Spring 2021

CS3613: Theory of Automata & Formal Languages

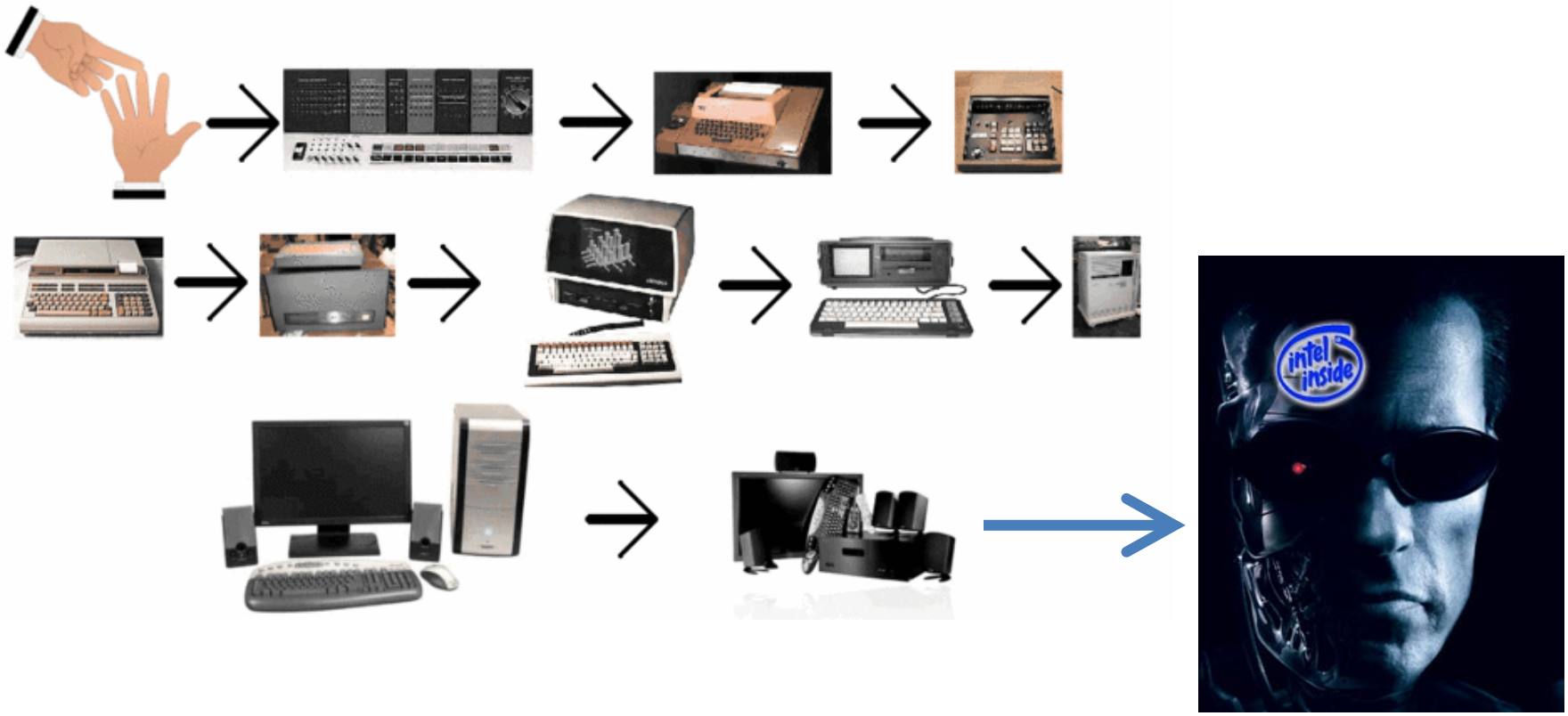
Hashim Ayub

Associate Lecturer

Finite Automata

What is a Computer?

- This question, considering hardware, may have different answers in different eras!



