Biçimsel Diller ve Otomata

Giriş

Dr. Ahmet YAZICI
Eskisehir Osmangazi University
Computer Engineering Department
Eskisehir, TURKEY

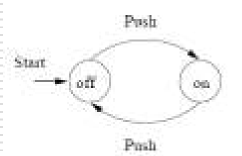
Dersin Amacı

- Biçimsel (Formal) düşünme yeteneği kazandırmak
  - "Biçimsel Diller ve Otomata" ile ilgili temel kavramlar ve modelleri öğretmek.
- Niye Otomata teorisi?
  - Derslerin çoğu günümüzde varolan bilgisayar sistem yapılarına dayanarak anlatılmaktadır.
  - Otomata teorisi, öncelikle problemleri soyut olarak çözmeye yönelik modeller ortaya koymaktadır.
    - Varolan bilgisayar sistemlerinin çeşitli parçaları.
    - Varolan bilgisayar sistemleri.
    - İleridevarolabilecek bilgisayar sistemleri
- Biçimsel diller (formal languages)?
  - Bilgisayar dillerinin tanımı için modeller ortaya koymaktadır

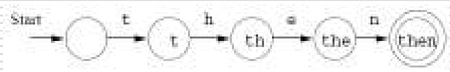
Introduction to Finite Automata

- Finite Automata are used as a model for
  - Software for designing digital cicuits
  - Lexical analyzer of a compiler
  - Searching for keywords in a file or on the web.
  - Software for verifying finite state systems, such as communication protocols.

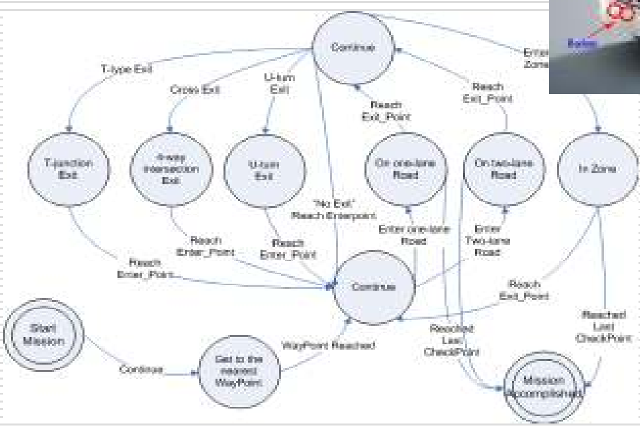
Example: Finite Automaton modeling an **on/of switch**



Example: Finite Automaton recognizing the string **then**



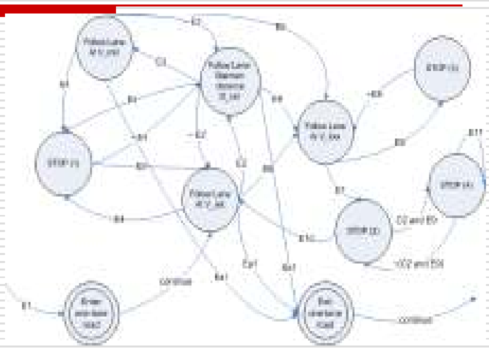
Ex:The Meta-States for OSU-ACT Intelligent Vehicle



Embedded System Application

The Single Lane Meta-State

- E1: Entering an one-lane road
- E2: Slow traffic ahead, distance < 30m
- E3: Slow traffic ahead stopped
- E4: Slow traffic ahead, distance < 2L;
- ~ E4: distance > 2L
- E5: Traffic cleared
- E6: Close to stop line, <2L
- E7: Reached the stop line, < 1m
- E8: Exists precedent stopped vehicle
- E9: Traffic in the cross lane is within 10s
- E10: Intersection cleared,
- E11: Precedent vehicles don't move > 12s+rand() after E10
- Ea1: Reached the Exit_Point



C1: There is a stop sign at the exit point
C2: The cross lane doesn't have stop sign

Structural Representations

These are alternative ways of specifying a machine

- Grammars: A rule like E => E+E species an arithmetic expression

  Eg.:Lineup => Person.Lineup

  says that a lineup is a person in front of a lineup.
- Regular expressions(RE) denote structure of data and used in many systems.
  - E.g., UNIX a.*b.
  - E.g., Document-Type-Definition(DTD)'s describe XML tags with a RE format like person (name, addr, child*).
  - E.g. '[A-Z][a-z]*[][A-Z][A-Z]' matches ***Ithaca NY***
    - []:Blank character, *:any number of

Structural Representations

- Context-free grammars(CFG) are used to describe the syntax of essentially every programming language.
  - Not to forget their important role in describing natural languages.
- And Document-Type-Definition(DTD)'s taken as a whole, are really CFG's.

7

Automata and Complexity

- When developing solutions to real problems, we often confront the limitations of what software can do.
  - *What can a computer do at all?*
    - *Decidable :problem can be solved by computer*
    - *Undecidable* things – no program whatever can do it.
  - *What can a computer do efficiently? (Intractability)*
  - *Tractable: programs are not slower than function that the size of the problem*
  - *Intractable* things – there are programs, but no fast programs.

8

Mathematical Preliminaries

- Sets
- Functions
- Relations
- Graphs
- Proof Techniques

9

SETS

A set is a collection of elements

$A = \{1,2,3\}$ $B = \{train, bus, bicycle, airplane\}$

We write $1 \in A$ $ship \notin B$

C = { a, b, c, d, e, f, g, h, i, j, k } ->finite

C = { a, b, …, k }

S = { 2, 4, 6, … } ->infinite

S = { j : j > 0, and j = 2k for some k>0 }

S = { j : j is nonnegative and even }

10

SETS

Subset: $A = \{ 1, 2, 3\}$ $B = \{ 1, 2, 3, 4, 5 \}$

Proper Subset: $A \subset B$

Disjoint Sets: $A = \{ 1, 2, 3 \}$ $B = \{ 5, 6\}$ $A \cap B = \emptyset$

Set Cardinality(size): $A = \{ 2, 5, 7 \}$, $|A| = 3$

Powersets: $S = \{ a, b, c \}$

Powerset of S = the set of all the subsets of S

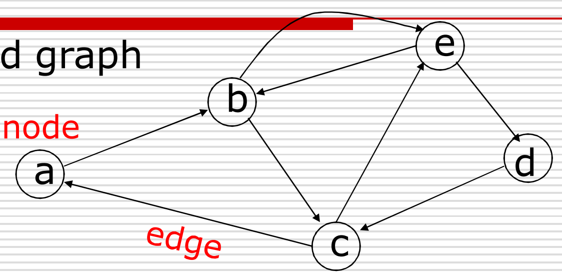
$2^S = \{ \emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\} \}$

Observation: $| 2^S | = 2^{|S|}$ $(8 = 2^3)$

11

GRAPHS

A directed graph



- Nodes (Vertices)

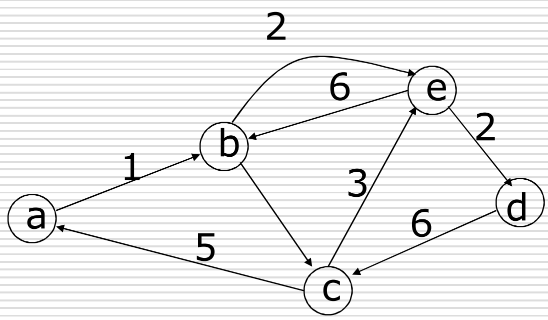
$V = \{ a, b, c, d, e \}$

- Edges

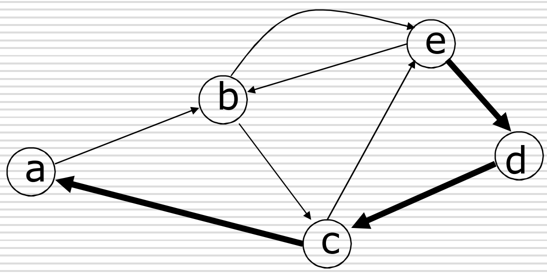
$E = \{ (a,b), (b,c), (b,e),(c,a), (c,e), (d,c), (e,b), (e,d) \}$

12

Labeled Graph



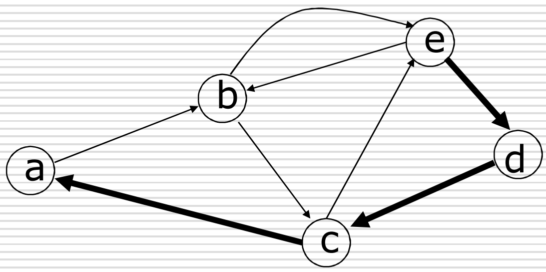
Walk



Walk is a sequence of adjacent edges

(e, d), (d, c), (c, a)

Path

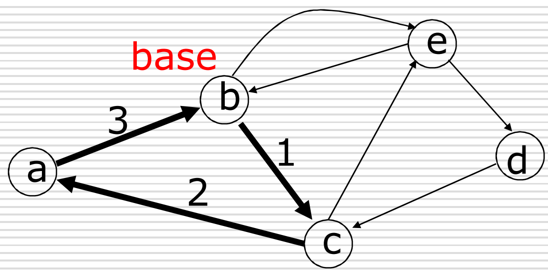


Path is a walk where no edge is repeated

Simple path: no node is repeated

15

Cycle

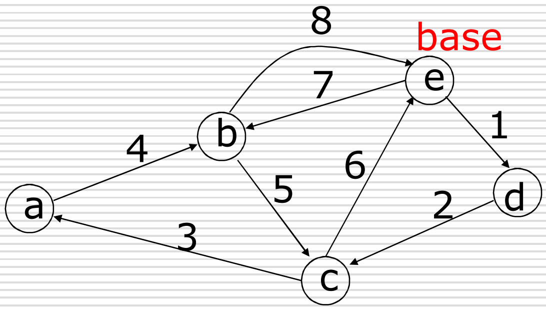


Cycle: a walk from a node (base) to itself

Simple cycle: only the base node is repeated

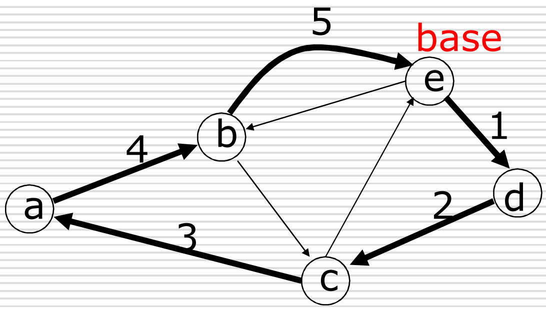
16

Euler Tour



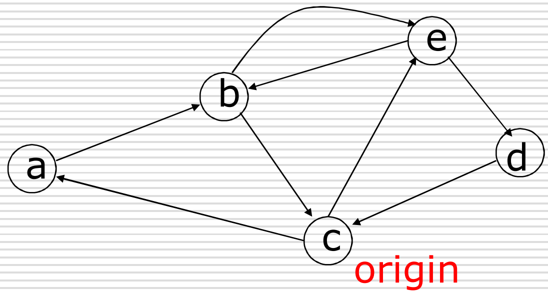
A cycle that contains each edge once

Hamiltonian Cycle



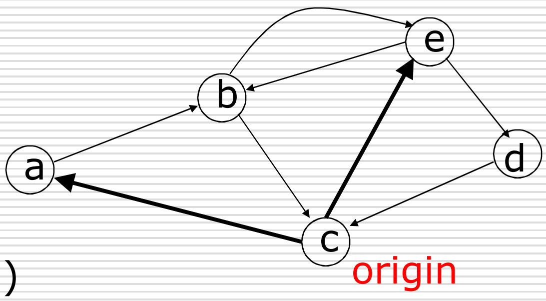
