Part III — Decision

Based on lectures by Brian
Notes taken by Dexter Chua

Lent 2017-2018

These notes are not endorsed by the lecturers, and I have modified them (often significantly) after lectures. They are nowhere near accurate representations of what was actually lectured, and in particular, all errors are almost surely mine.

Contents

1 Algorithms

1.1 Use an algorithm given in words

Example. The 'happy' algorithm is

- write down any integer
- square its digits and find the sum of the squares
- continue with this number
- repeat until either the answer is 1 (happy) or until you get trapped in a cycle (not happy).

Show that

(i) 70 is happy

(ii) 4 is unhappy

Proof. (i)

$$\begin{aligned} 7^2 + 0^2 &= 49 \\ 4^2 + 9^2 &= 96 \\ 9^2 + 7^2 &= 130 \\ 1^2 + 3^2 + 0^2 &= 10 \\ 1^2 + 0^2 &= 1 \\ \text{so 70 is happy} \end{aligned}$$

(ii)

$$\begin{aligned} 4^2 &= 16 \\ 1^2 + 6^2 &= 37 \\ 3^2 + 7^2 &= 58 \\ 5^2 + 8^2 &= 89 \\ 1^2 + 4^2 + 5^2 &= 42 \\ 4^2 + 2^2 &= 20 \\ 2^2 + 0^2 &= 4 \\ 4^2 &= 16 \\ \text{so 4 is unhappy} \end{aligned}$$

□

Example. Implement this algorithm

- Let $n + 1, A = 1, B = 1$
- Write down A and B
- Let $C = A + B$

- Write down C
- Let $n = n + 1, A = B, B = C$
- If $n < 5$ go to 3
- If $n = 5$ stop

Proof.

Instruction step	n	A	B	C	Write down
1	1	1	1		
2					1,1
3				2	
4					2
5	2	1	2		
6	Go to step 3				
3				3	
4					3
5	3	2	3		
6	Go to Step 3				
3				5	
4					5
5	4	3	5		
6	Go to Step 3				
3				8	
4					8
5	5	5	8		
6	Continue to Step 7				
7	Stop				

□

Example. The algorithm multiplies the two numbers A and B.

- Make a table with two columns
  Write A in the top row of the left hand column and B in the top row of the right hand column
- In the next row of the table write:
  - in the left hand column, the number that is half A, ignoring remainders
  - in the right hand column the number that is double B
- Repeat step 2 until you reach the row which has a 1 in the left hand column
- Delete all rows where the number in the left hand column is even
- Find the sum of the non-deleted numbers in the right hand column
  This is the product AB

Implement this algorithm when

(i) $A = 29$ and $B = 34$

(ii) $A = 66$ and $B = 56$

Proof. (i)

A	B	Step 4
14	68	Delete
7	136	
3	272	
1	544	
Total	986	

So $29 \times 34 = 986$

(ii)

A	B	Step 4
66	56	Delete
33	112	
16	224	Delete
8	448	Delete
4	896	Delete
2	1792	Delete
1	3584	
Total	3584	

So $66 \times 56 = 3696$

□

1.2 Implement an algorithm given in the form of a flow chat

There are three shapes of boxes which are used in the examination

Example. (i) Implement this algorithm using a trace table

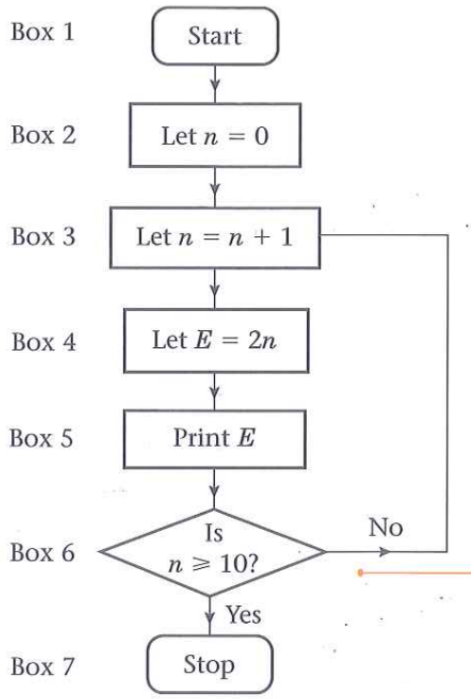
(ii) Alter box 4 to read 'Let $E = 3n$' and implement the algorithm again. How does this alter the algorithm?

