ECS 222A: Assignment #6

Due on Thursday,March 5, 2015

Daniel Gusfield TR 4:40pm-6:00pm

Wenhao Wu

Contents

Problem 1

We saw a randomized algorithm that tries to find a global Min cut in an undirected, unweighted graph G with m edges. Now suppose we want to find a cut that has a *large* number of edges, i.e., a partition of the nodes of G into two sets S and T so that the number of edges that have one node in S and one node in T is large. Denote the maximum possible number as $Max(G)$. The problem of finding $Max(G)$ is NP-hard, so we would like a randomized algorithm that finds an S, T cut where the *expected* number of edges that cross the cut is a large fraction of $Max(G)$.

Here is a particularly brainless algorithm that does it. For each vertex, flip a coin: if the coin comes up heads, put the vertex in S, otherwise, put it in T. Assume that the coin is fair, i.e., the probability of heads is 1/2.

Problem 1(a)

Using this randomized algorithm, what is the expected number of edges that have one node in S and one in T? Explain.

Answer:

Problem 1(b)

Prove that in any undirected graph G, there is a cut that contains at least half of the edges in G.

Answer:

Problem 1(c)

Often in the analysis of social media, people build graphs representing who knows or likes (or hates or dates) whomever else. Then they analyze the graphs for particular features, such as large cliques, large independent sets, small cuts, nodes with high degree, the number of nodes of degree 1, number of triangles, etc. and they ascribe a "meaning" to each of these features. For example, a node with high degree is a "hub" or "kingpin"; and a large clique represents a "socially cohesive unit"; and a small cut is a "bottleneck", etc. This same approach to studying interactions is used in a huge variety of other systems. For example, graphs (call them "biological networks") are also to represent interactions between molecules, or between animals in a biological system, and biological, behavioural or chemical meanings are ascribed to features in these graphs. In general, such networks are called "interaction networks".

Now consider a "large cut" as a feature of an interaction network. Make up (use your imagination) as many meanings you can for a large cut in a social or biological network, or any other interaction network you can describe.

Given the fact stated in problem 1b, how large must a cut be before it could plausibly say anything meaningful about the interactions represented in an interaction graph? If you were a "network analyst" trying to find meaningful information from an interaction network, and you didn't know the fact stated in problem 1b, would that be a problem?

Answer:

Problem 2

Extend the analysis done for 3-SAT (in class and also in Section 13.4) to 4-SAT, i.e., the assumption that every clause has 4 literals. Then generalize to t-SAT for any fixed integer t. That is, generalize statments 13.14, 13.15, and 13.16, and justify your answers.

Problem 2 continued on next page...

Answer:

Problem 3

Do problem 7 in chapter 13 of the book. How does the answer to this problem relate to the answer of problem 2?

Answer:

Problem 4

Assume that SAT is NP-complete. Define twice-SAT as the problem of determing whether a given Boolean formula can be satisfied in at least two *different* ways. Two ways to satisfy a Boolean formula are different if at least one variable is set differently (i.e., true in one and false in the other).
Show how to reduce the SAT problem to the twice-SAT problem in polynomial time.
Assuming SAT is NP-complete, show that twice-SAT is NP complete.

Answer:

Problem 5

Approximation Algorithm for Node Cover
Recall the node cover problem:
Let G be an undirected graph with each node i given weight $w(i) > 0$. A set of nodes S is a *node cover* of G if every edge of G is incident to at least one node of S. The *weight* of a node cover S is the summation of the weights, denoted $w(S)$, of the nodes in S; the weighted node cover problem is to select a node cover with minimum weight.
The node cover problem (even when all weights are one) is known to be NP-hard, and hence we do not expect to find a polynomial-time (in terms of worst case) algorithm that is always correct. Therefore, we relax somewhat the insistence that the method be both correct and efficient for all problem instances. There are many types of relaxations that have been developed for NP-hard problems. The most common is the constant-factor, polynomial-time approximation algorithm.
For a graph G with node weights, let $S(G)$ denote the minimum weight node cover. Let A be a polynomial time algorithm that always finds a node cover, but one that is not necessarily minimum; let $S(G)$ denote the node cover of G that A finds. Then A is called a *constant-error polynomial-time approximation algorithm* (or approximation algorithm for short) if for any graph G, $S(G)/S(G) \leq c$ for some fixed constant c.
For the node cover problem we will give an approximation algorithm, based on network flow, with $c = 2$. First, recall that the node cover problem has a nice solution when the graph G is bipartite. This was done on a previous homework. You may take it as a black box at this point.
The Approximation Algorithm for General Graphs
Given G (not necessarilly bipartite), create bipartite graph $B = (N, N', E)$ as follows: for each node i in G, create two nodes i and i', placing i on the N side, and i' on the N' side of B; give both of these nodes the weight $w(i)$ of the original node i in G. If (i, j) is an edge in G, create an edge in B from i to j' and one from j to i'. Now find a minimum cost node cover $S(B)$ of graph B. From $S(B)$, create a node cover $S(G)$ in G as follows: for any node i in G, if either i or i' is in $S(B)$, then put i in $S(G)$.

Problem 5 continued on next page...

Problem 5(a)

It is easy to find examples where $S(G)$ is not a minimum node cover of G, and where $S(G)/S(G) = 2$. Do it.

Answer:

Problem 5(b)

However, no worse error ever happens.

Theorem 5.1 $S(G)/S(G) \leq 2$ *for any* G *and any choice of node weights for* G*.*

Prove the theorem.

Answer: