Three Basic Concepts

COT 4210 Discrete Structures II
Summer 2025
Department of Computer Science
Dr. Steinberg

Important Note

The following presentation is not referenced in Sipser, but I believe it is important to first discuss some notations and definitions before diving into the material.

What are the three fundamental ideas?

- There are three fundamental ideas that you will be utilizing throughout the semester.
  - Languages
  - Grammars
  - Automata

Languages

The First Concept

Languages

- Dictionaries defines the **language** as a system suitable for the expression of certain ideas, facts, or concepts, including a set of symbols and rules for their manipulation.
- An **alphabet**, which will be denoted as (Σ), is a finite, non-empty set of symbols.
  - $\Sigma = \{a, b, c\}$, $\Sigma = \{0,1\}$, etc…
- A **string** is a finite sequence of symbols from the alphabet.
  - $aabb, bacc, cab$, etc…

Languages cont.

- **Concatenation** of two strings u and v, denoted uv is the string obtained by adding the symbols in v to the of u.
  - $u = a_1 a_2 \dots a_n$ and $v = b_1 b_2 \dots b_m$ $uv = a_1 a_2 \dots a_n b_1 b_2 \dots b_m$
- **Reverse** of a string w usually denoted w^R, means all symbols in w are in reverse order.
  - $w = aabbac$

$$w^R = cabbaa$$

- **Length** of a string w, usually denoted as $|w|$ represents the number of symbols in w
  - $w = aabbac$

$$|w| = 6$$

- **Empty string**, denoted in Sipser textbook as ϵ, however other textbooks may use λ.
  - IMPORTANT NOTE: I plan to use λ to represent the empty if needed.

Languages cont.

- A **substring** of w is a sequence of consecutive symbols in string w
  - $w = cabac$
    - cab is substring
    - bac is substring
    - aac is NOT substring
- w^n string obtained by concatenating w for n times
  - $w = abc$
    $w^0 = \lambda$
    $w^1 = abc$
    $w^2 = abcabc$
    $\dots$

Languages cont.

- **Star-Closure** of Σ which is denoted as Σ^* contains all strings obtained by concatenating 0 or more symbols in the alphabet all strings formed with symbols from Σ, including λ.
- **Positive-Closure** of Σ which is denoted as Σ^+ contains all strings with symbols from the alphabet, however λ is excluded.
  - $\Sigma = \{a, b\}$
    $\Sigma^* = \{\lambda, a, b, aa, ab, ba, bb, aaa, \ldots\}$
    $\Sigma^+ = \{a, b, aa, ab, ba, bb, aaa, \ldots\}$

Formal Languages

- A language L for an alphabet Σ, is a subset of Σ^*
  - Strings usually have to satisfy certain rules.
- A string in a language is called a **sentence**.
  - $\Sigma = \{a, b\}$
    $L_1 = \{\lambda, a, b, aa, ab, ba, bb\}$ (contains strings with length at most 2)
    $L_2 = \{a^n b^n : n \geq 0, n \in \mathbb{Z}\}$
    $L_3 = \{a^n b^n : n \geq 1, n \in \mathbb{Z}\}$
    $L_4 = \{a^n b^m : m \geq 0, n \geq 0, m, n \in \mathbb{Z}\}$
    $L_5 = \{\lambda, ab, aabb, \ldots\}$ (contains strings that have a number of a's followed by the same number of b's)

Formal Languages

- **Reverse** of a language which is denoted L^R reverses all strings in L
  $L^R = \{w^R : w \in L\}$
  $L = \{aab, baba, baa\}$
  $L^R = \{baa, abab, aab\}$
- **Complement** of a language L which is denoted $\bar{L}$
  - $\bar{L} = \Sigma^* - L$ (set difference) or $\bar{L} = \{w : w \in \Sigma^* \text{ and } w \notin L\}$
- Union, intersection, and difference operations for languages are similar as for the sets.

Formal Languages

- **Concatenation** of two languages L_1 and L_2, denoted L_1L_2 contains any string in L_1 followed by any string in L_2
  $L_1 = \{aa, bb, ab, ba\}$
  $L_2 = \{aba, ca\}$
  $L_1L_2 = \{aaaba, aaca, bbaba, bbca, ababa, abca, baaba, baca\}$
- L^n language obtained by concatenating L for n times.
  $L^0 = \{\lambda\}$
  $L^1 = L$
  $L^2 = LL$
  $L^3 = LLL$
  ...

