COMP3121/9101 Term 2, 2021
Assignment 4 Solutions

Due date Friday, August 6, 2021 at 8 PM sharp.

You have **four problems**, marked out of a total of 100 marks.

1. There are N computers in a network, labelled $\{1, 2, 3, \ldots, N\}$. 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: Construct a flow network with computers as vertices and with links between computers as edges of capacity set to the price of disconnecting such a link. The source of the network is set to vertex 1 and the sink at vertex N. Find a max flow in such a network and the corresponding min cut. Disconnect the edges which cross the cut in the direction of the partition containing 1 to the partition containing N.

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.

   Solution: First sort time travel distances d_i in an increasing order; thus, we can assume that $d_{i+1} \geq d_i$. Consider a value d_i for some i and construct a bipartite graph G_i with warehouses w_j as the left side of the partition and with shops s_j $(1 \leq j \leq n)$. Connect all warehouses with all shops which are within travel distance times d_i. Use max flow to see if such bipartite graph has a perfect maximum matching of size n. Use a binary search to find the smallest i such that graph G_i has a matching of size n, i.e., a matching in which every warehouse has been matched with a shop, so that different warehouses are assigned different shops.

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 $i+1, i+2, \ldots, i+k$ 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 $A[1, \ldots, n-1]$, 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 $i+1, \ldots, i+k$ with a directed edge of infinite capacity. Now limit the capacity of each square appropriately and use max flow.

   Solution: Construct a flow network with vertices $0, 1, 2, \ldots, n$ such that for all $i > 0$ vertex i corresponds to hopscotch square i. Connect the additional entry vertex 0 with vertices corresponding to squares 1 to k and every vertex corresponding to square i with squares $i+1, \ldots, i+k$, all with edges of infinite capacity. Set the capacity of each vertex $0 < i < n$ to $A[i]$. Set 0 vertex as a source and vertex n as a sink and find max flow in such a network. The number of kids which can reach square n is equal to the max flow in such a network,

4. Use max flow algorithm to solve the following problem. You are given an $n \times n$ chess board with k white bishops on the board at the given cells (a_i, b_i), $(1 \leq a_i, b_i \leq n,\ 1 \leq i \leq k)$. You have to determine the largest number of black rooks which 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).(25 pts)

   Solution: Construct a bipartite graph with n left side vertices representing rows of the board and n right side vertices representing columns. Connect with directed edges of capacity one every left side vertex i with all right side vertices j which satisfy the property that the cell (i, j) of the board is not under attack from one of the bishops. Add a super source and connect it with all left vertices with edges of capacity one; add a super sink and connect every right side vertex with such super sink with edges of capacity one. Find max flow; if the flow equals n then the problem has a solution. The rooks should be placed on cells (i, j) which correspond to edges with flow in them.