Proof. (i)

n	E	box 6
14	68	no
7	136	no
3	272	no
1	544	no
Total	986	no

(ii)

n	E	box 6
66	56	no
33	112	no
16	224	no
8	448	no
4	896	no
2	1792	no
1	3584	no
Total	3584	no



□

1.3 Bubble sort

1.4 Binary search

1.5 Implement the three bin packing algorithms

2 Graphs and networks

2.1 Basic Graph theory

Definition (Vertices and Edges of Graph)**.** In the graph G, above

- the vertices (or nodes) are:
- the edges (or arcs) are:

Definition (Subgraph)**.** A subgraph of G is a graph , each of whose vertices belongs to G and each of whose edges belongs to G. It is simply a part of the original graph.

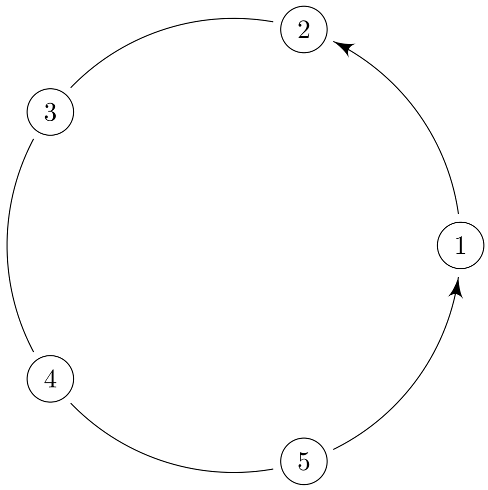
Definition (Degree)**.** The degree or valency or order of a vertex is the number of edges incident to it

Proposition. If the degree of a vertex is even , we say it has even degree

Definition (Path)**.** A path is a finite sequence of edges, such that the end vertex of one edge in the sequence is the start vertex of the next, and in which no vertex appears more than once.

Definition (Walk)**.** A walk is a path in which you are permitted to return to vertices more than once.

Definition (Cycle)**.** A cycle is a closed 'path'.



Definition (Connected)**.** Two vertices are connected if there is a path between them. A graph is connected if all its vertices are connected.

Definition (Loop)**.** A loop is an edge that starts and finishes at the same vertex

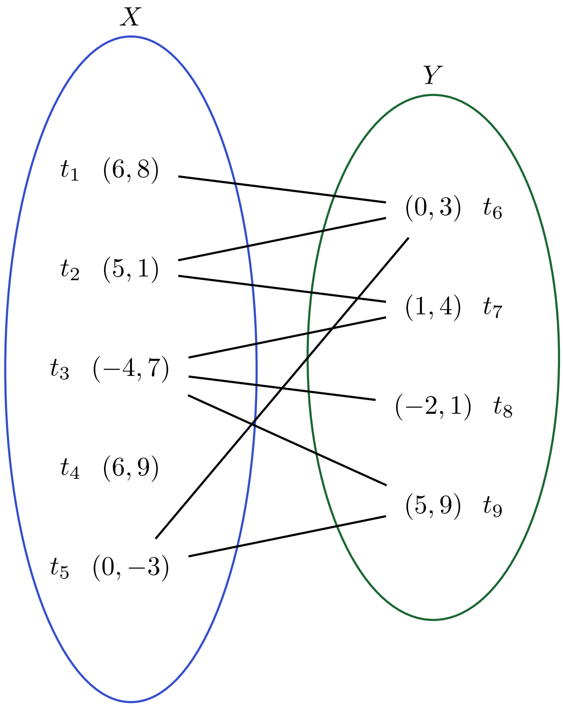
Definition (Simple graph)**.** A simple graph is one in which there are no loops and not have more than one edge connecting any pair of vertices

Definition (Digraph)**.** If the edges of a graph have a direction associated with them they are known as directed edges and the graph is known as a digraph.