A simple cycle that contains all nodes

Finding All Simple Paths



19

Step 1

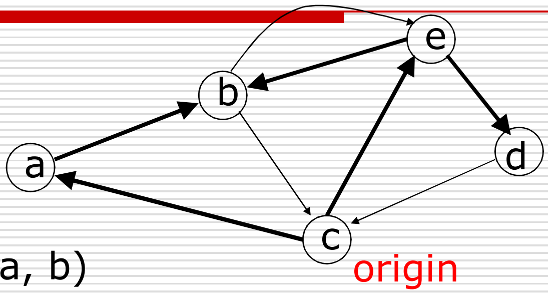


(c, a)

(c, e)

20

Step 2



(c, a)

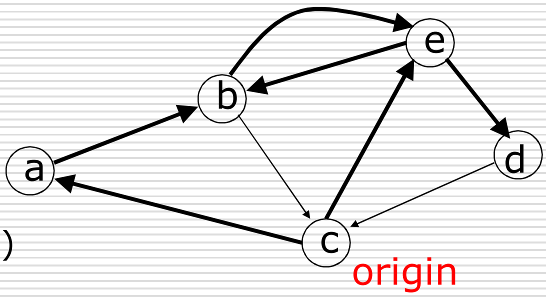
(c, a), (a, b)

(c, e)

(c, e), (e, b)

(c, e), (e, d)

Step 3



(c, a)

(c, a), (a, b)

(c, a), (a, b), (b, e)

(c, e)

(c, e), (e, b)

(c, e), (e, d)

Step 4

(c, a)

(c, a), (a, b)

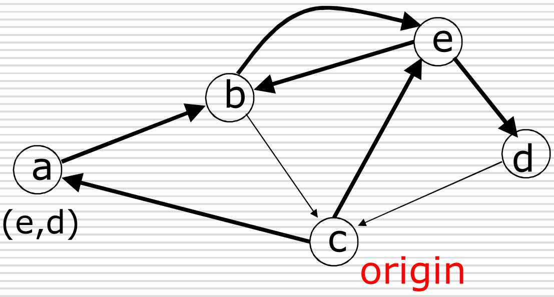
(c, a), (a, b), (b, e)

(c, a), (a, b), (b, e), (e,d)

(c, e)

(c, e), (e, b)

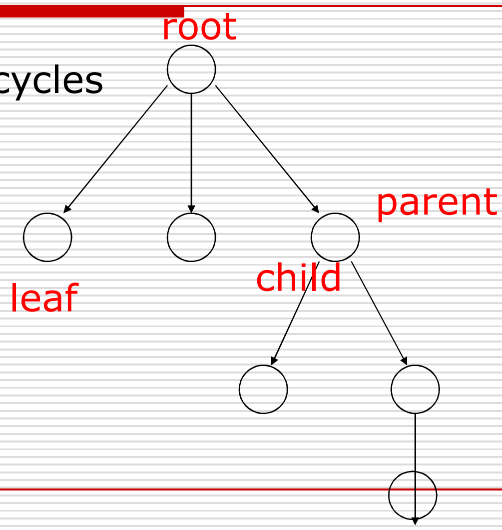
(c, e), (e, d)

