جامعة المستقبل
AL MUSTAQBAL UNIVERSITY

كلية العلوم

قسم الأمن السيبراني

Lecture: (8)

Sequences of sets

Subject: Discrete Structures

First Stage: Semester II

Lecturer: BAQER KAREEM SALIM

Sequences of sets

A sequence is a discrete structure used to represent an ordered list.

For example,

1, 2, 3, 5, 8 is a sequence with five terms (called a *list*)

1, 3, 9, 27, 81 , . . . , 3*n*, . . . is an infinite sequence.

A *sequence* is a function from subset of the set of integers (usually either the set {0, 1, 2, . . .} or the set {1, 2, 3, . . .}) to a set *S*.. The notation a_n is used to denote the image of the integer n that called the term of the sequence and used to describe the sequence . Thus a sequence is usually denoted by

*a*1, *a*2, *a*3, . . .

We describe sequences by listing the terms of the sequence in order of increasing subscripts.

EXAMPLE 1

Consider the sequence {*an*}, where

$$a_n = \frac{1}{n};$$

The list of the terms of this sequence, beginning with *a*1, namely,

*a*1, *a*2, *a*3, *a*4, . . . ,

starts with

$$1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \ldots$$

EXAMPLE 2

a-The sequences {bn} with bn = $(-1)^n$

if we start at n = 0, the list of terms begins with 1,−1, 1,−1, 1, . . .

b-The sequences {cn} with cn = 2×5^n

if we start at n = 0, the list of terms begins with

2, 10, 50, 250, 1250, . . .

c- The sequences {dn} with dn = $6 \times (1/3)^n$

if we start at n = 0, The list of terms begins with

$$6, 2, \frac{2}{3}, \frac{2}{9}, \frac{2}{27}, \ldots.$$

d- The sequences {bn} with $b_n = 2^{-n}$

if we start at n = 0, The list of terms begins with

$$1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \ldots$$

e- The sequences {bn} with $a_n = \frac{1}{n}$

if we start at n = 1, The list of terms begins with

$$1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \ldots$$

RECURSIVELY DEFINED FUNCTIONS

A function is said to be *recursively defined* if the function definition refers to itself. In order for the definition not to be circular, the function definition must have the following two properties:

(1) There must be certain arguments, called *base values*, for which the function does not refer to itself.

(2) Each time the function does refer to itself, the argument of the function must be closer to a base value.

A recursive function with these two properties is said to be *well-defined*.

Definition of Factorial Function:

(a) If $n = 0$, then $n! = 1$.

(b) If $n > 0$, then $n! = n \cdot (n - 1)!$

The definition of $n!$ is recursive, since it refers to itself when it uses $(n - 1)!$. However:

(1) The value of $n!$ is explicitly given when $n = 0$ (thus 0 is a base value).

(2) The value of $n!$ for arbitrary n is defined in terms of a smaller value of n which is closer to the base value 0.

Accordingly, the definition is not circular, or, in other words, the function is well-defined.

EXAMPLE 7: the 4! Can be calculated in 9 steps using the recursive definition .

(1) $4! = 4 \cdot 3!$

(2) $3! = 3 \cdot 2!$

(3) $2! = 2 \cdot 1!$

(4) $1! = 1 \cdot 0!$

(5) $0! = 1$

(6) $1! = 1 \cdot 1 = 1$

(7) $2! = 2 \cdot 1 = 2$

(8) $3! = 3 \cdot 2 = 6$

(9) $4! = 4 \cdot 6 = 24$

Fibonacci Sequence

The Fibonacci sequence is a particularly useful sequence that is important for many applications, including modeling the population growth of rabbits. It is usually denoted by F0, F1, F2, . . and can be defined by:

$$0, \quad 1, \quad 1, \quad 2, \quad 3, \quad 5, \quad 8, \quad 13, \quad 21, \quad 34, \quad 55, \quad \ldots$$

That is, F0 = 0 and F1 = 1 and each succeeding term is the sum of the two preceding terms. For example, the next two terms of the sequence are

$34 + 55 = 89$ and
$55 + 89 = 144$

Fibonacci Sequence can be defined:

(a) If n = 0, or n = 1, then Fn = n.
(b) If n > 1, then Fn = $F_{n-1} + F_{n-2}$.