Grammars

The Second Concept

Grammars

- We need a mechanism to describe a language. In other words, generating the sentences of the respective language.
- A grammar tells us whether a sentence is well formed or not.
- Example: Find a language containing all identifiers, where an identifier

  Contains only lower-case letters or digits

  Must start with a letter

  Examples: $a594$, $zb29$

  $< id > \to < letter >< rest >$

  $< letter > \to a|b|c| \ldots |z$

  $< rest > \to < letter >< rest > |< digit >< rest >| \lambda$

  $< digit > \to 0 \mid 1 \mid 2 \mid \ldots \mid 9$

Generating a Sentence

$< id > \to < letter >< rest >$
$< letter > \to a|b|c| \dots |z$
$< rest > \to < letter >< rest > |< digit >< rest >| \lambda$
$< digit > \to 0 \mid 1 \mid 2 \mid \dots \mid 9$

Generate $b7$

$$< id > \Rightarrow < letter >< rest > \Rightarrow b < rest > \Rightarrow b < digit >< rest > \Rightarrow b7 < rest > \Rightarrow b7\lambda \Rightarrow b7$$

Grammars

- $< id > \rightarrow < letter >< rest >$
  $< letter > \rightarrow a|b|c| \ldots |z$
  $< rest > \rightarrow < letter >< rest > |< digit >< rest >| \lambda$
  $< digit > \rightarrow 0 \mid 1 \mid 2 \mid \ldots \mid 9$
- Variables: $< id >$, $< letter >$, $< rest >$, $< digit >$
- Starting Variable: $< id >$
- Terminals: $a, b, c, \ldots, z, 0, 1, 2, \ldots, 9$
- Production Rules (as seen from above which follows the →)

Definition of Grammar for Formal Languages

- A grammar G is quadruple $G = (V, \Sigma, R, S)$, where
  - V is the finite set called **variables**.
  - Σ is the finite set, disjoint from V, called the **terminals**. This is our <u>**alphabet**</u>!
  - R is the finite set of **rules**, with each rule being a variable and string of variables and terminals
  - $S \in V$ is the **start variable**.
- Given $G = (V, \Sigma, R, S)$, the language generated by G is $L(G) = \{w \in \Sigma^* : s \Rightarrow^* w\}$

Grammar Example 1

$G = (V, \Sigma, R, S)$
$V = \{S\}$
$\Sigma = \{a, b\}$
$R: \begin{cases} S \to aSb \\ S \to \lambda \end{cases}$

What language does this grammar describe?

Grammar Example 1

$G = (V, \Sigma, R, S)$
$V = \{S\}$
$\Sigma = \{a, b\}$
$R: \begin{cases} S \rightarrow aSb \\ S \rightarrow \lambda \end{cases}$

$S \Rightarrow \lambda$

One string we can get is an empty string, but that doesn't tell us much about the language.

Grammar Example 1

$$G = (V, \Sigma, R, S)$$
$$V = \{S\}$$
$$\Sigma = \{a, b\}$$
$$R: \begin{cases} S \to aSb \\ S \to \lambda \end{cases}$$

$$S \Rightarrow \lambda$$
$$S \Rightarrow aSb \Rightarrow a\lambda b \Rightarrow ab$$

The second string we can get is an ‘a’ followed by a ‘b’. We still don’t know much about the language.

Grammar Example 1

$$G = (V, \Sigma, R, S)$$
$$V = \{S\}$$
$$\Sigma = \{a, b\}$$
$$R: \begin{cases} S \rightarrow aSb \\ S \rightarrow \lambda \end{cases}$$

$$S \Rightarrow \lambda$$
$$S \Rightarrow aSb \Rightarrow a\lambda b \Rightarrow ab$$
$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aa\lambda bb \Rightarrow aabb$$

The third string we get are two a's followed by two b's. Based on the rules and strings we can generate, a language can be described.

$$\boldsymbol{L(G) = \{a^n b^n : n \geq 0, n \in \mathbb{Z}\}}$$

Grammar Example 2

Find a grammar G that generates the following language.
$L(G) = \{a^n b^{n+1} : n \geq 0, n \in \mathbb{Z}\}$

Grammar Example 2

Find a grammar G that generates the following language.
$L(G) = \{a^n b^{n+1} : n \geq 0, n \in \mathbb{Z}\}$

Let's use our formal definition of grammars.
$\boldsymbol{G = (V, \Sigma, R, S)}$

Grammar Example 2

Find a grammar G that generates the following language.

