MT365 Examination 2009 **Solutions**

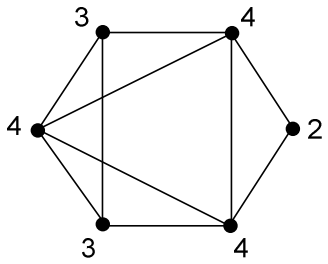
Part 1

Q1 (a) The degree sequence of G is (2, 3, 3, 4, 4, 4)

(b) The number of edges = 10
The sum of the vertex degrees = 2 + 3 + 3 + 4 + 4 + 4 = 20

The sum of the vertex degrees = 2 × The number of edges

(c) A simple connected graph with the same degree sequence is



Q2 (a) (i) The maximum number of vertex disjoint st-paths is 2

set, sft

(ii) The maximum number of edge disjoint st-paths is 6

saeit, set, scfgt, sdeht, sft, sbfjt

(b) Vertex Connectivity $\kappa(G) = 2$, a set of vertices disconnecting G is {e, f}

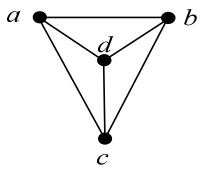
Edge Connectivity $\lambda(G) = 3$, a set of edges disconnecting G is {as, ac, ae}

Q3 (a) The standard 3-orthotope is the unit cube.
The standard 3-dimensional cross polytope is the octahedron.

(b) (i) The standard 4-orthotope has $2^n = 2^4 = 16$ vertices
(ii) The standard 7-dimensional cross polytope has $2n = 2 \times 7 = 14$ vertices.

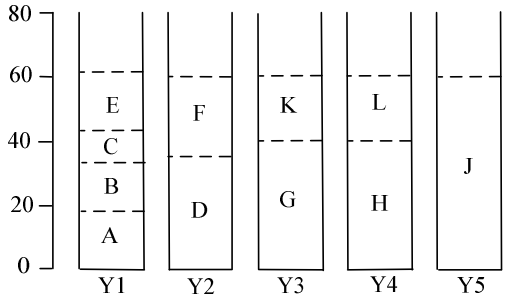
Thus the standard 4-othotope has the greater number of vertices.

Q4 (a) The complete graph K_4 below has $4^{4-2} = 16$ distinct labelled spanning trees.

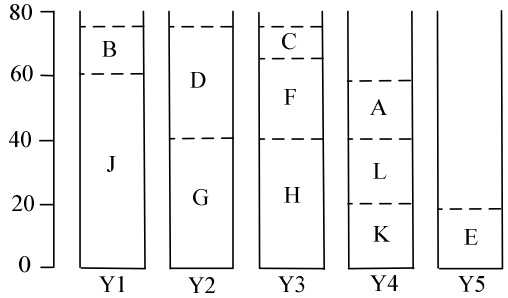


(b) The graph G has $16 \times 16 \times 16 = 2^{12} = 4096$ distinct labelled spanning trees.

Q5 (a) (i) The order of the items is A, B, C, D, E, F, G, H, J, K, L so
An assignment of items to yaks using the first fit packing algorithm is

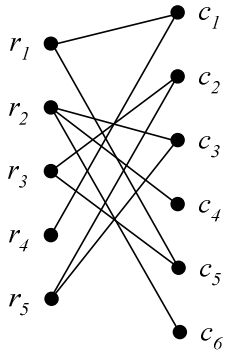


(ii) The order of items the items is J, G, H, D, F, K, L, A, E, B, C so
An assignment of items to yaks using first fit decreasing packing algorithm is



(b) Since both assignments require 5 yaks, the first assignment is preferable because the yaks are more evenly loaded.

Q6 (a) (i) The bipartite graph corresponding to the given braced rectangular framework is



There is a path from each vertex to every other, so the bipartite graph is connected and the framework is rigid.

(ii) There are 11 vertices and 10 edges so the bipartite graph is a tree, so the bracing is a minimum bracing. Alternatively there are no cycles so no edges can be removed without disconnecting the tree so it is a minimum bracing.

(b) Two different Reuleaux pairs that may be involved in opening a bottle are

(i) Cylindric pair – a cork in a bottle of wine

(ii) Screw pair – a metal screw cap on a bottle of lemonade.