Where : The base values are 0 and 1, and the value of Fn is defined in terms of smaller values of n which are closer to the base values.
Accordingly, this function is well-defined.

Graphs:

Graphs are discrete structures consisting of vertices and edges that connect these vertices, so a graph G(V,E) consists of:

(i) V, a nonempty set of *vertices* (or *nodes*).

(ii) *E*, a set of *edges*. Each *edge* has either one or two vertices associated with it, called its *endpoints*.

Graphs are used in a wide variety of models with computer science such as communication network, logical design, transportation networks, formal languages, compiler writing and retrieval.
For example: in a communication network, where computers can be represented by vertices and communication links by edges. A graph in which each edge connects two different vertices and where no two edges connect the same pair of vertices is called a **simple graph**.

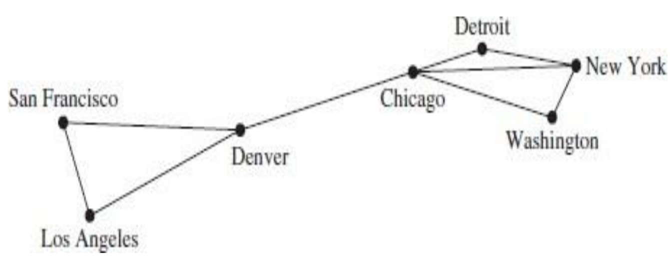


Figure (1): simple graph

A computer network may contain multiple links between data centers, as shown in Figure 2. To model such networks we need graphs that have more than one edge connecting the same pair of vertices. Graphs that may have **multiple edges** connecting the same vertices are called **multigraphs**.

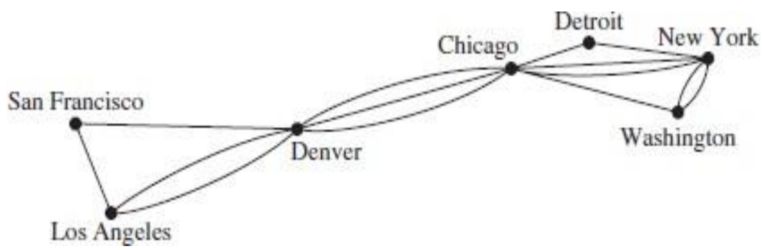


Figure (2): multigraphs

Sometimes a communications link connects a data center with itself, perhaps a feedback loop for diagnostic purposes. Such a network is illustrated in Figure 3. To model this network we need to include edges that connect a vertex to itself. Such edges are called **loops**,

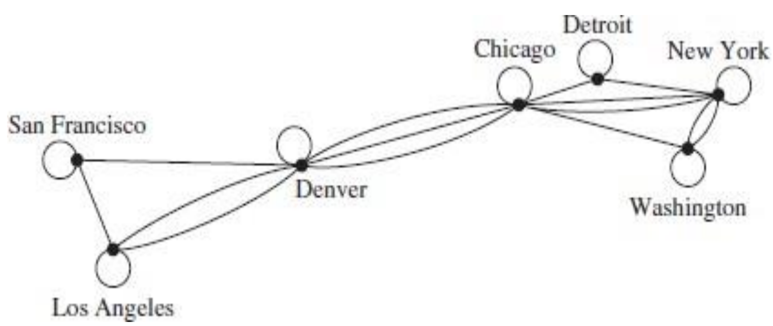


Figure (3): multigraphs with loops

In a computer network, some links may operate in only one direction (such links are called single duplex lines). This may be the case if there is a large amount of traffic sent to some data centers, with little or no traffic going in the opposite direction. Such a network is shown in Figure 4. To model such a computer network we use a **directed graph**. Each edge of a directed graph is associated to an ordered pair.