$L(G) = \{a^n b^{n+1} : n \geq 0, n \in \mathbb{Z}\}$

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

Based on the language description, we can denote that there are two symbols ‘a’ and ‘b’. This allows us to denote the set $\boldsymbol{\Sigma = \{a, b\}}$

Grammar Example 2

Find a grammar G that generates the following language.

$L(G) = \{a^n b^{n+1} : n \geq 0, n \in \mathbb{Z}\}$

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

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

Now, we need to figure how we can generate the language. Based on observation, we have a sequence of a's followed by the same number plus one of b's.

Grammar Example 2

Find a grammar G that generates the following language.
$L(G) = \{a^n b^{n+1} : n \geq 0, n \in \mathbb{Z}\}$
$G = (V, \Sigma, R, S)$
$\Sigma = \{a, b\}$

We can at least create a rule where we have at least the same number of a's followed by the same number of b's.
$\boldsymbol{S \rightarrow aSb}$

Grammar Example 2

IMPORTANT!
'|' means OR

Find a grammar G that generates the following language.

$L(G) = \{a^n b^{n+1} : n \geq 0, n \in \mathbb{Z}\}$

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

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

$S \to aSb$

We should also consider the scenario where n = 0. If n is assigned to 0, then we should get the string 'b'. That is our basic case. We can update our rule S to include this!

$$S \to aSb \mid b$$

Grammar Example 2

Find a grammar G that generates the following language.

$L(G) = \{a^n b^{n+1} : n \geq 0, n \in \mathbb{Z}\}$

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

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

$S \rightarrow aSb \mid b$

That's it! We were able to provide a grammar that describes the language.

Grammar Example 2

Find a grammar G that generates the following language.

$L(G) = \{a^n b^{n+1} : n \geq 0, n \in \mathbb{Z}\}$

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

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

$S \rightarrow aSb \mid b$

There is also a second solution that exists!

Alternative Grammar

$S \rightarrow Ab$

$A \rightarrow aAb \mid \lambda$

Grammar Example 3

Find a grammar G that generates the following language.
$L(G) = \{a^{2n}b^n : n \geq 0, n \in \mathbb{Z}\}$

Grammar Example 3

Find a grammar G that generates the following language.
$L(G) = \{a^{2n}b^n : n \geq 0, n \in \mathbb{Z}\}$
$G = (V, \Sigma, R, S)$

Let's use our formal definition of grammars.
$\boldsymbol{G = (V, \Sigma, R, S)}$

Grammar Example 3

Find a grammar G that generates the following language.

$L(G) = \{a^{2n}b^n : n \geq 0, n \in \mathbb{Z}\}$

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

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

Based on the language description, we can denote that there are two symbols 'a' and 'b'. This allows us to denote the set $\boldsymbol{\Sigma = \{a, b\}}$

Grammar Example 3

Find a grammar G that generates the following language.

$L(G) = \{a^{2n}b^n : n \geq 0, n \in \mathbb{Z}\}$

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

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

$S \rightarrow aaSb$

Based on the language description, we notice there are twice as many a's followed by b's.

Here is a base string when n = 1, aab

We could create the following rule.

$\boldsymbol{S \rightarrow aaSb}$

Grammar Example 3

Find a grammar G that generates the following language.

$L(G) = \{a^{2n}b^n : n \geq 0, n \in \mathbb{Z}\}$

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

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

$S \rightarrow aaSb \mid \lambda$

We also must consider the case when n = 0. If n is zero, then we have the empty string λ. We would need to update the rule to

$$\boldsymbol{S \rightarrow aaSb \mid \lambda}$$

Grammar Example 3

Find a grammar G that generates the following language.

$L(G) = \{a^{2n}b^n : n \geq 0, n \in \mathbb{Z}\}$

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

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

$S \rightarrow aaSb \mid \lambda$

That's it! We were able to provide a grammar that describes the language.

More Grammar Examples!

Let’s go to the board!