Q7 (a) $\chi(G) = 2$ since G is a bipartite graph.

(b) $\chi'(G) = 4$ since G is a bipartite graph and Konig's Theorem applies.

(c) dom(G) = 2 since vertices e and i dominate all vertices.

Q8 (a) Alternating paths are dxcz, dyaw, dyatbxcz.

(b) Improved matchings using each of these paths are respectively

dx, cz, ay, bt. dy, aw, cx, bt. dy, at, bx, cz

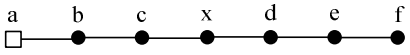
Q9 (a) The other five codewords arise from adding the given three in all possible ways

0000000, 0111001, 1101100, 1010101, 0011110.

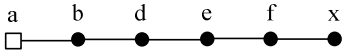
(b) No, the code is not cyclic since e.g. 1001011 is a codeword but 1100101 is not

(c) The minimum distance $\delta = 4$, so the code can correct 1 error and detect 2 errors.

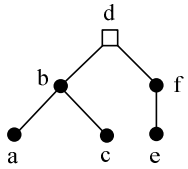
Q10 (a) (i)



(ii)



(b)



Q11 (a) (i)

1	2	3
4	5	6
7	8	9

(ii)

1	2	3	4	5	6	7	8	9	10	11	12
1	4	7	1	2	3	1	2	3	1	2	3
2	5	8	4	5	6	6	4	5	5	6	4
3	6	9	7	8	9	8	9	7	9	7	8

Blocks 1 2 3 from the rows.
Blocks 4 5 6 from the cols.
Blocks 7 8 9 from the 1st and 10 11 12 from the 2nd of the Latin squares

(b) For this design r = 4, v = 9, k = 3 and $\lambda = 1$

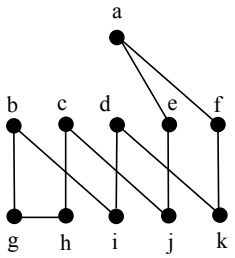
$\lambda (v - 1) = 1 \times 8 = 8$ while $r (k - 1) = 4 \times 2 = 8$ so $\lambda (v - 1) = r (k - 1)$

Part 2

Q12 (a) A Hamiltonian cycle in G containing the three edges bg, gh and hc is

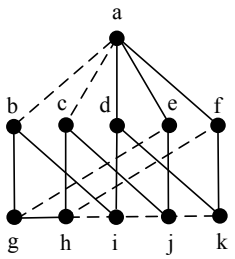
b g h c j e a f k d i b

(b) If the Hamiltonian cycle contains bg, gh, and hc then vertices e and f must involve only edges ae, ej and af, fk respectively, since e and f are of degree 3 and the edges ge and hf are impossible.



Since a involves edges ae and af then edges ab and ac are impossible. Thus b must be joined to i since b is of degree 3 and ba is impossible, while c must be linked to j since c is of degree 3 and ac is impossible.

Now jk is impossible so k must be connected to d since d is of degree 3 and this means d must be linked to i because ad is impossible and d is of degree 3.



There is thus only one Hamiltonian cycle containing the three edges bg, gh and hc.

(c) Since we have a Hamiltonian cycle in G we can use the cycle method to determine planarity.

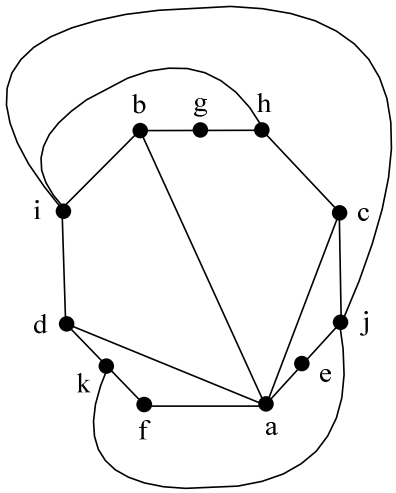
The edges not in the cycle are ab, ac, ad, hi, ij, jk, ge, hf

Inside the cycle :- ab, ac, ad

Outside the cycle :- hi, ij, jk

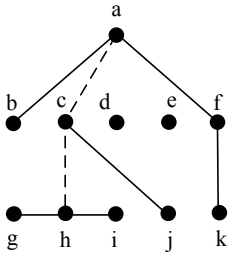
Neither inside nor outside the cycle :- ge, hf

Thus the graph G is non-planar.