Timeless response

A Computer

≡

An idealized Computer

≡

A Computational Model

Note: Like a Mathematical model, a computational model may only be accurate in certain aspects.

Examples of re-programmable machines

- Jacquard loom
- Babbage Difference Engine
- Colossus by Turing
- DNA Computer
- Quantum Computer
- ……???

The simplest Computational Model

- The simplest Computational Model is called the ***finite state machine*** or ***finite automaton***.

- *To this simplest computational model, lets define its building blocks !*

Finite State Machine or Finite Automaton (FSM)

- A Finite State Machine is a model of computation based on one or more states.
- Only one single state of this machine can be active at the same time.
- It means the machine must transition from one state to another to perform different actions.

Finite State Machine (Cont)

Let us further understand with example

Coin-operated turnstile

- States: locked, unlocked
- Transitions: pointing a coin in the slot will unlock the turnstile, pushing the arm of the unlocked turnstile will let the costumer pass and lock the turnstile again

Traffic Light

- States: Red, Yellow, Green
- Transitions: After a given time, Red will change to Green, Green to Yellow, and Yellow to Red

A Safe

- States: Multiple “locked” states, one “unlocked” state
- Transitions: Correct combinations move us from initial locked states to locked states closer to the unlocked state, until we finally get to the unlocked state. Incorrect combinations land us back in the initial locked state

Few Important Terms

Symbol / Character / Letter: In theory of computation a letter, character or number is:

1) 0,1,2,3,4,5,6,7,8,9,……..
2) a,b,c, ……. , z
3) A,B,C, ……. , Z
4) @,%, &, + , -, …… //special character

Alphabets

- We define an ***alphabet*** to be any non-empty finite set of symbols, usually denoted by Σ.
- Σ is known as sigma.

Examples:

$\Sigma_1 = \{0, 1\}$

$\Sigma_2 = \{a,b,c\ldots, x,y,z\}$

- Σ_1 and Σ_2 are two alphabets
- 0, 1, a, b, c… are called ***letters/symbols/characters***

Strings

- A *word* w is a string of letters from alphabets (Σ) in a linear sequence. eg. 0110 that is taken from $\Sigma\{0,1\}$.
- We are interested only in finite words (bounded length).
- $|w|$ denotes the *length* of word w.
- The *empty string* contains no letters and is written as λ .

Languages

- A *language* L is a set of strings.

- For example, if $\Sigma = \{0,1\}$,

  then $L_1 = \{\varepsilon, 0, 1, 00, 01, 10, 11\}$ is a language

$L_2 = \{0000, 1111\}$ is another language over Σ

A real world example

- The controller of an automatic door is one example of such finite automaton [Sipser07]

