Characteristics of Axiomatic Systems [Draft]

Mrenal Kanti Das Mridul

Timeline: From April 23, 2025
To [End Date]

Abstract: [to be written later when I finish]

We discuss the mathematical construction of an axiomatic system, or any logical system in general that has some formal statements. We go observing how it behaves, and how initial statements could lead to proofs and other syntactical properties, and all that.

CONTENTS [will be written later]

Author's Note

The author is a self-taught individual with limited resources, currently pursuing his education at a technical school. He writes independently on the topics that sparks his mathematical curiosity, and tries to publish them through the means that are available to him.

Given such background, there might be errors, outdated knowledge, and lack of comprehensive references in his literature. The author humbly asks for the reader's understanding in this regard, and sincerely welcomes any corrections, suggestions, or feedbacks. If need be for any correspondence, please reach out to him via email.

Mrenal Kanti Das Mridul
mrenal@mrenal.onmicrosoft.com

Definition

An **Axiomatic System** is a set of formal statements, i.e. axioms, inference rules, formal language, etc., used to logically derive other statements such as lemma or theorems. An axiomatic system is composed of some key components such as:

1. A **Language**, or a **Formal Language** $\mathcal{L}$ is a set of strings whose symbols are taken from a set called alphabet.

2. Axiom, Postulate, or **Assumption**, $\mathcal{A}x$ are some fundamental logical statements, or propositions that are taken to be true without proofs, to serve as premise or starting points for further reasoning and complex arguments.

3. Rules of Inference, $\mathcal{R}$ are ways and means of logically deriving conclusions from premises.

4. Theorem, $\mathcal{T}$ are statements that are proven, or can be proven through logical arguments and inference rules.

5. Model, $\mathcal{M}$ is a mathematical structure that gives meaning to the symbols and formulas of a formal language.

through which one can approach a statement through formal reasoning without ambiguity that allows one mathematical rigor to study abstract structures.

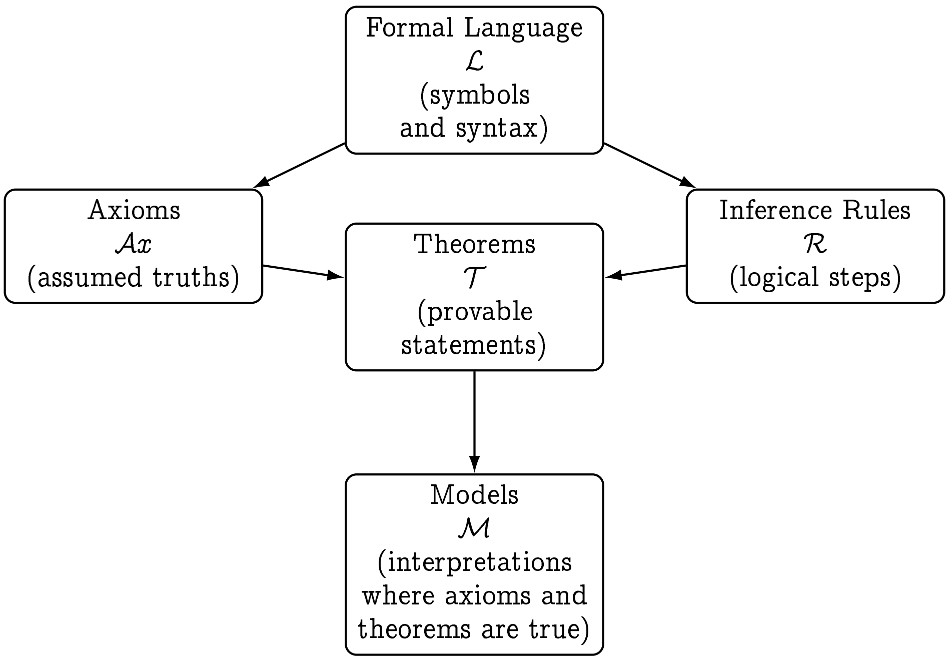

Fig: Components and Flow in an Axiomatic System.