(d) If G has a Hamiltonian cycle containing both the edges gh and hi then vertex f must be joined to vertices a and k.

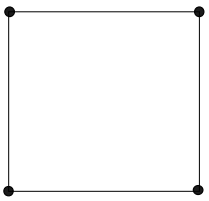
Now vertex b can only be contained within the Hamiltonian cycle by being joined to a and g or to a and i since if b were joined to g and i there would be a closed cycle bghib. Thus b must be joined to a.



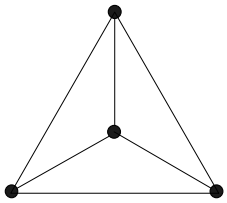
But now vertex c is isolated since c cannot be connected to either a or h and has only one edge connecting it to j remaining

Thus it is impossible to form a Hamiltonian cycle containing both the edges gh and hi

Q13 (a)



A (2, 4)-graph



A (3, 3)-graph

The (2, 4)-graph is the cycle graph C_4, the (3, 3)-graph is the complete graph K_4

(b) Given an (x, y)-graph with n vertices and m edges since every vertex has the same degree x by the handshaking lemma $n\,x = 2\,m$

Given an (x, y)-graph with n vertices and f faces since every face has the same degree y by the handshaking lemma for planar graphs $f\,y = 2\,m$

(c) Since an (x, y)-graph is planar, Euler's formula applies

$$n - m + f = 2$$

But from the two results above $n = \frac{2m}{x}$ and $f = \frac{2m}{y}$

$$\therefore \quad \frac{2m}{x} - m + \frac{2m}{y} = 2 \qquad \therefore \quad \frac{1}{x} - \frac{1}{2} + \frac{1}{y} = \frac{1}{m} \qquad \therefore \quad \frac{1}{x} + \frac{1}{y} = \frac{1}{m} + \frac{1}{2}$$

(d) For a (3, 4)-graph, x = 3 and y = 4 and so

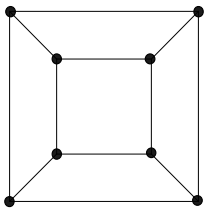
$$\frac{1}{3} + \frac{1}{4} = \frac{1}{m} + \frac{1}{2} \qquad \therefore \quad \frac{1}{m} = \frac{1}{3} + \frac{1}{4} - \frac{1}{2}$$

$$\therefore \quad \frac{1}{m} = \frac{4}{12} + \frac{3}{12} - \frac{6}{12} = \frac{1}{12} \qquad \therefore \quad m = 12$$

Thus $n = \frac{2m}{x} = \frac{2\times 12}{3} = 8$

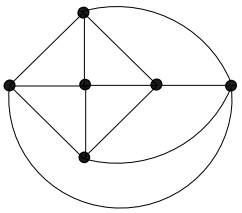
$f = \frac{2m}{y} = \frac{2\times 12}{4} = 6$

So $n = 8,\ m = 12,\ f = 6$



A (3, 4)-graph

(e) To draw a (4, 3)-graph we must find the dual of the (3, 4)-graph, that is we interchange vertices and faces to give



A (4, 3)-graph

This essay is just one version of what could be produced; it is probably more detailed than would be expected and is included to provide an overview of the topic as well as an example.

Q14 An algorithm is a mathematical procedure which, given certain data, follows a sequence of steps, that in a finite time, produces an output and stops. It can be the case that, even though an algorithm is simple and well defined, it is inefficient and useless in practice. The efficiency of an algorithm is measured by the time taken for it to execute and produce a result when implemented on a computer. Such a measure can be found by calculating the time complexity function for the algorithm.

The most time consuming instructions are those which compare two items of data, so we ignore other types of instructions and determine the time complexity function from the number of comparisons. Thus, given an input with n data items, the time complexity function T(n) is a formula for the maximum number of comparisons made in executing the algorithm.

