Homework #15

Knapsack without repetition

Use dynamic programming to fill a knapsack without repetition having a maximum weight capacity of 10 units with a load of maximum value from the following objects:

Object	Weight	Value
A	1	1
B	2	7
C	5	11
D	6	21
E	7	31

Maximum value at weight

w,v	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	1	1	1	1	1	1	1	1	1
2	0	1	7	8	8	8	8	8	8	8	8
3	0	1	7	8	8	11	12	18	19	19	19
4	0	1	7	8	8	11	21	22	28	29	29
5	0	1	7	8	8	11	21	31	32	38	39

The maximum value is 39 and that knapsack configuration comprises {E, B, A}.

Matrix multiplication

M1: 20x5, M2: 5x10, M3: 10x12, M4: 12x6, M5: 6x25

The first thing we do is calculate the cost of multiplying any two adjacent matrices.

M1*M2	M1
M2	1000

M2*M3	M2
M3	600

M3*M4	M3
M4	720

M4*M5	M4
M5	1800

We now calculate the minimum cost of multiplying any three consecutive matrices.

For example, to calculate the cost of M1*M2*M3, we consider the cost of M1*(M2*M3) and (M1*M2)*M3. The total cost includes the cost of M2*M3 and M1*M2 respectively.

M1*M2*M3	M1	M1*M2
M2*M3	1200+600=1800	
M3		2400+1000=3400

M2*M3*M4	M2	M2*M3
M3*M4	300+720=1020	
M4		360+600=960

M3*M4*M5	M3	M3*M4
M4*M5	3000+1800=4800	
M5		1500+720=2220

We do the same thing for any four consecutive matrices, referring to the tables above as necessary. When determining how much to add from a table above, we choose the minimum. In this example, to obtain M1*M2*M3*M4, it is cheapest to calculate M2*M3*M4 and then multiply M1 by that result.

M1*M2*M3*M4	M1	M1*M2	M1*M2*M3
M2*M3*M4	600+960=1560		
M3*M4		1200+1000+720=2920	
M4			1440+1800=3240

M2*M3*M4*M5	M2	M2*M3	M2*M3*M4
M3*M4*M5	1250+2220=3470		
M4*M5		1500+600+1800=3900	
M5			750+960=1710

Finally, we can calculate the minimum cost to find the product of all the matrices.

M1*M2*M3*M4*M5	M1	M1*M2	M1*M2*M3	M1*M2*M3*M4
M2*M3*M4*M5	2500+1710=4210			
M3*M4*M5		4000+1000+2220=7220		
M4*M5			3600+1800+1800=7200	
M5				3000+1560=4560

All Paths:

The Initial table of 1-hop distances from the graph, Table$(i, j, 0)$. We only need to consider the upper triangle, because the graph is undirected, so distances are symmetric.

	a	b	c	d	e
a	0	3	5	-	9
b		0	-	-	-
c			0	2	4
d				0	1
e					0

Table$(i, j, 1)$, being the shortest distances considering hopping through vertex a:

	a	b	c	d	e
a	0	3	5	-	9
b		0	8	-	12
c			0	2	4
d				0	1
e					0

Table$(i, j, 2)$, being the shortest distances considering hopping through vertex b:

	a	b	c	d	e
a	0	3	5	-	9
b		0	8	-	12
c			0	2	4
d				0	1
e					0

Table$(i, j, 3)$, being the shortest distances considering hopping through vertex c:

	a	b	c	d	e
a	0	3	5	7	9
b		0	8	10	12
c			0	2	4
d				0	1
e					0

Table$(i, j, 4)$, being the shortest distances considering hopping through vertex d:

	a	b	c	d	e
a	0	3	5	7	8
b		0	8	10	11
c			0	2	3
d				0	1
e					0

Table$(i, j, 5)$, being the shortest distances considering hopping through vertex e:

	a	b	c	d	e
a	0	3	5	7	8
b		0	8	10	11
c			0	2	3
d				0	1
e					0