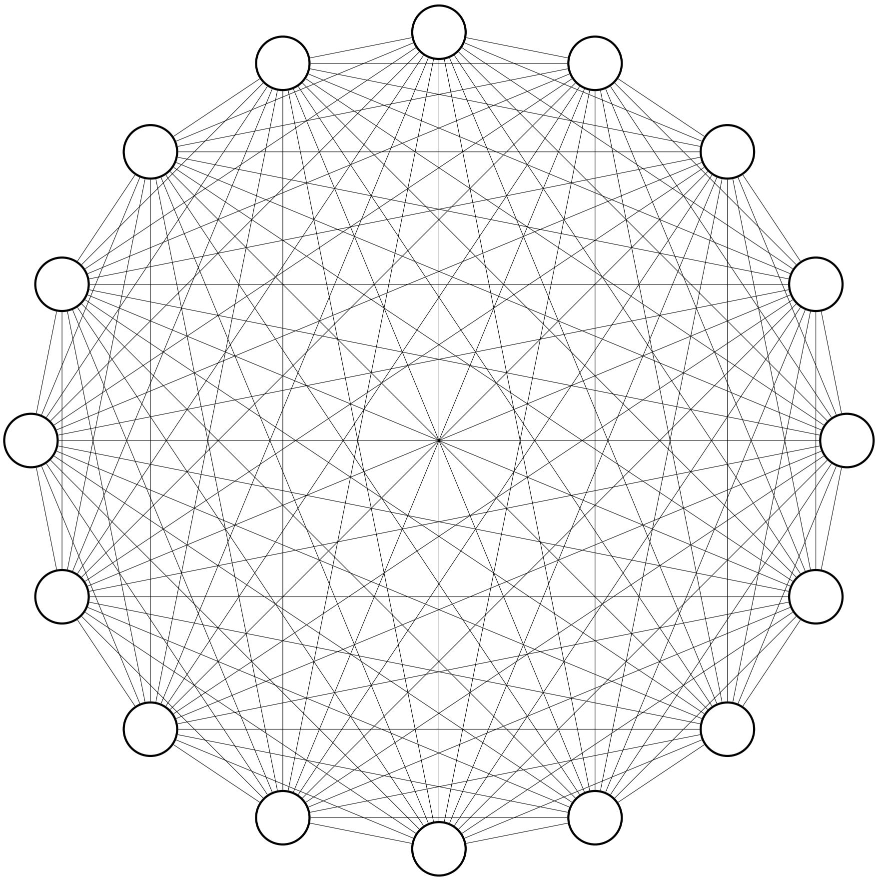
Definition (Tree)**.** A tree is a connected graph with no cycles

Definition (Spanning tree)**.** A spanning tree of a graph ,G is a subgraph which includes all the vertices of G and is also a tree.

Definition (Bipartite graph)**.** A bipartite graph consists of two sets of vertices , X and Y. The edges only join vertices in X to vertices in Y, not vertices within a set



Definition (Complete graph)**.** A complete graph is a graph in which every vertex is directly connected by an edge to each of the other vertices. If the graph has n vertices the connected graph is denoted by K_n.



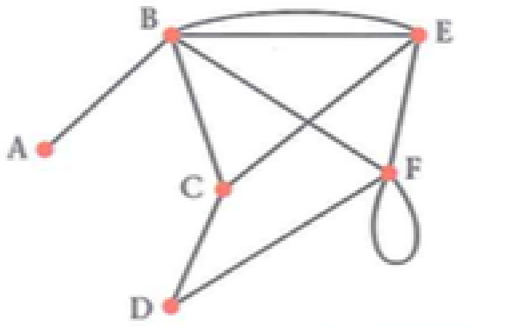
Definition (Complete bipartite graph)**.** A complete bipartite graph (denoted by $K_{r,s}$) in which there are r vertices in set X and s vertices in set Y.

Definition (Isomorphic graphs)**.** Isomorphic graphs are graphs that who the same information but are drawn differently

2.2 Adjacency Matrix and Distance matrix

Definition (Adjacency Matrix)**.** A adjacency matrix records the number of direct links between vertices

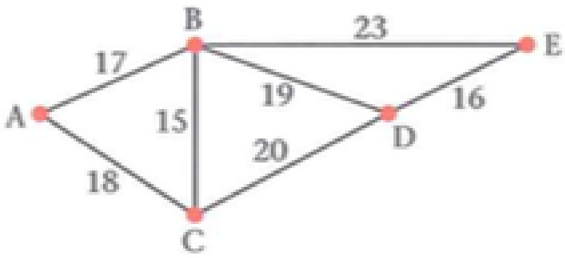
Example. Use an adjacency matrix to represent this graph



	A	B	C	D	E	F
A	0	1	0	0	0	0
B	1	0	1	0	2	1
C	0	1	0	1	1	0
D	0	0	1	0	0	1
E	0	2	1	0	0	1
F	0	1	0	1	1	2

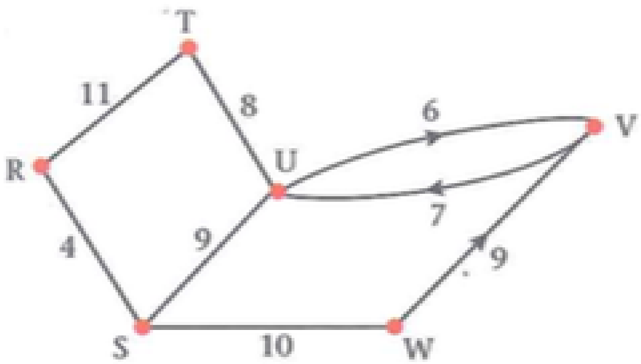
Definition (Distance Matrix)**.** A distance matrix records the weights on the edges. Where there is no edge , we write '−'

Example. Use a distance matrix to represent this network.



	A	B	C	D	E	F
A	0	1	0	0	0	0
B	1	0	1	0	2	1
C	0	1	0	1	1	0
D	0	0	1	0	0	1
E	0	2	1	0	0	1
F	0	1	0	1	1	2

Example. Use a distance matrix to represent this directed network.