In an axiomatic system, a language ($\mathcal{L}$) provides the formal symbols and rules for building all possible expressions, which one identifies to be the vocabulary to state the ideas precisely. Axioms ($\mathcal{A}x$) are foundational statements written in that language, accepted as true without proofs, and serve as the initial assumptions of the system. Inference rules ($\mathcal{R}$) define how one can logically derive new truths from the old known ones (i.e. the previously proven theorems), ensuring deductive steps and the means of reasoning are valid. From the axioms, and inference rules, one derives theorems ($\mathcal{T}$); the statements that are logically proven through steps within the system. Finally, the models ($\mathcal{M}$), or a model gives concrete interpretation to the abstract symbols of a language in a way that the axioms (and thus all the axioms) hold true. Together, all these components provide a framework where truth can be explored, reasoned about, and verified in a mathematically defined rigorous way.

Mathematical Characterization

A **Formal Language** is a set of strings constructed from a finite set of symbols, defined by precise syntactic rules. So one begins with the basic blocks of any language,

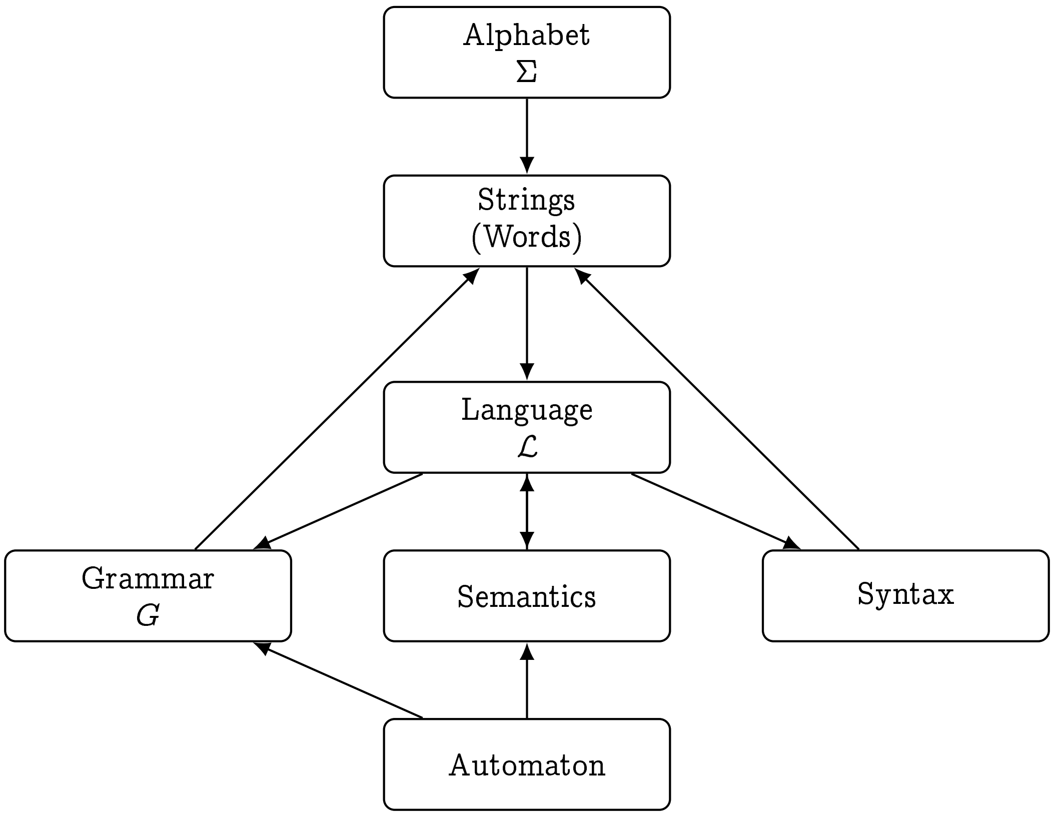

Fig: Components and Flow in a Formal Language

Alphabet

One lets Σ to be a finite, non-empty set called **alphabet**, and each element $a \in \Sigma$ is called a symbol or character. These symbols serve as the building blocks for constructing strings, and consequently a formal language. Mathematically an alphabet is denoted as:

$$\Sigma = \{a_1, a_2, a_3, \cdots, a_n\}$$

where each a_i is an individual symbol. Some examples can be:

$$\Sigma_{\text{binary}} = \{0, 1\}$$

a binary alphabet, which is often used to define binary strings, and $|\Sigma| = 2$. One also has DNA alphabet as a finite set:

$$\Sigma_{\text{DNA}} = \{\text{A, C, G, T}\}$$

where each letter represents one of the four nucleotides found in DNA. One notices that it is a **Finite Alphabet**, where $|\Sigma| = 4$. One also has:

$$\Sigma = \{a, b, c, \cdots, x, y, z\}$$

which is the general English alphabet in lower case, and $|\Sigma| = 26$

Strings

Once one has defined an alphabet Σ, one approaches to construct strings from its elements, as explained in the previous discussion. One has for instance:

$$\Sigma = \{a, b\}$$

then, $a \in \Sigma$ is a symbol, and $ab \in \Sigma^2$ is a string, or a sequence of symbols from Σ.

