# Topics by Exam Board specifications - Discrete (aka Decision) [Further Maths only]

(9 pages; 30/7/19)

**OCR** ('Discrete')

D: material common to AS and AL

D*: material for 2nd year of AL only

**OCR (MEI)**

Modelling with Algorithms [A] [can be taken at either AS or AL]

**AQA** ('Discrete')

D: material common to AS and AL

D*: material for 2nd year of AL only

**EDX** ('Decision')

D1: material common to AS and AL

D1*: material for 2nd year of AL only

D2: material common to AS and AL

D2*: material for 2nd year of AL only

| | fmng reference (Y⇒ note exists) | OCR | OCR (MEI) | AQA | Edx |
|---|---|---|---|---|---|
| **Preliminaries** | | | | | |
| Problem-solving terminology | | D | | | |
| Sets | | D | | | |
| Pigeonhole principle | | D | | | |
| Permutations & combinations | Prob. & Stats | D | | | |
| Inclusion-exclusion principle | | D | | | |
| | | | | | |
| **Algorithms** | | | | | |
| Terminology | | D | A | | D1 |
| Practicalities | | D | | | |
| Tracing through algorithm | | D | A | | D1 |
| Order, efficiency & complexity of an algorithm | | D, D* | A | | |
| Heuristic algorithms | | | A | | |
| **Sorting algorithms** | | | | | |
| Bubble Sort | | D | | | D1 |
| Shuttle Sort | | D | | | |
| Quick Sort | | D* | A | | D1 |
| - counting comparisons | | | A | | |

| | | | | | |
|---|---|---|---|---|---|
| **(Bin-)packing algorithms** | Y | D, D* | A | | D1 |
| - counting comparisons | | | A | | |
| | | | | | |
| **Allocation problems** | | | | | D2 |
| Introduction | | D | | | |
| Cost matrix reduction | | | | | D2 |
| Hungarian algorithm | Y | | | | D2 |
| - dummy location | | | | | D2 |
| - incomplete data | | | | | D2 |
| - maximum profit | | | | | D2 |
| Formulation as a LP problem | | | | | D2* |
| | | | | | |
| **Critical Path analysis** | Y | | | | |
| Construct activity-on-arc networks | P1 | D | A | | D1 |
| Construct activity-on-node networks | P1 | | | D | |
| Creation of the Precedence table from the network | | | | | D1 |
| Perform forward & backward passes | P1 | D | A | D | D1 |

| | | | | | |
|---|---|---|---|---|---|
| Determine floats | P1 | D,D* | A | D | D1 |
| Use Cascade (aka Gantt) chart | P2 | D* | | D* | D1 |
| Resource & scheduling problems | | | A | D* | |
| Lower bound for number of workers needed | P2 | | | | D1 |
| Resource Levelling | P2 | | | D* | D1* |
| Construct Resource Histograms | P2 | | | D* | D1* |
| Construct scheduling diagram | P2 | D* | | | D1* |
| | | | | | |
| **Decision Analysis** | | | | | D2* |
| | | | | | |
| **Dynamic Programming** | | | | | D2* |
| Bellman's principle of optimality | | | | | D2* |
| Stage & State variables | | | | | D2* |
| Use of tabulation to solve max., min., minimax, maximin problems | | | | | D2* |
| | | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **Dijkstra's algorithm** | Y | D | A | | D1 |
| | | | | | |
| **Floyd's algorithm** | Y | | | | D1* |
| | | | | | |
| **Game Theory** (zero sum) | | D | | D | D2 |
| Converting to zero-sum form | | D | | | |
| Reducing a matrix using a dominance argument | | D | | D | D2* |
| Identifying play-safe strategies and stable solutions | | D | | D | D2 |
| Value of game | | | | D | |
| Nash equilibrium solution | | D* | | | |
| Determining optimal mixed strategy | | D | | D | D2* |
| Reformulating as LP problem | | D* | | D* | D2* |
| | | | | | |
| **Graphs & Networks** | | | | | |
| Terminology | Y | D | A | D | |
| Complete graphs | | D | A | D | D1 |
| Bipartite graphs | Max. matching algorithm | D,D* | A | D | |

| | | | | | |
|---|---|---|---|---|---|
| Eulerian graphs | | D | | D | D1 |
| Hamiltonian graphs | | D* | | D | D1* |
| Isomorphism | | D | | D* | D1 |
| Adjacency matrix (graphs) | | D | | D | |
| Weighted matrix (networks) | | D | | | D1 |
| Complement of graph | | | | D | |
| **Planar Graphs** | | D | | D | D1 |
| Euler's formula | | D* | | D | |
| Subgraph | | D* | | D | |
| Subdivision | | D* | | D | |
| contraction | | D* | | | |
| thickness | | D* | | | |
| Kuratowski's thm | | D* | | D* | |
| Planarity algorithm | | | | | D1 |
| | | | | | |
| **Linear Programming** | Y | | | | |
| Formulate constrained optimisation problems (in standard form) | | D* | A | D | D1 |
| Reformulating non-standard forms (if variables can | | | A | | |

| | | | | | |
|---|---|---|---|---|---|
| be negative or objective function is to be minimised) | | | | | |
| Formulate network problems as LP: shortest path, critical path, network flows, matching, allocation, transportation | Y | | A | | |
| Graphical solution | | D | A | D | D1 |
| Integer solution | | D | A | | D1 |
| Branch & Bound method | | D* | | | |
| 3D LP | | | A | | |
| Reduction of 3D LP to 2D | | | A | | |
| **Simplex method** | Y | D* | A | D* | D1* |
| Formulate LP problem in augmented (or 'slack') form | | D* | A | D* | D1* |
| Minimisation | P2 | | | D* | D1* |
| Interpretation | | D* | A | D* | |
| 2 Stage Simplex | P2 | | A | | D1* |
| Big M method | P2 | | | | D1* |
| Equality constraints | P2 | | A | | |
| Using LP program | | | A | | |

| | | | | | |
|---|---|---|---|---|---|
| **Minimum Connector problem** | Y | | | D | |
| Prim's algorithm - graphical | | D | A | | D1 |
| Prim's algorithm - tabular | | D | A | | D1 |
| Kruskal's algorithm | | D | A | | D1 |
| | | | | | |
| **Network Flows** | | | A | | |
| Max. flow - min. cut thm | | | A | D | D2 |
| Supersources & supersinks | | | | D | D2* |
| Augmenting flows to find max. flow (labelling procedure) | | | | D* | D2 |
| Arcs with upper or lower capacities | | | | D* | D2* |
| Nodes with restricted capacity | | | | D* | D2* |
| | | | | | |
| **Route Inspection problem** | Y | D* | | D | D1 |
| | | | | | |
| **Transportation problems** | | | | | D2* |

| | | | | | |
|---|---|---|---|---|---|
| (North-West corner method) | | | | | |
| Introduction | | D | | | |
| Stepping stone method | | | | | D2* |
| Dummy locations | | | | | D2* |
| Degeneracy | | | | | D2* |
| Formulation as a LP problem | | | | | D2* |
| | | | | | |
| **Travelling Salesman problem** | Y | | | | |
| Practical & classical problems | | | | | D1* |
| Creating network of shortest distances | | | | | D1* |
| Nearest neighbour algorithm | | D* | | D | D1* |
| Lower bound algorithm | | D* | | D | D1* |
| Use of spanning tree to find upper bound | | | | | D1* |
| Use of shortcuts to improve upper bound | | | | | D1* |