COMP2012/G52LAC Languages and Computation Lecture 8

Introduction to Context-free Grammars

Henrik Nilsson

University of Nottingham

Non-regular Languages (1)

We have established that the following language is not regular:

$$L = \{0^i 1^i \mid i \in \mathbb{N}\}$$

Non-regular Languages (1)

We have established that the following language is not regular:

$$L = \{0^i 1^i \mid i \in \mathbb{N}\}$$

Others? What about B: the language of "balanced parentheses"? E.g.

$$\begin{aligned} ()() &\in B \\ ((()())()) &\in B \\ )(&\notin B \\ (() &\notin B \end{aligned}$$

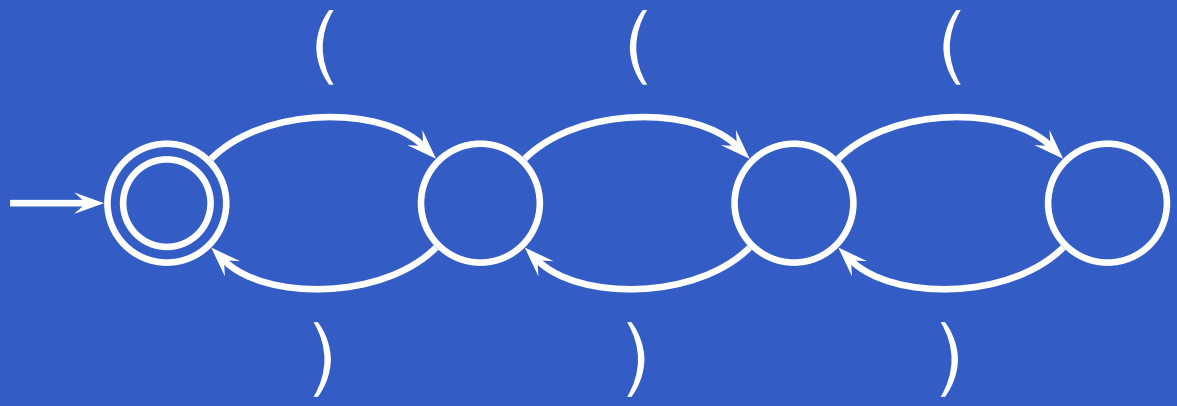
Non-regular Languages (2)

Is B regular?

Non-regular Languages (2)

Is B regular?

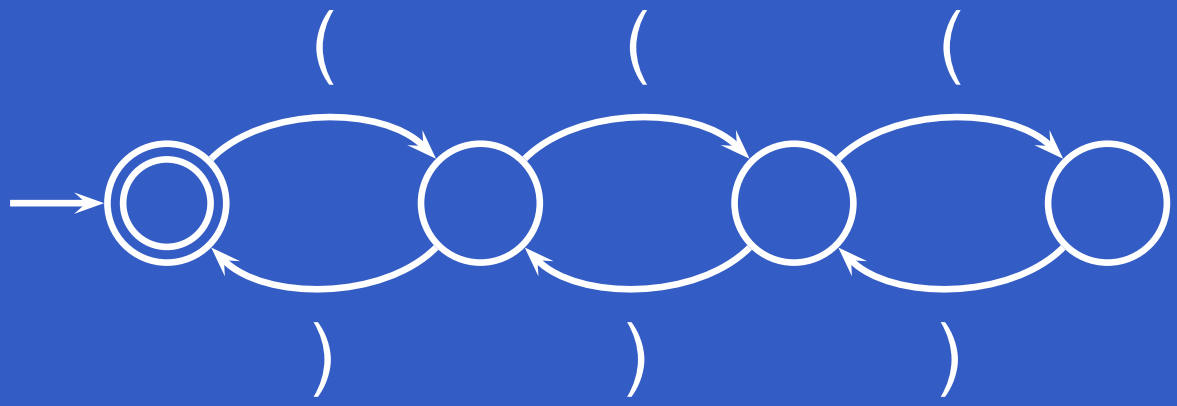
NFA for up to three levels of parentheses:



Non-regular Languages (2)

Is B regular?

NFA for up to three levels of parentheses:

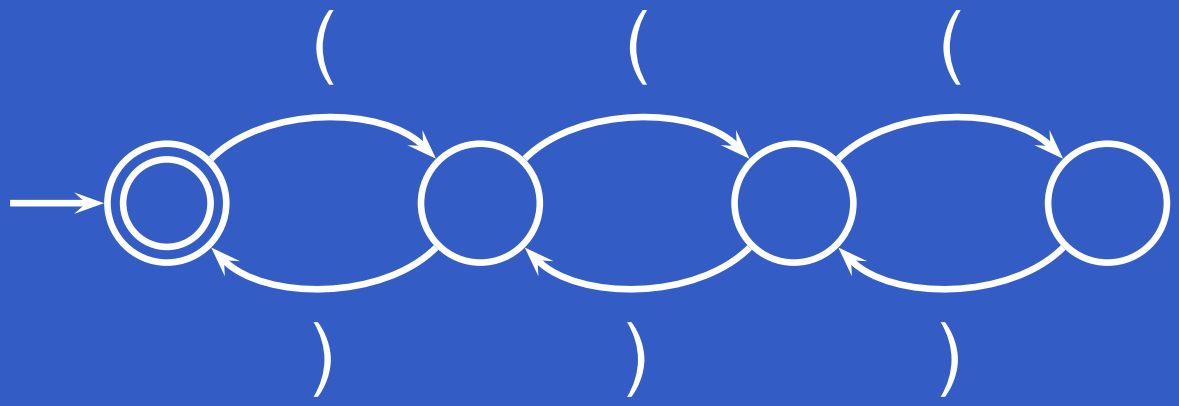


How many states to recognize B? Is B regular?