For some algorithms T(n) is found to be proportional to a power of n such as n^2, n^3 or in general n^k, a polynomial function of n. Such algorithms are called polynomial-time algorithms and are said to be in the set **'P'** for short. They are generally efficient and the associated problems are called tractable, although this might not be the case if the coefficient of proportionality is very large. Examples of such tractable algorithms are the Greedy algorithms of Kruskal and Prim for solving minimum connector problems, the Hungarian algorithm for the assignment and transportation problems and the Sorting algorithms such as bubble-sort and merge-sort.

Other algorithms have a T(n) which depends on the n'th power of some number such as 2^n, 3^n, or k^n for some $k > 1$. These are referred to as exponential-time algorithms, they are considered to be inefficient in general and the related problems are called intractable. The exhaustion method, which examines all possible routes, for the travelling salesman problem is the only known method of guaranteeing a solution. For n cities it has a T(n) proportional to $(n - 1)!$, which is greater than 2^n when $n > 5$. Other intractable problems include the Knapsack problem and the Scheduling problem.

Time complexity functions can be arranged in an order hierarchy of sets using the big-oh notation:-

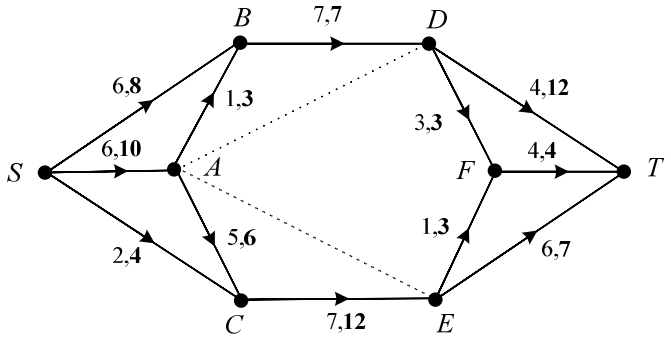
$$O(1) \subset O(\log_2 n) \subset O(n) \subset O(n\log_2 n) \subset O(n^2) \subset \ldots \subset O(n^k) \subset \ldots \subset O(2^n) \subset \ldots \subset O(k^n) \subset \ldots \subset O(n!)$$

the further left in the order T(n) occurs the faster is the algorithm, for large n.

It is often difficult to prove that no polynomial-time algorithm is possible for a given problem, that is whether it is in '**P**' or not. For this reason the theoretical concept of **'NP'** has been introduced, '**NP**' stands for non-deterministic polynomial-time. It allows us to differentiate between problems with a really complicated algorithm and problems with an essentially simple procedure but one that has to be repeated many times because of the extremely large number of choices. The travelling salesman problem is such an **NP** problem, what makes it difficult to solve is not that it is complicated, but that a simple calculation, which can be completed in polynomial time, has to be repeated over and over again a very large number of times.

NP-complete problems are those for which no polynomial-time algorithm is known, but equally no proof has yet been found that such an algorithm does not exist. If we could find a polynomial-time procedure for solving any one of the **NP-complete** problems then such a procedure could be transformed into a polynomial-time algorithm to solve any other NP-complete problem. However, a proof for any NP-complete problem that no polynomial-time algorithm exists implies it cannot exist for any of them. The travelling salesman problem and many other practical problems in industry and commerce are **NP-complete** problems.

Q15

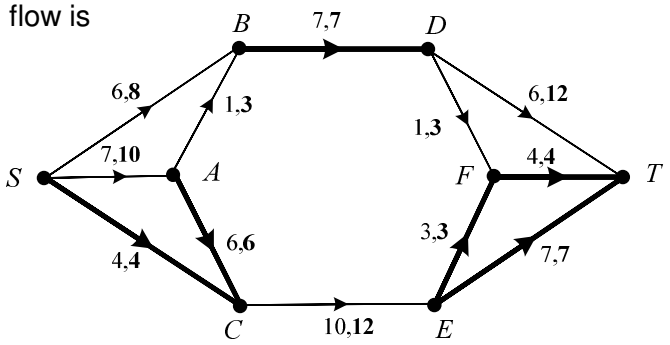


(a) The value of the flow from S to T is 14

(b) Flow Augmenting Paths are

SACEFDT, we can send 1 which saturates AC
SCEFDT, we can send 1 which saturates EF
SCET, we can send 1 which saturates SC and ET