And given Σ, the set of all strings (finite sequences) that can be formed from the symbols within Σ is denoted as Σ^*, called the **Kleene Closure** of Σ. Defined:

$$\Sigma^* = \bigcup_{n=0}^{\infty} \Sigma^n$$

where Σ^n represents the set of strings of length n over Σ. The set contains all possible combinations of symbols, including the empty string ε, with:

$$\begin{aligned} &\Sigma^0 = \{\varepsilon\} \quad \text{(the empty string)} \\ &\Sigma^1 = \Sigma \quad \text{(all symbols in } \Sigma\text{)} \\ &\Sigma^2 = \{aa, bb, cc, \cdots, xx, yy, zz\} \quad \text{(pairwise concatenation)} \\ \text{and so on...} \end{aligned}$$

Meaning;

$$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \cdots \cup \Sigma^n$$

As example, one knows that Σ_{DNA} is a finite alphabet, and one denotes Σ^*_{DNA} the Kleene star of Σ_{DNA}, i.e., the set of all finite strings over the DNA alphabet:

$$\Sigma^*_{\text{DNA}} = \bigcup_{n=0}^{\infty} \Sigma^n_{\text{DNA}}$$

where,

$$\varepsilon \in \Sigma^*_{\text{DNA}}, \quad \text{A, C, G, T} \in \Sigma^1_{\text{DNA}}, \quad \text{AC, ATG, AGCT}, \cdots \text{ ACGTGC} \in \Sigma^*_{\text{DNA}}$$

which can be modeled as a language $\mathcal{L}_{\text{DNA}}$ that represents a set of biologically meaningful sequences:

$$\mathcal{L}_{\text{DNA}} = \{w \in \Sigma^*_{\text{DNA}} \mid w \text{ encodes a valid protein}\}$$

Grammar

A **grammar** is a formal mechanism for generating the strings of a formal language. In other terms, it is a formal system that describes the syntaxes of a language. A grammar is a 4-tuple:

$$G = (V, \Sigma, P, S)$$

where: V is a finite set of **non-terminal symbols** (also called variables), Σ is a finite set of **terminal symbols**, such that $V \cap \Sigma = \emptyset$, P is a finite set of **production rules**, where each rule is of the form:

$$\alpha \to \beta, \quad \text{with} \quad \alpha, \beta \in (V \cup \Sigma)^*,\ \alpha \neq \varepsilon,$$

and $S \in V$ is the **start symbol**, from which the derivations begin.

The production rules in P specify how strings can be rewritten. For any strings $u, v \in (V \cup \Sigma)^*$, the derivation relation $\Rightarrow$ is defined as:

$$u\alpha v \Rightarrow u\beta v \quad \text{if } \alpha \to \beta \in P, u, v \in (V \cup \Sigma)^*$$

The reflexive-transitive closure of $\Rightarrow$, denoted $\Rightarrow^*$, represents any number of derivation steps (including zero).

Language Generated by a Grammar

The **language generated** by a grammar G is the set:

$$\mathcal{L}(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\},$$

i.e., the set of all strings over Σ that can be derived from the start symbol S using the production rules.