Non-regular Languages (2)

Is B regular?

NFA for up to three levels of parentheses:



How many states to recognize B? Is B regular?

Use Pumping Lemma for regular languages to formally prove B not regular. ***Exercise!***

Non-regular Languages (3)

But of course, “balanced parentheses” is a key feature of many important classes of languages; e.g.:

- Arithmetic expressions: `(`, `)`
- Matching keywords in programming languages: `begin`, `end`, `repeat`, `until`
- Markup languages; e.g. HTML: `<p>`, `</p>`, `<a href="...">`, `</a>`

Non-regular Languages (3)

But of course, “balanced parentheses” is a key feature of many important classes of languages; e.g.:

- Arithmetic expressions: `(`, `)`
- Matching keywords in programming languages: `begin`, `end`, `repeat`, `until`
- Markup languages; e.g. HTML: `<p>`, `</p>`, `<a href="...">`, `</a>`

Q: Can such languages be described formally? How?

Non-regular Languages (3)

But of course, “balanced parentheses” is a key feature of many important classes of languages; e.g.:

- Arithmetic expressions: `(`, `)`
- Matching keywords in programming languages: `begin`, `end`, `repeat`, `until`
- Markup languages; e.g. HTML: `<p>`, `</p>`, `<a href="...">`, `</a>`

Q: Can such languages be described formally? How?

A: Through ***Context-free Grammars*** (CFG).

Context Free Grammars (CFG)

CFGs originated as an attempt to describe grammars for natural languages like English.

Context Free Grammars (CFG)

CFGs originated as an attempt to describe grammars for natural languages like English.

Key idea: Rules, called ***productions***, that describe how symbols called ***nonterminals*** (or ***variables*** or ***syntactic categories***) can be replaced by nonterminals and ***terminals*** until only terminals left.

Context Free Grammars (CFG)

CFGs originated as an attempt to describe grammars for natural languages like English.

Key idea: Rules, called ***productions***, that describe how symbols called ***nonterminals*** (or ***variables*** or ***syntactic categories***) can be replaced by nonterminals and ***terminals*** until only terminals left.

$$nonterminal \rightarrow terminals \ and \ nonterminals$$

Context Free Grammars (CFG)

CFGs originated as an attempt to describe grammars for natural languages like English.

Key idea: Rules, called ***productions***, that describe how symbols called ***nonterminals*** (or ***variables*** or ***syntactic categories***) can be replaced by nonterminals and ***terminals*** until only terminals left.

$$nonterminal \rightarrow terminals\ and\ nonterminals$$

Let us consider the language ***Grammatically Correct Sentences of Extremely Simplified English*** (GCSESE)

GCSESE (1)

Nonterminals		Terminals
S:	Sentence	boy
NP:	Noun Phrase	girl
VP:	Verb Phrase	little
N:	Noun	big
V:	Verb	walks
		runs
		slowly
		fast

GCSESE (2)

Productions for GCSESE:

$$
\begin{array}{rcl}
S & \rightarrow & NP\ VP \\
NP & \rightarrow & Adj\ NP \\
NP & \rightarrow & N \\
N & \rightarrow & \text{boy} \\
N & \rightarrow & \text{girl} \\
Adj & \rightarrow & \text{little} \\
Adj & \rightarrow & \text{big}
\end{array}
\qquad
\begin{array}{rcl}
VP & \rightarrow & V\ Adv \\
VP & \rightarrow & V \\
V & \rightarrow & \text{walks} \\
V & \rightarrow & \text{runs} \\
Adv & \rightarrow & \text{slowly} \\
Adv & \rightarrow & \text{fast}
\end{array}
$$

GCSESE (2)

Productions for GCSESE:

$$
\begin{array}{rcl}
S & \rightarrow & NP\ VP \\
NP & \rightarrow & Adj\ NP \\
NP & \rightarrow & N \\
N & \rightarrow & \text{boy} \\
N & \rightarrow & \text{girl} \\
Adj & \rightarrow & \text{little} \\
Adj & \rightarrow & \text{big}
\end{array}
\qquad
\begin{array}{rcl}
VP & \rightarrow & V\ Adv \\
VP & \rightarrow & V \\
V & \rightarrow & \text{walks} \\
V & \rightarrow & \text{runs} \\
Adv & \rightarrow & \text{slowly} \\
Adv & \rightarrow & \text{fast}
\end{array}
$$

Note: The terminals constitute the ***alphabet*** of the language being defined.

Definition of CFG

A CFG $G = (N, T, P, S)$ where

- N is a finite set of ***nonterminals*** (or ***variables*** or ***syntactic categories***)
- T is a finite set of ***terminals***
- $N \cap T = \emptyset$ (disjoint)
- P is a finite set of ***productions*** of the form $A \to \alpha$ where $A \in N$ and $\alpha \in (N \cup T)^*$
- $S \in N$ is the ***start symbol***