The maximum flow is



The maximum flow has a value of 17

(c) (i) The two minimum cuts are {BD, AC, SC} and {BD, EF, ET}

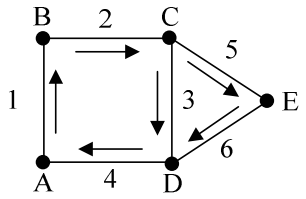
(ii) The arc BD being common to the two minimum cuts is the only single arc for which an increase in capacity would increase the maximum flow. Any other arc would increase the capacity of only one of the minimum cuts and so by the maximum flow minimum cut theorem will not allow an increase in maximum flow

(iii) An increase of 4 in the maximum flow could be achieved in this way since the flow into B could not be increased beyond 8 + 3 = 11 the sum of the capacities of the two arcs SB and AB into B. The greatest increase is by 4 to a value of 21.

(d) Adding an arc of unrestricted capacity from A to D (dotted above) will allow an extra flow of 4. The capacity of the minimum cut {BD, AB, SA, SC} will be 7 + 10 + 4 = 21

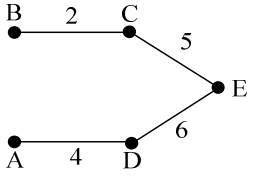
Adding an arc of unrestricted capacity from A to E (dotted above) will not change the Maximum flow since the minimum cut will still be {BD, EF, ET} of capacity 17

Q16

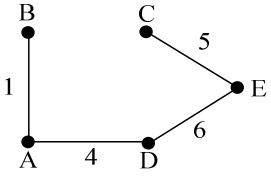


(a) The current equation for vertex D is $i_3 + i_6 - i_4 = 0$

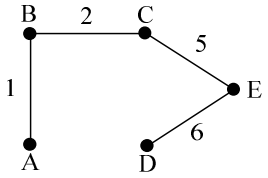
(b) (i) v_5 and v_6 occur in both cycle equations and so 5 and 6 must be branches of the spanning tree. v_3 is the only other edge voltage in the first equation and so 3 must be a chord. From the second equation one of v_1, v_2 and v_4 must correspond to the other chord so there are three possible spanning trees with the other chord as edge 1, 2 or 4.



Spanning Tree {2, 4, 5, 6]



Spanning Tree {1, 4, 5, 6}



Spanning Tree {1, 2, 5, 6}

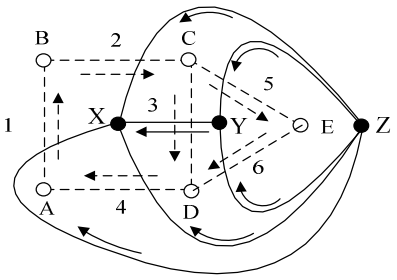
(ii) Choosing the first spanning tree given above, the fundamental cycle matrix is

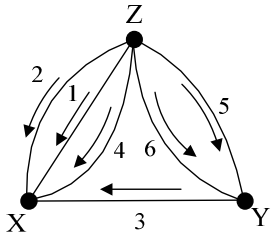
$$\begin{array}{c} \\ 1 \\ 3 \end{array}\begin{array}{c} \begin{array}{cccccc} 2 & 4 & 5 & 6 & 1 & 3 \end{array} \\ \left[\begin{array}{cccc|cc} 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & -1 & -1 & 0 & 1 \end{array}\right] \end{array}$$

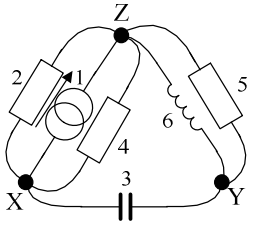
(iii) The fundamental cutset matrix for the first spanning tree given above is

$$\begin{array}{c} \\ 2 \\ 4 \\ 5 \\ 6 \end{array}\begin{array}{c} \begin{array}{cccccc} 2 & 4 & 5 & 6 & 1 & 3 \end{array} \\ \left[\begin{array}{cccc|cc} 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 & -1 & 1 \end{array}\right] \end{array}$$

(d)







The dual oriented graph and dual network in which 2, 4 and 5 are resistors, 1 is a current source, 3 is a capacitor and 6 an inductor.

