Parul® University

NAAC A++

Unrestricted Grammars & Turing Machine Equivalence

Chapter 4: Turing machines

Prof. Riddhi Atulkumar Mehta
Assistant Professor
Department of Computer Science and Engineering

Parul® University

NAAC A++

Content

INDEX

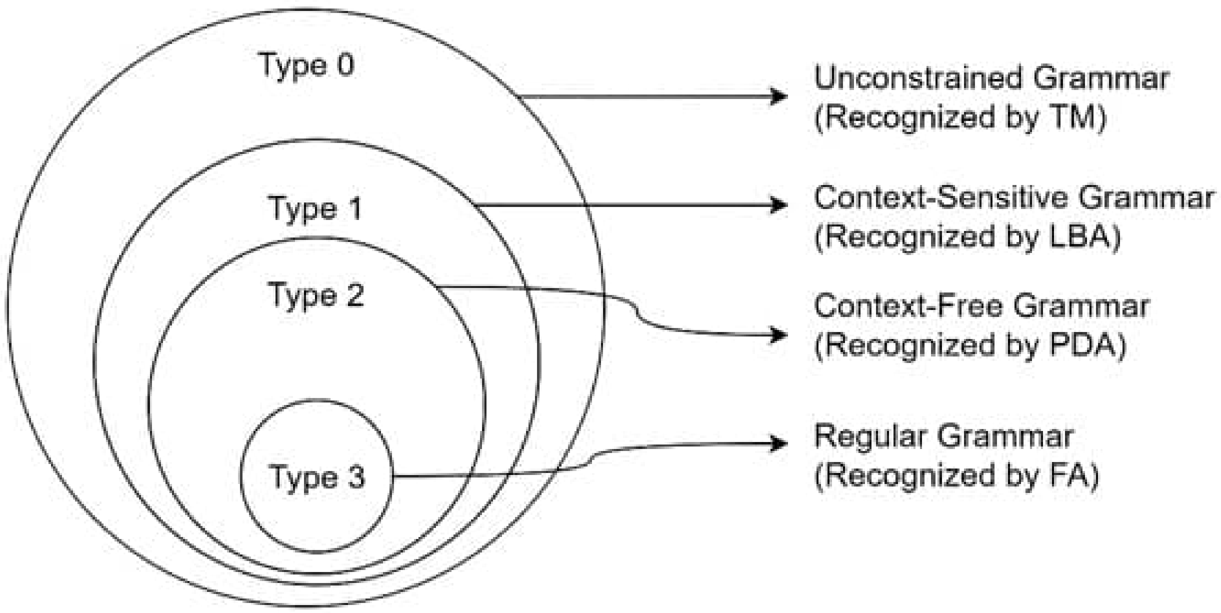
Unrestricted Grammar

- Unrestricted grammar is a type of formal grammar that is defined without any restrictions on the form of its production rules.

- Formally, a grammar $G = (V_N, \Sigma, P, S)$ is called an unrestricted grammar if all its productions are in the form $LS \rightarrow RS$, where LS is a string of non-terminal and terminal symbols, and RS is a string of non-terminal and terminal symbols or the empty string.

- This form of grammar is known as Type 0 grammar in the Chomsky hierarchy, and it is the most general form of grammar.

Unrestricted Grammar



Unrestricted Grammar

- The lack of restrictions allows for a more flexible and powerful method of string generation, which leads to the capabilities of a Turing machine.
- Every language that can be generated by an unrestricted grammar can be recognized by a Turing machine, and vice versa.
- This states the idea that the set of languages generated by unrestricted grammar is equivalent to the set of recursively enumerable languages.

Unrestricted Grammar

An unrestricted grammar (Type-0) is a 4-tuple:

G = (V, **Σ**, R, S) where:

- V: Variables (non-terminals)
- Σ: Terminals
- R: Rules of the form **α → β**, where:
    - $\alpha \in (V \cup \Sigma)^+$ ($\alpha \neq \varepsilon$)
    - $\beta \in (V \cup \Sigma)^*$
- S: Start symbol

Equivalence of Unrestricted Grammars and Turing Machines

- Theorem: A language is generated by an unrestricted grammar if and only if it is recursively enumerable (i.e., it is semidecided by some Turing machine M).
- Proof:
- Only if (grammar → TM): by construction of a nondeterministic Turing machine.
- If (TM → grammar): by construction of a grammar that mimics backward computations of M.

Direction 1 – Grammar ⇒ TM

Given an unrestricted grammar G, we can construct a TM M such that:

- M simulates leftmost derivation of G on input string w
- If derivation leads to w, M accepts

🛠 Method:

- Encode derivations on the TM tape
- Simulate rule application step-by-step

Direction 2 – TM ⇒ Grammar

Given a TM M, we can construct an unrestricted grammar G such that:

- G generates all strings accepted by M

🛠 Idea:

- Simulate TM configurations as strings
- Use productions to mimic TM transitions

Parul® University | NAAC GRADE A++

https://paruluniversity.ac.in/