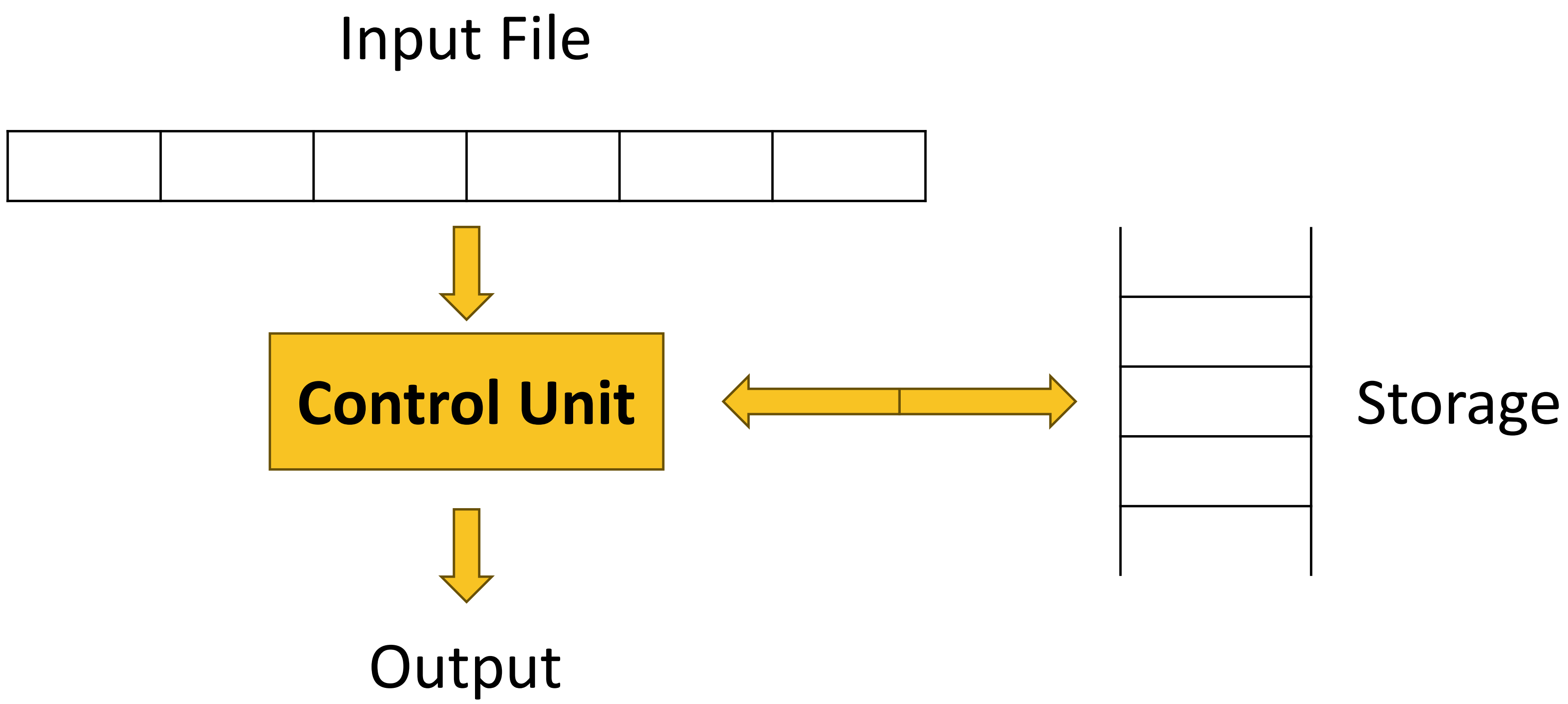
Automata

The Third Concept

Automata is an abstract model for a digital computer

- **Input File**
  - Contains the input string
  - Cannot alter the input
  - Processed from left to right until the end of the string is reached
- **Storage**
  - Infinite # of cell where each cell represents a symbol from the alphabet which can be the same or different than the input alphabet
  - Information can be altered in the storage
- **Control Unit**
  - # of internal states
  - Transition function that decide next internal state.
  - Uses a discrete timeframe
- **Configuration**
  - Current internal state, input file, and storage
- **Move**
  - Transition from on configuration to another

Automata Visual



UCF

Acknowledgement

Some Notes and content come from Dr. Gerber, Dr. Hughes, and Mr. Guha's COT4210 class and the Sipser Textbook, *Introduction to the Theory of Computation*, 3rd ed., 2013