(e) For the original network capacitor $i_6 = C_6 \dfrac{dv_6}{dt}$ and inductor $v_3 = L_3 \dfrac{di_3}{dt}$ and so

for the dual network capacitor $i_3 = L_3 \dfrac{dv_3}{dt}$ and inductor $v_6 = C_6 \dfrac{di_6}{dt}$

This essay is just one version of what could be produced; it is probably more detailed than would be expected and is included to provide an overview of the topic as well as an example.

Q17 The Hungarian algorithm is a step by step procedure for solving a collection of problems all of which can be modeled by complete bipartite graphs and which include both transportation and assignment problems.

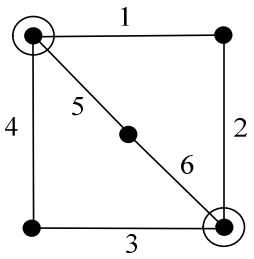
In a transportation problem there are a number of sources, for example factories, from each of which a supply is available, and a number of sinks, such as depots, which have various demands. The costs involved in transporting one unit of goods from each factory to each depot are usually given in a table called a cost matrix. The total supply and demand are equal and the problem is to satisfy the demands by transporting goods from the supply vertices in the cheapest way possible. The solution is arrived at by an iterative procedure. An initial flow pattern is found which uses the cheapest routes available from each supply vertex and to each demand vertex. This flow pattern is a minimum cost flow pattern but is not in general a feasible flow pattern, in the sense that, it does not satisfy all the demands. There will still be supply available at one or more of the supply vertices. In a full iteration of the algorithm the next cheapest supply routes are found and using these, an attempt is made to satisfy the remaining demands by use of one or more flow augmenting paths. If this is not possible, yet another iteration is performed, finding the next cheapest routes. Eventually all the demands are satisfied by the use of the cheapest routes available and the minimum cost solution is determined.

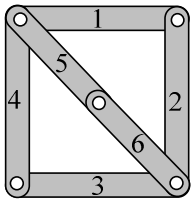
A related problem, solved by the same algorithm, is the assignment problem in which there are a collection of applicants and an equal number of jobs. Each applicant is to be given a job for which they are best suited, their suitability being specified by a cost matrix in which the lower the cost the greater the suitability of an applicant for a job. An initial allocation of applicants to jobs is made on the basis of the cheapest and hence most suitable pairings. This usually does not produce a full allocation or maximum matching and the algorithm proceeds to find the next most suitable pairings and to try to improve the matching by use of an alternating path. The assignment problem is a special case of the transportation problem if a pairing of an applicant to a job is considered a flow of 1 between two vertices.

In both the assignment and transportation algorithms each of the iterations starts with a labelling procedure. In this, vertices with supply (or applicants) available are starred and vertices reachable from them along edges of the partial graph are labelled. Having labelled demand vertices we then label supply vertices reachable from them along edges ‘carrying a flow’ in the transportation problem or ‘in the matching’ for the assignment problem. The labelling continues until a ’breakthrough’ occurs, which means a vertex with demand still to be satisfied is labelled. If this is not achieved then the cost matrix is modified by finding the lowest cost δ available on any labelled supply and unlabelled demand vertex. This introduces new edges to the partial graph on which we then proceed to generate a new labelling. The algorithm always ensures that all allocations are minimum cost.

There are various modifications of the basic algorithm. If the total supply and demand are not equal a dummy demand vertex can be introduced to soak up any excess supply, the flows to it being ignored at the end. Another modification of the transportation problem is called the transhipment problem and allows goods to be sent both to and from all factories and depots. A larger bipartite graph must be used in this case in which each factory and depot appears as both a supply and a demand vertex. Finally, the assignment problem can be modified to the bottleneck problem in which we assign people to jobs on a production line which can operate only as fast as the slowest worker. In this case we need to find a maximum matching in which the time taken by the slowest worker is as small as possible.

