is **3,244,140**

Around 100 times more than a 2-player tournament!

The Platypus Game

4 player tournament

1 vs 1 vs 1 vs 1
1 vs 1 vs 1 vs 2
…
1 vs 1 vs 1 vs n
1 vs 1 vs 2 vs 2
…
1 vs 2 vs 2 vs 2
…
1 vs n vs n vs n
2 vs 2 vs 2 vs 2
2 vs 2 vs 2 vs 3
…
2 vs n vs n vs n
…
3 vs 3 vs 3 vs n
…
(n-1) vs (n-1) vs (n-1) vs n
n vs n vs n vs n

$${}_{i=1}^{n} i(i+1)(i+2)/6$$

$$= \left({}_{i=1}^{n} i^3 + 3 {}_{i=1}^{n} i^2 + 2 {}_{i=1}^{n} i \right)/6$$

$$= n^2(n+1)^2/4 + n(n+1)(2n+1)/2 + n(n+1)$$

$$= n(n+1)(n+2)(n+3)/24$$

When n = 268,
this is **219,790,485**

Around 10,000 times more than a 2-player tournament!

When n = 90, this is **2,919,735**

Assignment 2

- Detailed specification will be released soon
- Platypus tournament for 2,500 machines
- ‘Second version’ of Universality task from Assignment 1
- Research on at least two intractable problems

Break time! (We resume when all the pictures are gone! This will take 3 minutes!)

I AM BACK!

Alternative Scheme?

Troll	Dreadful	Poor	Acceptable	Exceeds Expectations	Outstanding

Outstanding – CONGRATULATIONS! Your exemplary powers of deduction and a formidable knowledge of the inner workings of the magical world reveal you to be a witch or wizard of genuine skill and learning.

Exceeds Expectations – Well done – a most creditable performance!

Acceptable – demonstrates real magical potential.

Poor – Alas – we regret to inform you that you have narrowly failed. This may have been due to factors outside your control (eg: poltergeist intervention, examination nerves or a malfunctioning quill.) Please do not disconsolate.

Dreadful – We are sorry to inform you that you have failed.

Troll – You would appear either to have abandoned the test due to factors outside your control (eg, earthquake, poltergeist attack), or else you are a troll, in which case you are to be congratulated on being able to use a computer and have achieved the grade of O.F.T. (Outstanding for Trolls).