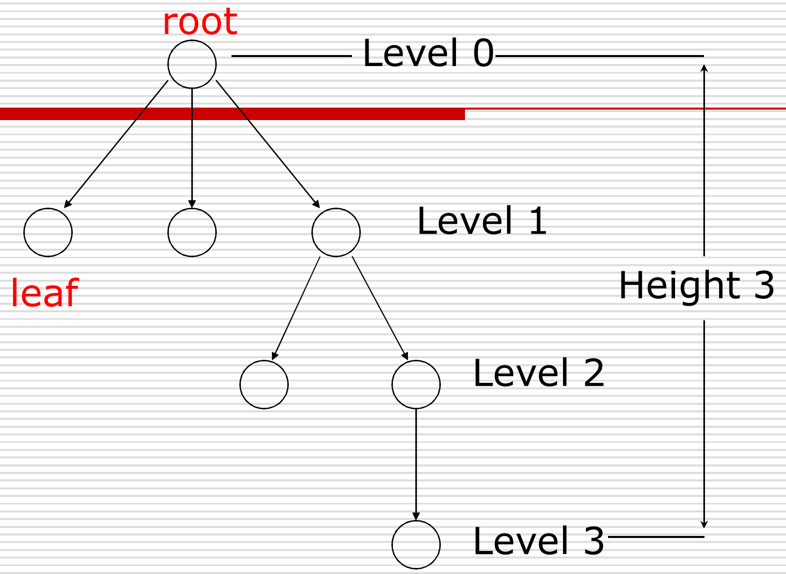


23

Trees

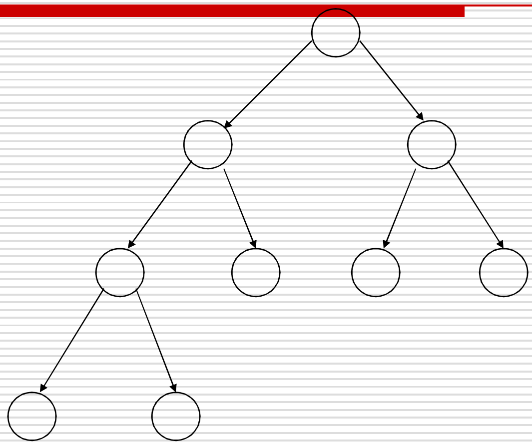
Trees have no cycles





25

Binary Trees



26

PROOF TECHNIQUES

- Proof by induction
- Proof by contradiction

27

Induction

We have statements $P_1, P_2, P_3, \ldots$

If we know

- for some b that $P_1, P_2, \ldots, P_b$ are true
- for any $k >= b$ that

$P_1, P_2, \ldots, P_k$ imply P_{k+1}

Then

Every P_i is true

28

Proof by Induction

- Inductive basis

  Find $P_1, P_2, \ldots, P_b$ which are true

- Inductive hypothesis

  Let's assume $P_1, P_2, \ldots, P_k$ are true,

  for any $k >= b$

- Inductive step

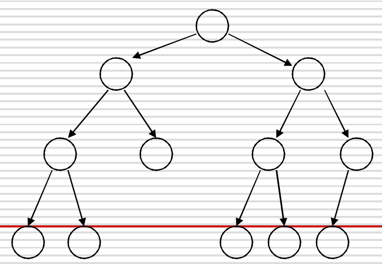
  Show that P_{k+1} is true

Example

Theorem: A binary tree of height n has at most 2^n leaves.

Proof by induction:

let $L(i)$ be the maximum number of leaves of any subtree at height i



We want to show: $L(i) <= 2^i$

- Inductive basis

  $L(0) = 1$ (the root node)

- Inductive hypothesis

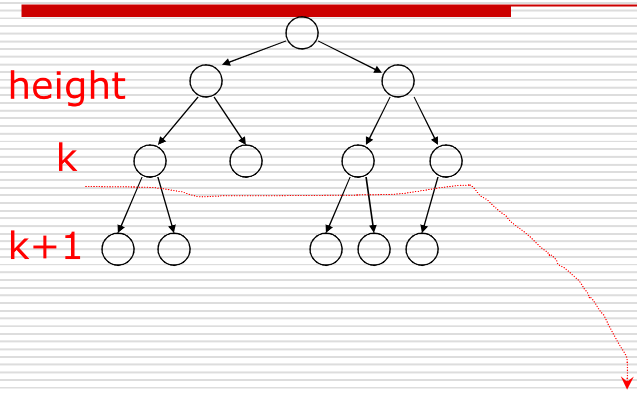
  Let's assume $L(i) <= 2^i$ for all $i = 0, 1, \ldots, k$