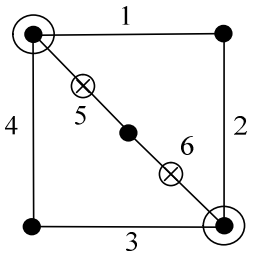
Q18 The direct graph of the given planar kinematic system has the ternary joints circled

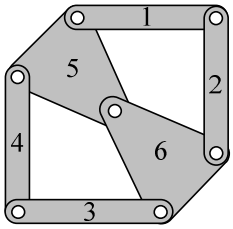


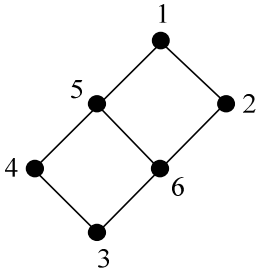


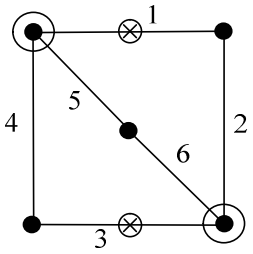
(a) We expand the ternary joints in two different ways using the links marked as ⊗

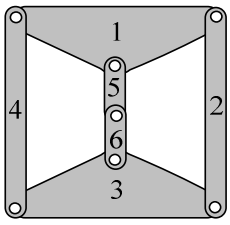
(i) and (ii) The systems and their corresponding interchange graphs are

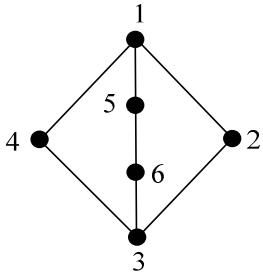












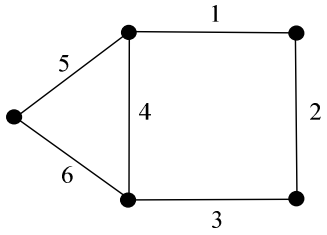
(b) (i) In both systems the number of links = 6 and the number of joints = 7

(ii) The mobility of both systems is $M = 3(n-1) - 2j = 3(6-1) - 2 \times 7 = 15 - 14 = 1$

The first mobility criterion has been used since once expanded the system only has binary joints and the first mobility criterion applies in such a situation.

(c) If we are to have a direct graph of a system which on expansion of its two ternary joints produces the same two kinematic systems as those in part (a) then the direct graph must have two vertices of degree 3 and three vertices of degree 2.
If this direct graph is not to be isomorphic to that given in the question then the two vertices of degree 3 must be incident with each other.

Such a direct graph is



Q19 (a) Since the parity check matrix is an $(n-k) \times n$ matrix, $n - k = 2$ and $n = 5$, thus

The length of code C = 5, and since k = 3, the rate of the code = $\frac{3}{5}$

(b) The generator matrix of the dual code C* is the parity check matrix of C.

Codewords of C* are thus 00000, 10110, 01001, 11111

(c) No, the codes C and C* are not equivalent because the dimension k of C is 3 as noted above while, since C* has $4 = 2^2$ codewords, the dimension of C* is 2.

(d) Since the parity check matrix H is in standard form the code is systematic.

$$H = \left[\begin{array}{ccc|cc} 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{array}\right]$$

(e) Using the result at the bottom of page 26 in Design of Codes we can derive a generator matrix for C from the parity check matrix in standard form above

$$G = \left[\begin{array}{ccc|cc} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{array}\right]$$

(f) (i) Adding a parity check digit in the first position we obtain a parity check for C'

$$H' = \left[\begin{array}{c|ccccc} 1 & 1 & 1 & 1 & 1 & 1 \\ \hline 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{array}\right]$$

(ii) The parameters of C' are

$n' = n + 1 = 6$ $k' = k = 3$ $\delta' = \delta = 2$ since δ is even

(iii) There is no change in the minimum distance when the overall parity check digit is appended since $\delta' = \delta = 2$. Thus the code C' can detect and correct the same number of errors as the code C. However, the code C' has an extra bit so the rate is $\frac{3}{6} = \frac{1}{2} < \frac{3}{5}$ so C' is less efficient than C and it would be better to use C rather than C'.

This essay is just one version of what could be produced; it is probably more detailed than would be expected and is included to provide an overview of the topic as well as an example.

Q20 A polyhedron is a three dimensional solid whose faces are polygons in edge to edge contact. In general the polygons can be of any size and shape and the solid so formed can be convex or not. A polygon is a plane figure whose edges are straight lines, it is called regular if all its edges are equal in length and consequently all of its internal angles are also equal. Of special interest are the convex polyhedra for which any line segment joining two points within or on the polyhedron itself lies within or on the polyhedron.

The convex polyhedra for which the faces are congruent regular polygons are called the regular polyhedra or Platonic solids. All vertices of such polyhedra have exactly the same arrangement of polygons around them and there are in fact just five examples of such polyhedra. They are the tetrahedron consisting of four equilateral triangles, the octahedron consisting of eight equilateral triangles, the cube consisting of six squares, the icosahedron consisting of twenty equilateral triangles and finally the dodecahedron consisting of twelve regular pentagons. The dual of a regular polyhedron is formed by placing a vertex at the centre of each face and joining these vertices by an edge whenever the corresponding faces are in edge to edge contact. In this way it is easily shown that the duals of Platonic solids are all Platonic solids. In fact the tetrahedron is self dual while the cube and octahedron are duals of each other, as are the icosahedron and dodecahedron.

A semi-regular polyhedron is one in which the polygonal faces, while they are all regular, are not congruent, however, each vertex has the same arrangement of polygons around it. There are an infinite number of semi-regular polyhedra consisting of two infinite sets plus a set of thirteen. The infinite sets are the prisms and antiprisms, these both have end faces as regular n-polygons but the prisms have n squares around the sides, while the antiprisms have offset end faces and 2n equilateral triangles around the sides. The thirteen individual semi-regular polyhedra are called the Archimedean solids, five have names which indicate the way they can be formed by truncation from the Platonic solids such as truncated tetrahedron, truncated cube etc. The cuboctahedron is a half way truncation between the cube and the octahedron and similarly the icosidodecahedron is a half way truncation between the icosahedron and the dodecahedron. Finally there are six which are derived from the cuboctahedron and icosidodecahedron either by inserting extra equilateral triangles producing snub versions or further truncation in two different ways. The five basic truncations, the cuboctahedron and icosidodecahedron and their snub versions and one of the further truncations of the cuboctahedron all have two types of faces. The other three have three types of faces.

Euler's polyhedron formula is a simple formula which relates the number of vertices v, edges e and faces f of any convex polyhedron, $v - e + f = 2$. In addition there is a simple result called the handshaking lemma for polyhedra which connects the degree of a face of a polyhedron to the number of edges. Since the degree of a face is defined to be the number of edges surrounding it, then it is obvious, since each edge is adjacent to two faces, that the sum of the face degrees must be equal to twice the number of edges. Euler's formula and the handshaking lemma have many important consequences not the least of which is the result given earlier that there are only five Platonic solids.

Given any convex polyhedron we can imagine removing a face and then opening out the polyhedron and flattening it onto the plane. In this way we can obtain a planar graph with the same number of vertices and edges as the original polyhedron. If we count the region outside the graph as a face then there will also be the same number of faces as in the original polyhedron and so Euler's formula will apply to planar graphs as well as to polyhedra.

There are many other types of polyhedra such as various types of pyramids which have different arrangements of polygons around different vertices or the stellated polyhedra which are non-convex.

Essay Outlines

Q14. Introduction – definition of an algorithm – efficiency.

Time complexity functions – order hierarchy.

Polynomial Time Algorithms – examples

Exponential Time Algorithms – examples.

P and NP Problems.

Conclusion – NP-complete problems

Q17. Introduction – use of the Hungarian algorithm with bipartite graphs.

The Transportation Problem.

The Assignment Problem.

The Labelling Procedure.

Partial graphs and the minimum cost nature of the algorithm.

Conclusion – modifications, Dummy Vertices, Transhipment and Bottlenecks.

Q20. Introduction – definition of polyhedron, polygon and convex

Regular convex polyhedra – Platonic Solids – Duals

Semi-regular polyhedra – Archimedean Solids – Prisms and Anti-prisms

Euler's Polyhedron Formula – Handshaking Lemma for Polyhedra

Planar graphs – Euler's formula – the link with polyhedra.

Conclusion – other polyhedra.