CS466/666, Fall 2009: Assignment 4

Out: November 2, Due: November 18, 5pm

1. **Reduction:** The **NAE-3SAT** problem is defined as follows: "Given N boolean variables, and M clauses that consist of 3 literals each, does there exist an assignment of TRUE and FALSE to the variables such that for each clause, the three literals do not all have the same value?" Thus, NAE-3SAT is almost the same as 3SAT, except that it is not allowed that all three literals in a clause are TRUE.

   The **Max-Cut** problem is defined as follows: "Given a graph $G = (V, E)$ and a number k, does there exists a set $C \subset V$ such that at least k edges have one endpoint in C and the other endpoint not in C?"

   Show that NAE-3SAT reduces to MaxCut.

2. **Maximum matching:** Recall that the *Maximum matching* problem asks, given a graph $G = (V, E)$, to find a largest possible matching, i.e., a set M of edges such that no two edges in M have a common endpoint. Finding a maximum matching is polynomial. But the algorithm to find it is fairly complicated and not linear. [1] Therefore it still makes sense to apply some of the techniques for NP-hard problems, especially if they lead to linear-time algorithms.

   (a) Formulate Maximum Matching as an integer program. Clearly state the intended meaning of variables and explain your constraints.

   (b) Give a linear-time algorithm that computes a $\frac{1}{2}$-approximation for maximum matching.

   (c) Let G be a graph with maximum degree 3. Give an algorithm that tests whether G has a matching of size k that has run-time $O(f(k)(n + m))$, where $f(k)$ is any function that is independent of n and m.

   (d) (Bonus) Same as (c), except make your algorithm work for any graph, not only those with bounded maximum degree.

3. **Vertex Cover in bipartite graphs:** Recall that a graph is called *bipartite* if the vertices can be split as $V = A \cup B$ such that all edges have one endpoint in A and the other in B.

   This problem asks to find a small vertex cover in a bipartite graph. One natural approach to this would be to "take the smaller side", i.e., if $|A| \leq |B|$, then return A as the vertex cover, else return B. Clearly this returns a vertex cover, but is it the minimum?

[1] Knowing the algorithm is not required and will not help to answer the questions.

(a) Prove or disprove: "Taking the smaller side" gives a 2-approximation for minimum vertex cover.

(b) Show that for any $1 \leq a < 2$, "taking the smaller side" does not give an a-approximation.

(c) What can you say if the graph is a tree? (Every tree is bipartite.) Does "taking the smaller side" give the minimum vertex cover? Or at least an a-approximation for some $a < 2$?

4. **Euclidean Travelling Salesman:** Suppose the vertices of a TSP instance are distinct points in the plane, and the weight of an edge is the Euclidean distance between its endpoints. Show that in any optimal tour, no two edges in the tour cross.