- Induction step

  we need to show that $L(k + 1) <= 2^{k+1}$

31

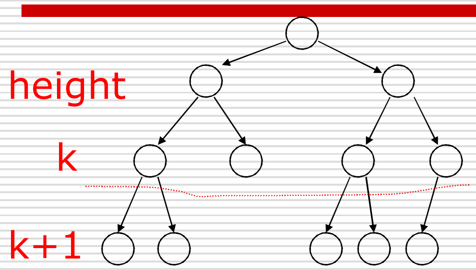
Induction Step



From Inductive hypothesis: $L(k) <= 2^k$

32

Induction Step



$L(k) <= 2^k$

$L(k+1) <= 2 * L(k) <= 2 * 2^k = 2^{k+1}$

(we add at most two nodes for every leaf of level k)

Proof by Contradiction

We want to prove that a statement P is true

- we assume that P is false
- then we arrive at an incorrect conclusion
- therefore, statement P must be true

Example

Theorem: $\sqrt{2}$ is not rational

Proof: Assume by contradiction that it is rational

$\sqrt{2}$ = n/m

n and m have no common factors

We will show that this is impossible

$\sqrt{2}$ = n/m $\Rightarrow$ 2 m^2 = n^2

Therefore, n^2 is even $\Rightarrow$ n is even

n = 2 k

2 m^2 = $4k^2$ $\Rightarrow$ m^2 = $2k^2$ $\Rightarrow$ m is even

m = 2 p

Thus, m and n have common factor 2

Contradiction!

Central Concepts

- **Alphabet**: Finite, nonempty set of symbols
  - Example: Σ = {0,1} binary alphabet
  - Example: Σ = { a, b, c, …,z} the set of all lower case letters
  - Example: The set of all ASCII characters
- **Strings**: Finite sequence of symbols from an alphabet Σ, e.g. 0011001
- **Empty String**: The string with zero occurrences of symbols from Σ
  - The empty string is denoted ε
- **Length of String**: Number of positions for symbols in the string.

  $|w|$ denotes the length of string w

  $|0110| = 4, |\epsilon| = 0$

Powers of an Alphabet: Σ^k = the set of strings of length k with symbols from Σ

Example: $\Sigma = \{0, 1\}$ $\quad \Sigma^1 = \{0, 1\}$ $\quad \Sigma^2 = \{00, 01, 10, 11\}$ $\quad \Sigma^0 = \{\epsilon\}$

Question: How many strings are there in Σ^3

$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \cdots$ $\quad \Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \ldots$ $\quad \Sigma^* = \Sigma^+ \cup \{\epsilon\}$

Concatenation: If x and y are strings, then xy is the string obtained by placing a copy of y immediately after a copy of x
x = a1a2 …ai; y = b1b2 …bj
xy = a1a2 …aib1b2 …bj
Example: x = 01101; y = 110; xy = 01101110

Languages

- If Σ is an alphabet, and $L \subseteq \Sigma^*$ then L is a language
- Examples of languages:
  - The set of legal English words
  - The set of legal C programs
  - The set of strings consisting of n 0's followed by n 1's

    $\{\epsilon, 01, 0011, 000111, \ldots\}$
  - The set of strings with equal number of 0's and 1's

    [illegible]
  - Lp = the set of binary numbers whose value is a prime

    $\{10, 11, 101, 111, 1011, \ldots\}$

Problem (PG.31)

Is a given string w a member of a language L?

Example: Is a binary number prime? = is it a meber in Lp

- Is $11101 \in LP$? What computational resources are needed to answer the question.
- Usually we think of problems not as a yes/no decision, but as something that transforms an input into an output.

Example: Parse a C-program = check if the program is correct, and if it is, produce a parse tree.

- Let LX be the set of all valid programs in proglang X.
- If we can show that determining membership in LX is hard, then parsing programs written in X cannot be easier.