**Algorithms and Complexity**

**2017**

**Mästarprov 2: Complexity**

1. Super connectors

CLIQUE problem is the problem of finding subsets of vertices, all adjacent to each other in a graph. Our problem is to find a group of super connectors. If we let every connector to be a vertex and let edge between vertices represent that there exists a connection between them. Then our super connector problem can be reduced from the CLIQUE problem of finding a CLIQUE of size k. the CLIQUE problem is a NP-complete problem so our problem is also NP-complete problem.

* 1. We have a fourpartite matching problem of members and a group of feasible sets . We try to form a tripartite matching problem of members for all combinations of men, women, dogs with cats. The number of member is then at most . With the original feasible sets, we form the following feasible sets where these is the same in all three members in the same feasible set. The size of the new sets is the same as the original one. Then we will have a tripartite matching problem that can be reduced to the original problem. Since the tripartite matching problem is NP-complete so the fourpartite matching problem is also NP-complete.
  2. Here we assume that . Let’s assume that the tripartite matching problem can be reduced from the bipartite matching problem. Since the tripartite matching problem is a NP-complete problem and all NP-complete problem can be reduced to each other so all NP-complete problem can be reduced from the bipartite matching problem. Since the bipartite matching problem is a P problem so all NP-complete problem can then be solved in P time. And since NP-complete problem is NP-hard and all NP problem can be reduced to any NP-hard problem, so all NP problem can be reduced to NP-complete problems. So, all NP problem will then be able to be solve in P time which mean which is a contradiction. So, under the assumption , the tripartite matching problem cannot be reduced from the bipartite matching problem.

1. To check if A has a winning strategy is the same as to see if B has a winning strategy.
2. The algorithm should first iterate from to 1 using to find the maximum which gives a in return which is the length of the longest path. This process use at most times.

Then we choose a vertex and remove it and all the edges connect to it and run the to the resulting graph with the same . If the answer is still then this vertex is not in the longest path and we can remove it and go to the next vertex that is connected to this vertex. If the answer is then this vertex is in the longest path and we store it in a result list and go to the next vertex that is connected to this vertex doing the same check with reduced by 1. The process ends when . Then the vertices in the result list forms the longest path. Since all edges can only be removed once so this process use at most times.

So, the total complexity is .