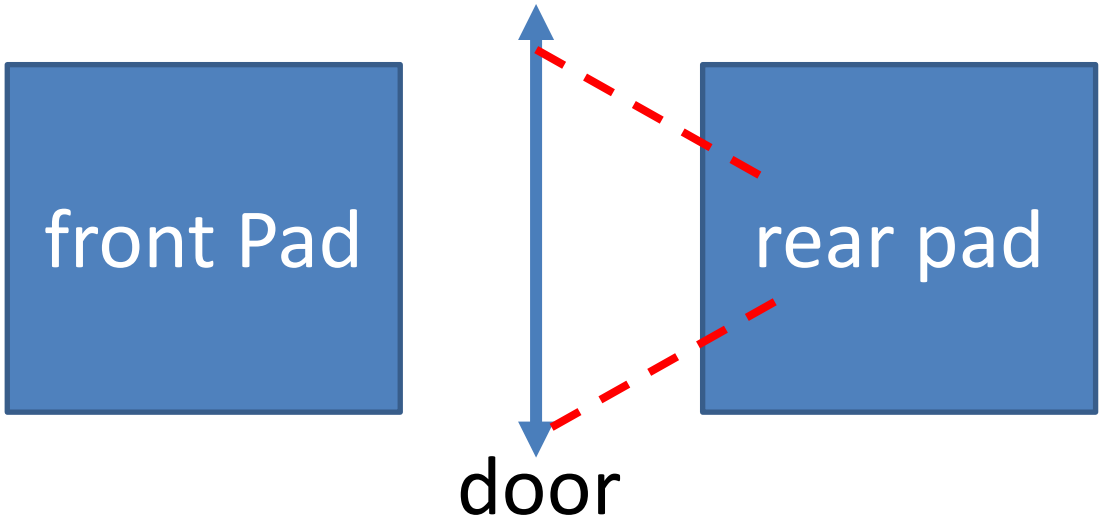


Fig. Top-down view of an automatic door

Kleene Star Closure

- **Kleene Star Closure** of the alphabet Σ, is denoted by "Σ^*".
- Its collection of all possible infinite strings defined over Σ including Null "λ".
- In simple words, It is also known as undetermined power. (0 or more).

- $\Sigma^* = \{a,b\}$

  $\Sigma^* = \{\lambda, a, b, aa, ab, ba, bb, ...\}$

So, strings are infinite many

Undetermined Power means

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

$\Sigma^0 = \{\lambda\}$

$\Sigma^1 = \{a,b\}$

$\Sigma^2 = \{aa, ab, ba, bb\}$

$\Sigma^3 = \{aaa, bbb, aba, ...\}$

$\Sigma^n = \{n\}$

Kleene Star Closure (Cont)

- $\Sigma = \{a,b,c\}$, find Σ^*.

  $\{a,b,c\}^0 = \{\lambda\}$

  $\{a,b,c\}^1 = \{a,b,c\}$

  $\{a,b,c\}^2 = \{aa,ab,ac,ba,bb,bc....\}$

  $\{a,b,c\}^3 = \{aaa, abc, bbb, ccc, ...\}$

  $\{a,b,c\}^n = \{n\}$

  So, string can be written as

  $\Sigma^* = \{\lambda ,a, b, c, aa, ab, ac, .., aaa, abc, cba .... , abca, abba, bcaa.....\}$

Kleene Star Closure (Cont)

- $\Sigma = \{a,b\}$, find Σ^*?

- What about kleene star in string?
- Example s = ab*a (hint) first and last element remains same, only b power will change.

  $\{ab^*a\}^0 = aa$

  $\{ab^*a\}^1 = aba$

  $\{ab^*a\}^2 = abba$

  $\{ab^*a\}^n = ab^na$

So, string can be written as

$\Sigma^* = \{aa, aba, abba, abbba……\}$

Kleene Plus Closure (Cont)

- **Kleene plus operation** is same as Kleene star.
- Only difference here is it does not include null string λ.
- So, it starts with 1 or more power.
- Denoted by Σ^+,
- Example

  s = ab^+a, find Σ^+?

So, string will be written as
Σ^+ = {aba, abba, abbba……}

~~$\{ab^+a\}^0 = aa$~~

$\{ab^+a\}^1 = aba$

$\{ab^+a\}^2 = abba$

……

……

……

$\{ab^+a\}^n = ab^na$

Kleene Plus Closure (Cont)

Summarizing

- **Kleene star vs Kleene plus operation**:
- Σ= {ab*a } , Σ^* and Σ^+.

  Σ^*= {involves null "λ" and other strings}

  Σ^+ = {Null "λ" not included}
- What about Σ= {0,1 } , find Σ^* and Σ^+?
- What about Σ= {aab,c } , Σ^* and Σ^+.

  Σ^*= {λ, aaB, c, aaBaaB, aaBc, caaB, cc, ……}

  Σ^+ = {aaB, c, aaBaaB, aaBc, caaB, cc, ……}

Method of Defining Languages

Back to language concepts, we will be discussing formal languages.

- Formal language works on Rules.

Question here:

- **How to automata accept or reject input?**
- **How can automata know that given string is valid or not?**
- Answer: Language will answer above questions.