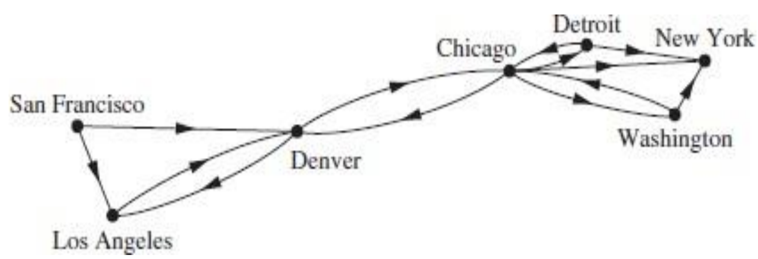


Figure (4): directed graph

For example we have in Figure (5) the graph G(V,E) where: V consists of four vertices A, B, C, D ; and, E consists of five edges

e1 ={A,B},
e2 = {B,C},
e3 = {C, D},
e4 = {A, C},
e5 = {B, D}.

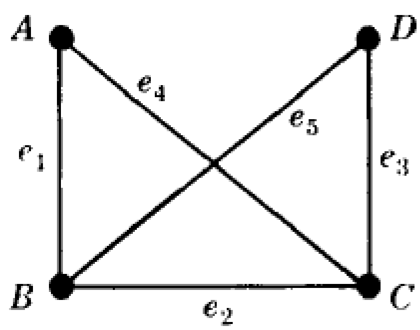
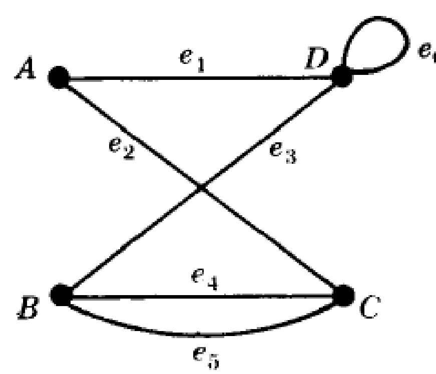


Figure (5)

Vertices u and v are said to be **adjacent** if there is an edge e={u,v}. In such a case, u and v are called the endpoints of e, and e is said to connect u and v. Also, the edge e is said to be **incident** on each of its endpoints u and v.



Subgraphs

Consider a graph G = G(V,E) and a graph H = H(V', E') is called a subgraph of G if the vertices and edges of H are contained in the vertices and edges of G, that is, if V' ⊆ V and E' ⊆ E.

Sometimes we need only part of a graph to solve a problem. For instance, we may care only about the part of a large computer network that involves the computer centers in New York, Denver, Detroit, and Atlanta. Then we can ignore the other computer centers and all telephone lines not linking two of these specific four computer centers. In the graph model for the large network, we can remove the vertices corresponding to the computer centers other than the four of interest, and we can remove all edges incident with a vertex that was removed. When edges and vertices are removed from a graph, without removing endpoints of any remaining edges, a smaller graph is obtained. Such a graph is called a **subgraph** of the original graph.

EXAMPLE 2: The graph *G* shown in Figure 7 is a subgraph of *K*5. If we add the edge connecting a, b, *c* and *e* to *G*, we obtain the subgraph induced by $W = \{a, b, c, e\}$.

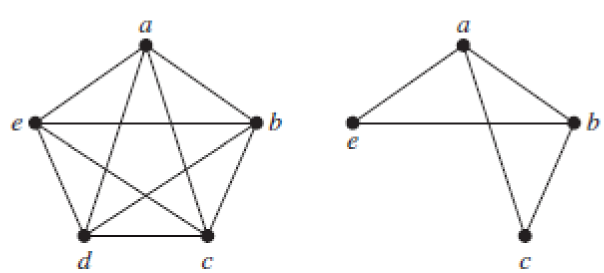


Figure 7