Ass 4

1. There are N computers in a network, labelled {1, 2, 3, . . . , N}. There are M one-directional links which connect pairs of computers. Computer 1 is trying to send a virus to computer N. This can happen as long as there is a path of links from computer 1 to computer N. To prevent this, you've decided to remove some of the links from the network so that the two computers are no longer connected. For each link, you've calculated the cost of removing it. What is the minimum total cost to disconnect the computers as required, and which edges should be removed to achieve this minimum cost? (25 pts)

   Solution:

   1. source: computer 1
   2. Sink: computer N
   3. links are Edges capacity ci = cost of removal
   4. 最大流算法找min cut，移除min cut就能保证完全阻止病毒到computerN，同时也是最小花费。

   Time Complexity O(V^3)

2. You have n warehouses and n shops. At each warehouse, a truck is loaded with enough goods to supply one shop. There are m roads, each going from a warehouse to a shop, and driving along the ith road takes d_i hours, where d_i is an integer. Design a polynomial time algorithm to send the trucks to the shops, minimising the time until all shops are supplied. (25 pts)

   Hint: Combine a binary search with a max flow. By sorting you can assume that d_i form an increasing sequence. Fix i and consider only roads which take $\leq d_i$ hours to travel from a warehouse to the corresponding shop and use max flow to see if they are enough to obtain a matching of warehouses with shops which is of size n. Use a binary search on i to find the smallest d_i which meets the requirements.

requirements

Solution:

sorting （按照hint改一下）- 对所有的路di排序。

对某个i，我们建一个bipartite graph

1. warehouses and shops are vertices 左侧warehouses，右侧shops
2. 如果某个warehouse和shop有路，就连一个edge从warehouse指向shop
3. super Source和warehouse相连，ci都是正无穷↑（大家问一下到底一个warehouse是不是只有一辆车？如果只有一辆车，那ci就是1，否则就是正无穷）
4. shops 和super Sink相连，ci都是1
5. 根据对应的di，如果某条warehouse和shop之间的一条路的dj<di, 那么cj=1，否则cj = 0
6. 跑最大流，如果结果=n，那就说明当前选择的i可以，继续进行binary search重复步骤1-5

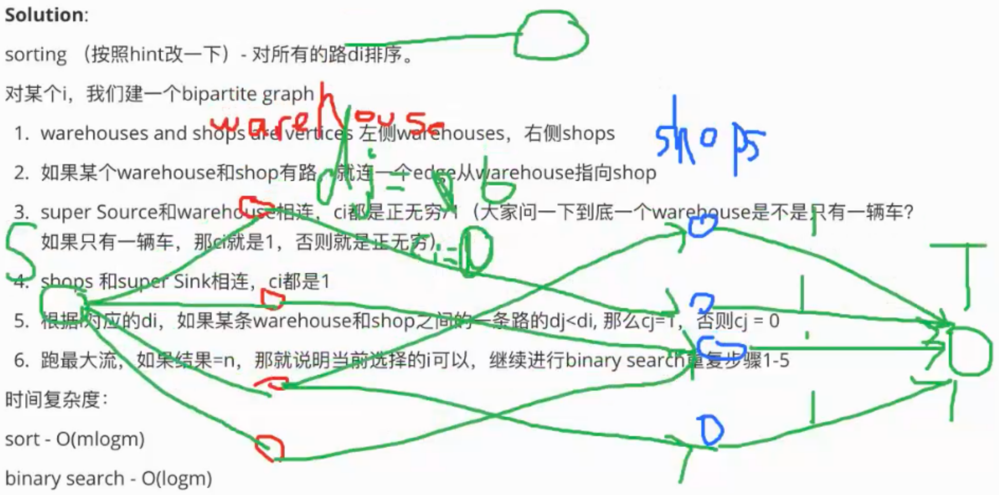
时间复杂度：

sort - O(mlogm)

binary search - O(logm)

Max-flow algorithm: Preflow-Push algorithm runs in time O(|2n+2| ^3).= O(n^3)

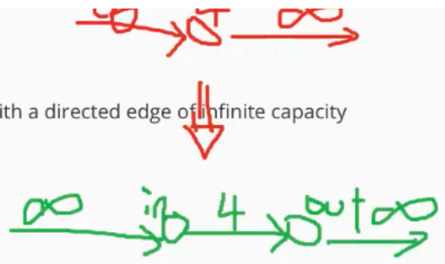
O(m logm) + O(n^3 log m)



3. A large group of children has assembled to play a modified version of hopscotch. The squares appear in a line, numbered from 0 to n, and a child is successful if they start at square 0 and make a sequence of jumps to reach square n. However, each child can only jump at most k < n squares at a time, i.e., from square i they can reach squares i + 1, i + 2, . . . , i + k but not i + k + 1, and a child cannot clear the entire game in one jump. An additional rule of the game specifies an array A[1, . . . , n − 1], where A[i] is the maximum number of children who can jump on square i. Assuming the children co-operate, what is the largest number of children who can successfully complete the game?(25 pts)

Hint: Connect every square i with squares i+1, . . . , i+k with a directed edge of infinite capacity. Now limit the capacity of each square appropriately and use max flow.

Solution: slides P25



1. 顶点是n+1个squares
2. every square i with squares i+1, . . . , i+k with a directed edge of infinite capacity
3. source是 square 0 capacity 是inf
4. sink 是square n capacity 是inf
5. 顶点的capacity是array A[i]
6. 把每个vertice分成两个，Vin Edge Vout， Edge上capacity是原来顶点的capacity
7. 跑最大流算法，得到的结果就是最多能到达的小孩子。复杂度：O((2n+1)^3)

4. You are given an n × n chess board with k white bishops on the board at the given cells (ai,bi), (1 ≤ ai,bi ≤ n, 1 ≤ i ≤ k). You have to determine the largest number of black rooks you can place on the board so that no two rooks are in the same row or in the same column or are under the attack of any of the k bishops (recall that bishops go diagonally).(20 pts)

Solution：

建二分图

左边n个vertices是ri - row

右边n个vertices是cj - columns

ri和cj的连线就表示一个可能的坐标（i，j）

设置super source s和 super sink t。 s到r流量1，因为一行只能一个black rooks c到t流量1，因为一列只能一个black rooks

中间连边，如果格子(i,j)不在bishops的攻击范围里，就连上边。如果这个格子在主教范围就无边

跑最大流 - 结果就是能放的最多的black roots - O（n^3）