CSE548 Fall 2019 Analysis of Algorithms Homework4

Tianao Wang 112819772

November 24, 2019

1. Let all switches be a set of A. Let all fixtures be a set of B. Then connect nodes in A and B which can be directly linked in graph without integrating with walls. After that, set a node s connected to all switches and a node t connected to all fixtures. Also let all edges weight 1. Then we can use the Ford-Fulkerson algorithm to this problem. If the maximum flow is n, the plan is ergonomic. If not, the plan is unergonomic.
   Proof: If the input of t and the output of s are n, this means each switch has connected to one fixture because each edge weights 1. This is a perfect match problem.
   Running time: $O(n^3)$.

2. False.
   We can construct a graph N*N grid graph with a source s and a sink t. In the graph every node has an edge connecting it to every node either horizontally or vertically adjacent to it except the source and sink. And we find a augmenting path from s to t like $s->u_n->v_n->u_{n-1}->v_{n-1}->...->u_1->v_1->t$. In this condition, the flow is 1 and the maximum flow is n. We can not find a constant b to bound n/1. So the argument is wrong.

3. We can construct a graph G'. Every vertex v in G has two nodes in G' which are v_{in} and v_{out}. There is an edge from v_{in} to v_{out} of capacity of c_v. If there is a edge in G from vertex u to v, then we add a edge called (u_{out}, v_{in}) of infinite capacity. The amount of input to v can not beyond c_v and every edge in G' represent an edge in G. Then we can run Ford-Fulkerson algorithm in G' to find the maximum flow.
   The MaxFlow MinCut Theorem in this problem is that the maximum s-t node-capacitied flow in G is equal to the minimum capacity of an s-t node-cut in G. We can use the prove of maxflow-mincut theorem in G' to prove this theorem.

4. (a) Let all n people be a set of A and let all n nights be a set of B. Add an edge (u,v) when u is able to cook in v night. We call this graph G.
   After adding all edges, we can construct G'. We can add a source s connected to all people and sink t connected to all nights. Also the weight of each edge is 1. Then we can find the maximum flow in this graph G'. If the maximum flow is n. This graph G is perfect matching and there exsits a time schedule.
   (b) Alanis's schedule has find a flow which values n-1 in G'. We can try to find an augmenting path is the residual graph for G'. If there is another augmenting path, we can increase the flow value to n and get a perfect matching. We can output this result. The running time is $O(n^2)$. If not, it means that Alanis's schedule cannot be improved.

5. We can construct a graph G. Let all users be a set of A and let all advertisers be a set of B. Set a source s which connected to every users and each edge weights 1. Set a sink t which connected to every advertisers and each edge weight r_i(each advertiser's requirement). Add

all possible edges from users to advertiser which means user may see some advertisement and these edges weight 1.

Then we can run a maximum flow algorithm to find the maximum flow. If the maximum flow is $\sum r_i$, we can show each user one ad which is calculated in this algorithm. If not, it means that we cannot find a satisfied plan to show advertisement.

6. First we can calculate a minimum cut in graph G called C and record the flow value of the cut C. Then we can increase each edge in C by 1 and recalculate the minimum cut. If the value of new minimum cut equals the original one, we can say the new cut is also the minimum cut in graph G and the minimum cut is not unique. In other words, if there is another minimum cut, some edges in the original C will not be in the other minimum cut. We just need to try the all edges in C, so this algorithm is polynomial time.