1 Problem Reduction

If each vertex of a graph G carries a positive weight, then the weight of a set of S of vertices is defined as the sum of weights of all the vertices in S and the optimal cover problem requires finding a cover of the smallest weight. Show that this problem can be turned into the minimum cut problem whenever G is bipartite.

Answer: If G is bipartite and each vertex of this graph G has a positive weight, we can reduce to a network-flow problem and find the minimum cut , this minimum cut is the optimal cover for the optimal problem. This reduction can be described as follows: we create another graph G’ which has only one source and one sink vertex s and t, we connect s to one set of vertexes in bipartite G and connect another set of vertexes to t with the corresponding weight for the cost of edge. The cost of middle edges in G can be infinite. The minimum cut for this problem can be the optimal cover.

For example:

3

33

13

3

43

t3

s3

1113

3

3

53

The red line is the min cut, so there are two sets A and B(A is up from red line and B is V-A) and also we have two sets C and D in G(C is the set of left points in the G and D is the set of right points in the G). The cover problem’s answer is the points which are and . First we will prove the min cut can be the answer to the cover problem and next we will prove it is the optimal answer.

First, the min cut should not cut the edge which the end points are in and because this edge is infinite. We can easily get the points and the points which cover all the edges.

Second, the min cut is to find the min cut for the graph G’, in this problem, it is to find the points which can cover all edges in min to the weights of these points. The min cut cuts the edge which ‘s weight isn’t infinite and it select the cuts which ‘s weight is as small as possible so it can be easily reducing to the cover problem which select the min weights of points.

2 Problem Reduction

Let M be an n\*n matrix with each entry equal to either 0 or 1. Let mij denote the entry in row i and column j. A diagonal entry is one of the form mii for some i. Swapping rows i and j of the matrix M denotes the following action: we swap the values mik and mjk for k = 1,2,…,n. Swapping two columns is defined analogously. We say that M is rearrangeable if it is possible to swap some of the pairs of rows and some of the pairs of columns (in any sequence) so that, after all the swapping, all the diagonal entries of M are equal to 1.

(a) Give an example of a matrix M that is not rearrangeable, but for which at least one entry in each row and each column is equal to 1.

(b) Give a polynomial-time algorithm that determines whether a matrix M with 0-1 entries is rearrangeable.

Answer:

1. We reduce this problem to the network-flow problem.

3 Unique Cut

Let G = (V,E) be a directed graph, with source s V , sink t V , and nonnegative edge capacities {}. Give a polynomial-time algorithm to decide whether G has a unique minimum s-t cut.

Answer: we get the max-flow or min-cut in the G and we have the . We can get the set A of nodes which can get from s and the set B of nodes which can get from t. If , then we can have unique cut ; otherwise we doesn’t have unique cut. Now we will prove this. If , then we have point , this point x means in , s cannot arrive at x and x cannot arrive at t. And it means in G, and , and , so we have , also this min-cut can cut all edges or cut all edges . So the min-cut cannot be unique.

5 Dogs and kennels

On a grid map there are n little dogs and n kennels. In each unit time, every little dog can move one unit step, either horizontally, or vertically, to an adjacent point. For each little dog, you need to pay a $1 travel fee for every step it moves, until it enters a kennel. The task is complicated with the restriction that each kennel can accommodate only one little dog.

Give a polynomial-time algorithm to compute the minimum amount of money you need to pay in order to send these n little dogs into those n different kennels.

Answer: we get the bipartite graph from this problem. For example , we have two dogs and two kennels, dog A arrive kennel A and kennel B in 1 step and 2 step while dog B arrive kennel A and kennel B in 3 step and 100 step. So the bipartite graph(add s and t point to create a network-flow) as follows:

1,1

1,3

2,2

1,2

3,3

2,100

100,100

3,100

We have s to dogs for weight which is the interval cost of min and max(min is this dog arrive kennels’ min and max is this dog arrive kennels’ max) the kennels to t is the same. The dogs to kennels’ s edges can be min and max(min is same as max for one to one match). So we get the max-flow or min-cut can be the answer. We prove the s to dogs’ s flow must be min or max or some value equals to the value this dog’s steps to any kennels. Because this e to dog x and but . And this is the same to the kennels to t. So it will make sure that one dog only to one kennel and this is the perfect matching for max steps. But we should get min steps so we can get the negative value.