Grammar Types: Chomsky Hierarchy

Depending on the form of the production rules, grammars are classified as follows:

Type	Name	Production Rule Form	Language Class
0	Unrestricted Grammar	$\alpha \to \beta$, $\alpha \neq \varepsilon$	Recursively enumerable
1	Context-sensitive Grammar	$\alpha \to \beta$, $\lvert\alpha\rvert \leq \lvert\beta\rvert$	Context-sensitive
2	Context-free Grammar	$A \to \beta$, $A \in V$	Context-free
3	Regular Grammar	$A \to aB$ or $A \to a$, $A \in V$, $a \in \Sigma$	Regular

Table 1: Chomsky hierarchy of formal grammars

Example of a Context-Free Grammar

Let $G = (V, \Sigma, P, S)$ be a context-free grammar defined as follows:

Component	Definition
Non-terminal Symbols V	$\{S\}$
Terminal Symbols Σ	$\{a, b\}$
Production Rules P	$\{S \to aSb, \quad S \to \varepsilon\}$
Start Symbol S	S

Table 2: Definition of a simple context-free grammar

The language generated by G is:

$$\mathcal{L}(G) = \{a^n b^n \mid n \in \mathbb{N}\}$$

That is, the set of strings consisting of n occurrences of a, followed by n occurrences of b, for any $n \geq 0$.

Example Derivation

To derive the string **aaabbb**, apply the production rules:

$$\begin{aligned} S &\Rightarrow aSb \\ &\Rightarrow aaSbb \\ &\Rightarrow aaaSbbb \\ &\Rightarrow aaa\varepsilon bbb = \text{aaabbb} \end{aligned}$$

Through combinations of an **alphabet** Σ, we form finite sequences called **strings**, which, under the structural rules of a **grammar** G, yield a **Formal Language** $\mathcal{L}$; a set of valid strings generated by the grammar.

$$\Sigma \longrightarrow \Sigma^* \xrightarrow{G} \mathcal{L} \subseteq \Sigma^*$$

More precisely:

$$G \Rightarrow \mathcal{L}(G) \subseteq \Sigma^*$$

is meant here.

Formal Language $\mathcal{L}$

The formal language $\mathcal{L}$ is a symbolic representation used to express axioms, definitions, and theorems. It provides the syntax and grammar for creating statements that belong to the system. For example, one might use symbols like $\forall$ (for "for all"), $\exists$ (for "there exists"), and $\in$ (for "is an element of") within the language $\mathcal{L}$ of set theory.

The axioms are written in this formal language, and the rules of inference (such as Modus Ponens, the law of Detachment, etc.) allow one to derive new propositions from the axioms.

Axioms $\mathcal{A}x$

Axioms are the foundational assumptions of a formal system, which are statements that are accepted as true without requiring any proofs. In formal mathematical terms, axioms are represented and expressed using a formal language $\mathcal{L}$, which consists of symbols and rules for manipulation of those.

Well-Formed Formulas (WFFs)

A **Well-Formed Formula** is a syntactically correct expression constructed from the symbols of a formal language $\mathcal{L}$ according to its specific rules for formation, and manipulation. One lets $\mathcal{L}$ be a formal language with:

1. An **alphabet** Σ consisting of logical symbols (e.g., variables, connectives, quantifiers, parentheses, etc.).
2. A set of **formation rules** defining how symbols may be combined to create valid expressions.

A WFF is any string of symbols from Σ that satisfies all of the formation rules. In other words, WFFs are the *grammatically correct* expressions of the language.

Example: Propositional Logic

One lets the alphabet be:

$$\Sigma = \{p, q, r, \neg, \wedge, \vee, \to, (,)\}$$

which has to have:

1. **Variables**: denoted as $a, y, z, cdots$
2. **Logical Connectives**: $\wedge$ (conjunction), $\vee$ (disjunction), $\neg$ (negation), $\implies$ (implication).
3. **Quantifiers**: $\forall$ (universal quantifier), $\exists$ (existential quantifier).
4. **Constants/Functions/Relations**: Depending on the nature of the system, we may have symbols like constants, functions, and relation symbols, e.g., f, g, R, etc.

The formation rules one has are:

1. Any propositional variable (e.g., p, q, r) is a WFF.
2. If φ is a WFF, then $\neg\varphi$ is a WFF.
3. If φ and ψ are WFFs, then the expressions $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, and $(\varphi \to \psi)$ are WFFs.

then one has some valid WFFs such as:

$$p, \quad \neg p, \quad (p \wedge q), \quad ((p \to q) \vee \neg r)$$

and some non-valid WFFs:

$$\to pq, \quad (p\wedge)$$

One should know that only WFFs can be used meaningfully to formulate the axioms, rules for inference, and then theorems. One also knows that they use syntactically rigorous means, which prevents ambiguous and ill-formed statements in a formal system. The set of WFFs in a formal language is commonly denoted by WFF($\mathcal{L}$).