	A	B	C	D	E
A	-	17	18	-	-
B	17	-	15	19	23
C	18	15	-	20	-
D	-	19	20	-	16
E	-	23	-	16	-

3 Algorithms on networks

3.1 Kruskal's algorithm

Kruskal's algorithm finds the shortest, cheapest or fastest way of linking all the nodes into one system
You can use Kruskal's algorithm to find a minimum spanning tree.

Definition (Minimum spanning tree)**.** A minimum spanning tree (MST) is spanning tree such that the total length of its edges is as small as possible. (An MST is sometimes called a minimum connectors).

Definition (Kruskal's algorithm)**.** Here is Kruskal's algorithm.

(i) Sort all the edges into ascending order of weight.

(ii) Select the edges of least weight to start the tree

(iii) Consider the next edge of least weight.

  - If it would form a cycle with the edges already selected, reject it.
  - If it does not form a cycle, add it to the tree.

  If there is choice of equal edges, consider each in turn.

(iv) Repeat step 3 until all vertices are connected.

Example. Use Kruskal's algorithm to find a minimum spanning tree

Example.

3.2 Prim's algorithm

Like Kruskal's algorithm, Prim's algorithm finds the minimum spanning tree, but it uses a different approach.

Definition (Prim's algorithm)**.** Here is Prim's algorithm

(i) Choose any vertex to start the tree.

(ii)
  - Select an edges of least weight that joins a vertex that is already in the tree to a vertex that is not yet in the tree.
  - If there is a choice of edges of equal weight , choose randomly.

(iii) Repeat Step 2 until all the vertices are connected.

Example.

You can apply Prim's algorithm to a distance matrix

Definition (Prim's algorithm to a distance matrix)**.** Here is the distance matrix form of Prim's algorithm

(i) Choose any vertex to start the tree.

(ii) Delete the row in the matrix for the chosen vertex.

(iii) Number the column in the matrix for the chosen vertex

(iv) Put a ring round the lowest undeleted entry in the numbered columns. (If there is an equal choice, choose randomly.)

(v) The ringed entry becomes the nest edge to be added to the tree.

(vi) Repeats steps 2,3,4 and 5 until all rows are deleted.

Example.

3.3 Dijkstra's algorithm

Dijkstra's algorithm is used to find the shortest, cheapest or quickest route between two vertices.

Definition (Dijkstra's algorithm)**.** Here is Dijkstra's algorithm
(to find the shortest path form S to T through a network).

(i) Label the start vertex , S , with the final label ,0.

(ii) Record a working value at every vertex , Y, that is directly connected to the vertex , X , that has just received its final label.

  - Working value at Y = final value at $X+$ weight of arc XY
  - If there is already a working value at Y, it is only replaced if the new value is smaller.
  - Ince a vertex has a final label it is not revisited and its working values are no longer considered.

(iii) Look at the working values at all vertices without final labels. Select the smallest working value. This now becomes the final label at that vertex. (If two vertices have the same smallest working value either may be given its final label first.)

(iv) Repeat steps 2 and 3 until the destination vertex , T, receives its final label.

(v) To find the shortest path, trace back from T to S. Given that B already lies on the route, include arc AB whenever final label of B - final label of A = weight of arc AB.

Example.

Example.

Example.

4 Route inspection (Chinese postman problem)

4.1 Traversable

Definition (Eulerian)**.** If all the degrees in a graph are even, then the graph is Eulerian. If precisely two degrees are odd, and all the rest are even, then the graph is semi-Eulerian

Definition (Traversable)**.** A graph is traversable if it is possible to traverse (travel along) every arc just once without taking your pen from the paper.

Proposition. A graph traversable if all the degrees are even

Proposition. A graph is semi-traversable if it has precisely two odd degrees. In this case the start point and the finish point will be the two vertices with odd degrees.

Proposition. A graph is not traversable if it has more than two odd degrees.

Example.

Example.

Example.

4.2 Chinese postman algorithm (route inspection algorithm)

You can use this algorithm to find the shortest route in a network that traverses every arc at least once and returns to the starting point.

Proposition. If all the vertices have even degree the network is traversable. The length of the shortest route will be equal to the weight of the network

Example.

Proposition.

Example.

Example.

Definition (Route inspection algorithm)**.** Here is the route inspection algorithm

(i) Identify any vertices with odd degree.

(ii) Consider all possible complete pairings of these vertices.

(iii) Select the complete pairing that has the least sum.

(iv) Add a repeat of the arcs indicated by this pairing to the network.

Example.

5 Critical path analysis

6 Linear programming

7 Matchings

8 Transportation problems

9 Allocation (assignment) problems

10 The travelling salesman problem

11 Further linear programming

12 Game theory

13 Network flows

14 Dynamic programming