So, Language can be defined in following ways:

- Descriptive Language
- Regular Expression (RE)
- Recursive Language
- Finite Automata (FA)
- Transition Graphs (TG)

Method of Defining Languages

- **Descriptive Language** describe the condition imposed on its words.
- Example 1:
- The language "L" of strings of odd length, defined over Σ={a}, can be defined as
- L={a, aaa, aaaaa,.....}
- For even length, defined over Σ={a}.
- We can write L={aa, aaaa, aaaaaa, aaaaaaaa.....}

Method of Defining Languages

- Example 2:
- The language L of strings that does not start with “y” defined over Σ={x,y,z}.
- L={x, z, xz, xy, zy, zyz, zyy, …….}
- Example 3:
- The language L of strings of length 2, defined over Σ={0,1,2} can be written as:
- L={00, 01, 11, 12, 21, 02….}
- Write language L of equal string with number of 1 equal to 0, defined over Σ={0,1}. Also, what about Even-Even scenario?

Method of Defining Languages

- Example 4:
- Write a language $\{a^n b^n\}$, of strings defined over $\Sigma=\{a,b\}$, as $\{a^n b^n: n=1,2,3\}$.
- Hint (use concept power of string)
- L={ab,aabb,aaabbb}

Method of Defining Languages

Recursive Language describe the language with three steps.

- **Rule 1**: Some basic words are defined in language
- **Rule 2**: Rules for constructing more words are defined in the language
- **Rule 3**: No strings excepts those constructed in above two steps are allowed to be in the language.
- Example 1:
- **Define language of positive even numbers using recursive definition.**
- **Step 1:** Basic even words i.e.{2,4,6,8,…..}
- **Step 2:** If x be positive even number in language, then so is, x+2
- **Step 3**: No strings except those constructed in above, are allowed to in positive even number.

Method of Defining Languages

- **Recursive Language** describe the language with three steps.
- **Rule 1**: Some basic words are defined in language
- **Rule 2**: Rules for constructing more words are defined in the language
- **Rule 3**: No strings excepts those constructed in above two steps are allowed to be in the language.

- Example 2:
- **Define language L of strings beginning and ending in same letters, defined over Σ={a,b} using recursive definition.**
- So, language L = {aa, bb, aba, bab, abbaba, babab, ……..}

- **Step 1:** Basic words i.e., a and b is in language L.
- **Step 2:** Beginning and ending with same letter.
  - if start with a: (a) s (a)
  - if start with b: (b) s (b).
  - Note s is string that belong to Σ* (Kleene start).
- **Step 3**: No strings except those constructed in above are allowed to be in language.

Method of Defining Languages

- **Recursive Language** describe the language with three steps.
- **Rule 1**: Some basic words are defined in language
- **Rule 2**: Rules for constructing more words are defined in the language
- **Rule 3**: No strings excepts those constructed in above two steps are allowed to be in the language.
- Solve:
- **Define language of positive odd numbers using recursive definition.**
- **Step 1:** Basic of words i.e.{……}
- **Step 2:** construct more words
- **Step 3**: No strings except those constructed in above, are allowed to in positive odd number.

Method of Defining Languages

- **Recursive Language** describe the language with three steps.
- **Rule 1**: Some basic words are defined in language
- **Rule 2**: Rules for constructing more words are defined in the language
- **Rule 3**: No strings excepts those constructed in above two steps are allowed to be in the language.
- Example 3:
- **Define language L of strings containing aa or bb, defined over Σ={a,b}**
- **Hint** (No specific location is given so) L = {aa, bb, aaa, baa, aab, babaa, ababb,…}
- **Step 1:** aa and bb are in Language L.
- **Step 2:** s(aa)s and s(bb)s are also in L, where s belong to Σ*.
- **Step 3**: No strings except those constructed in above, are allowed to in language.

Questions?

Thank You!