1 Exercise 1

1.

D^0, D^1, and D^2 are as follows:

$$\begin{pmatrix} 0 & 80 & +\infty & 10 \\ +\infty & 0 & 10 & +\infty \\ 40 & +\infty & 0 & +\infty \\ +\infty & 20 & 90 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 80 & +\infty & 10 \\ +\infty & 0 & 10 & +\infty \\ 40 & \mathbf{120} & 0 & \mathbf{50} \\ +\infty & 20 & 90 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 80 & \mathbf{90} & 10 \\ +\infty & 0 & 10 & +\infty \\ 40 & 120 & 0 & 50 \\ +\infty & 20 & \mathbf{30} & 0 \end{pmatrix}$$

2.1 $\Theta(n)$ **2.2** $\Theta(1)$ **2.3** $\Theta(1)$

3.1 6, F. Visited event points: ABCDEF
3.2 8, H. Visited event points: BFCDH

4.1 4
4.2 4
4.3 3

Range tree: $\Theta(\lg^2 n + k)$
kd tree: $\Theta(\sqrt{n} + k)$
2 BSTs: $\Theta(\lg n + k_x + k_y)$
workload 1: $k = \frac{n}{4}$, workload 2: $k = \frac{n}{100}$, workload 3: k is a constant.
5
5.1 12000 or 20000
5.2 4000
5.3 4000

1 TB file means $\frac{1TB}{1GB}$=1000 disk pages.

5.1 50 GB memory means 50 disk pages. 2-way merge sort. $\lceil log_2(1000) \rceil = 10$. 10 iterations. each iteration reads in 1000 disk and writes 1000 disk. in total, each iteration 2000 I/O operations. So 2000*10=20000.

Or: Depends on how do we define a 2-way merge sort. Initial main memory sortings (resulting in 20 runs) + ceil ($\log_2$ 20) = 1 + 5 = 6 iterations. each iteration 2000 I/O of the file so, 12000.

5.2 50 GB memory means 50 disk pages.multi-way merge sort. $\lceil log_{49}(1000) \rceil$ = 2. 2 iterations. each iteration reads in 1000 disk and writes 1000 disk. in total, each iteration 2000 I/O operations. So 2000*2=4000.

Or: Initial main memory sortings (resulting in 20 runs) + ceil ($\log_{49}$ 20) = 1 + 1 = 2 iterations. each iteration 2000 I/O of the file so, 4000.

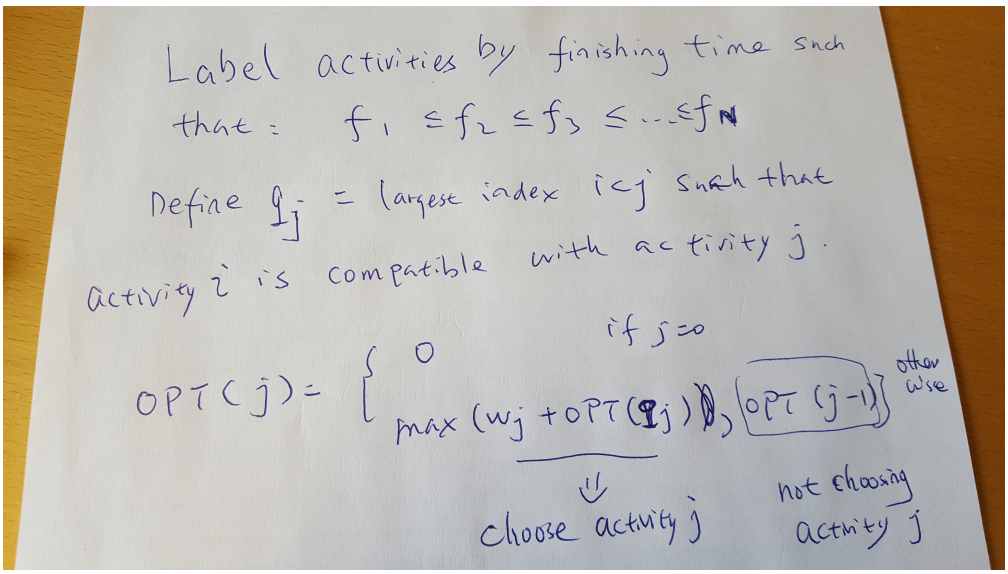
5.3 50 GB memory means 500 disk pages. multi-way merge sort. $\lceil log_{499}(1000) \rceil$ = 2. 2 iterations. each iteration reads in 1000 disk and writes 1000 disk. in total, each iteration 2000 I/O operations. So 2000*2=4000.

Or: Initial main memory sortings (resulting in 20 runs) + ceil ($\log_{499}$ 20) = 1 + 1 = 2 iterations. each iteration 2000 I/O of the file so, 4000.

6 3

2 Exercise 2

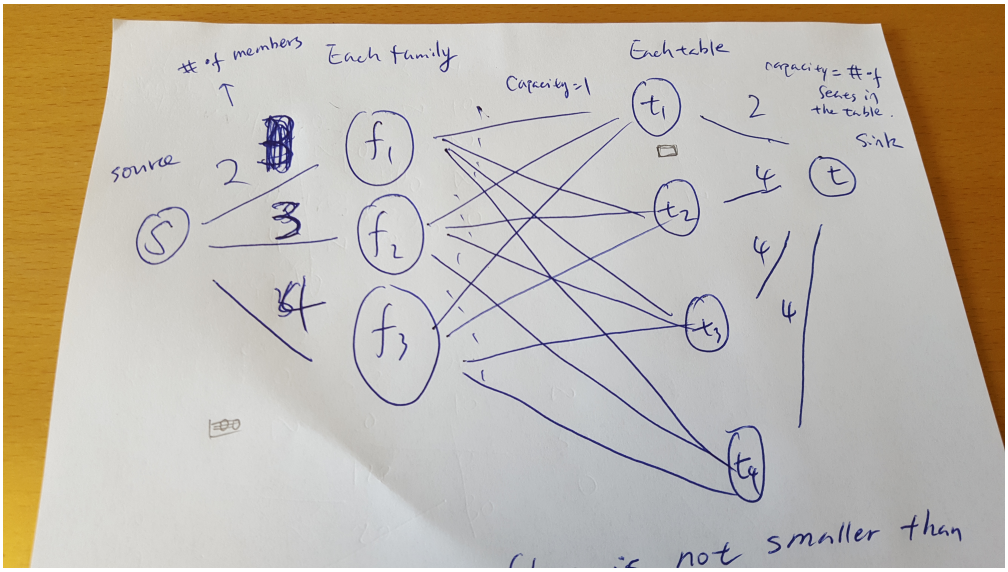
1



2 We have a source and a sink. We have X vertices, one for each family. We have Y more vertices, one for each table.

Connect the source to each family vertex with an edge, where the capacity is the number of the family members.

Connect the sink to each table vertex with an edge, where the capacity is the number of tables.



Connect each family vertex with each table vertex with an edge, where the capacity is 1.

If the maximum flow is not smaller than the total number of family members, it is possible to arrange the seats according to the rule. Otherwise, it is impossible.

For an edge that connect family vertex f_i and table vertex t_j: 1 means that a family member from f_i should sit table t_j.