Axioms $\mathcal{A}x$ in $\mathcal{L}$

The set of axioms, once specified, defines the system's structure and provides the basis for deriving other truths through logical inference. Axioms are considered to be the starting point of a formal system, setting up a base. A formal language is where axioms are written, which has a set of syntaxes and grammar. Axioms define the structure of a system, and how the choice of building on top of it affects the nature of the theory. One knows that when the axioms are set, logical inference is used to derive complex truths processed from the simple ones. The axioms represent the *primitive truths* assumed to be self-evident in a well-formed formal system. Formally, the set of axioms can be written as:

$$\mathcal{A}x = \{\varphi_1, \varphi_2, \dots, \varphi_n\}$$

where each $\varphi_i \in \mathcal{L}$, meaning each axiom φ_i is a well-formed formula in the language $\mathcal{L}$. One can write, an **axiomatic system** is a pair $(\mathcal{L}, \mathcal{A}x)$, where:

$$\mathcal{A}x \subseteq \mathrm{WFF}(\mathcal{L})$$

$\mathcal{L}$ is, of course a formal language, and a formula $\varphi_i \in WFF(\mathcal{L})$ is called an axiom if such is included in the axiom set $\mathcal{A}x$, which is generally chosen to satisfy some conditions, such as:

Consistency: There exists no formula φ for such both φ and $\neg\varphi$ are derivable, or true.

Sufficiency: The axioms are enough to derive all the intended theorems.

Independence: No axiom is derived from the other ones.

One can see that consistency is somewhat, obvious in this case, because a statement, and the negation of that cannot exist in a same framework, nor would it be derivable. The second, the sufficiency ensures that the axioms are powerful enough to derive all possible truths in a system, sometimes which is noted as **completeness**. Independence means each axiom contributes to the system equally, and removine one or a few might reduce the deductive power of the defined system.

Inference Rules $\mathcal{R}$

An inference rule is a syntactic transformation that allows derivation of conclusions from premises in a system. Formally, it can be written as:

$$\frac{\varphi_1, \varphi_2, \dots, \varphi_n}{\psi}$$

This means: if $\varphi_1, \dots, \varphi_n$ are derivable, then so is ψ.

Example: Modus Ponens

One takes an example of Modus Ponens, where

$$\frac{\varphi, \varphi \to \psi}{\psi}$$

means, if φ is a proposition and true, and $\varphi \to \psi$ is also true, then one concludes that ψ is true.

Derivability Relation

The derivability relation $\vdash$ is defined recursively, and we write

1. If $\varphi \in Ax$, then $Ax \vdash \varphi$,
2. If $Ax \vdash \varphi_1, \dots, Ax \vdash \varphi_n$, and a rule $\frac{\varphi_1, \dots, \varphi_n}{\psi} \in R$ exists, then $Ax \vdash \psi$.

to mean: "there exists finite sequances of formulas ending with φ, where each formula is either an axiom from $\mathcal{A}x$, or obtained from previous ones using one or some rules in $\mathcal{R}$". Such relation is called **syntactic entailment,** or **derivability**.

Deductive Closure

The deductive closure of the axiom set is the set of all formulas derivable (proofs) from $\mathcal{A}x$ using rules in $\mathcal{R}$:

$$\mathrm{Th}(\mathcal{A}x) = \{\varphi \in \mathrm{WFF}(\mathcal{L}) \mid \mathcal{A}x \vdash \varphi\}$$

This is the smallest superset of $\mathcal{A}x$ that is closed under all inference rules in $\mathcal{R}$. It contains all axioms $\mathcal{A}x \subseteq Th(\mathcal{A}x)$, and captures all the consequences of the axioms, meaning it defines the entire theory determined by the axioms.

Impact On The System

Inference rules determine the deductive power and internal logic of the system:

- They control which statements can be reached from $\mathcal{A}x$,
- They are essential for defining proof structure and formal reasoning,
- They affect *soundness* (only true conclusions are derivable),
- They affect *completeness* (all true statements are derivable),
- Without them, axioms cannot generate theorems.

References [to be given later]

[Godel1931] Kurt Gödel, *"On Formally Undecidable Propositions of Principia Mathematica and Related Systems"*, Monatshefte für Mathematik, 1931.

[Hilbert1930] David Hilbert, *"The Foundations of Mathematics"*, 1930.