Lehrstuhl für Informatik I
Algorithmen und Komplexität
Universität Würzburg

Würzburg, 23 November 2020

Jonathan Klawitter
Boris Klemz
Oksana Firman

4th Exercise Sheet
Advanced Algorithms (WS20)

Exercise 1 – Randomized Max Cut

Let G be the graph shown in Figure 1 (a). Apply the following steps of the algorithm RandomizedMaxCut from the lecture.

a) Formulate the quadratic program QP, whose optimal solution gives a maximal cut for G; i.e. give the variables, the constraints and the objective function with the respective values. **3 Points**

b) Formulate its relaxation QP^k, for $k = 2$. **1 Point**

An optimal solution for QP^2 is shown in Figure 1 (b). For the vectors $x^1, x^2, \ldots, x^6$ we have $x^1 = x^3 = (-10)$, $x^2 = (10)$, $x^4 = (01)$, $x^5 = (0-1)$, and $x^6 = (\frac{1}{\sqrt{2}} - \frac{1}{\sqrt{2}})$.

c) List all cuts that RandomizedMaxCut could compute from this solution and calculate their weight. What is the expected value compared to the optimal solution? **2 Points**

d) Why do we pick the vector r at random to get from a solution of QP^2 (or rather QP^n) to a cut in G? Could we not just pick r such that we get the best cut? **2 Points**

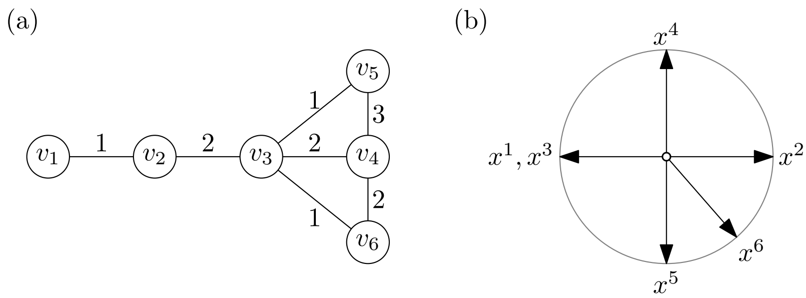

Figure 1: (a) Graph G for Exercise 1 and (b) solution for QP^2.

Exercise 2 – Deterministic 0.5-approximation for MaxCut

In the lecture we saw a randomized 0.5-approximation algorithm for the unweighted $\textsc{MaxCut}$ problem. We now want to derandomize this algorithm with the method of conditional probabilities.

Consider the first vertex v_1 for which we flipped a coin. We now want to decide deterministically, whether we should put v_1 in S or not. For this, we consider the expected weight $\mathrm{E}[W]$ of the cut where v is set to either be in S or not in S but the vertices $v_2, \rightarrow, v_n$ are still assigned randomly. More precisely, we put v in S if and only if $\mathrm{E}[W|v_1 \in S] \geq \mathrm{E}[W|v_1 \notin S]$. Note that $\mathrm{E}[W] = (\mathrm{E}[W|v_1 \in S] + \mathrm{E}[W|v_1 \notin S])/2$. Hence, by our choice $A_1 \in \{S, V \setminus S\}$, we know that $\mathrm{E}[W|v_1 \in A_1] \geq \mathrm{E}[W] \geq 0.5\mathrm{OPT}$. We can repeat this process with v_2 and put it in $A_2 \in \{S, V \setminus S\}$ based on whether $\mathrm{E}[W|v_1 \in A_1, v_2 \in S] \geq \mathrm{E}[W|v_1 \in A_1, v_2 \notin S]$. In fact, we can repeat this for all the reamining vertices $v_3, \ldots, v_n$. However, to develope an algorithm, we need to be able to efficiently compute $\mathrm{E}[W|v_1 \in A_1, \ldots, v_i \in A_i]$.

Describe how we can compute $\mathrm{E}[W|v_1 \in A_1, \ldots, v_i \in A_i]$ efficiently. What do our decision thus mean graph-theoertically? Derive a simple algorithm from this and given it pseudocode.
6 Points

Exercise 3 – QP for MAX-2SAT

Given a conjunctive normal form formula f of Boolean variables $x_1, \ldots, x_n$, and non-negative weights w_c for each clause c of f, the $\textsc{Max-Sat}$ problem asks for a truth assignment to the variables such that the total weight of satisfied clauses is maximized. For the problem $\textsc{Max-2Sat}$ the clauses $c_1, \ldots, c_m$ are restricted to contain at most 2 literals, e.g. $(x_1 \vee \neg x_3)$. Not just $\textsc{Max-Sat}$, but even $\textsc{Max-2Sat}$ is NP-hard.

Give a quadratic program for $\textsc{Max-2Sat}$. **6 Points**

(There exists an approximation algorithm for $\textsc{Max-2Sat}$ similar to the Goemans-Williamson algorithm for $\textsc{MaxCut}$ from the lecture. For details see Vazirani [Vaz Ch 26].)

This assignment is due on November 30 at 10 am. Please submit your solutions via WueCampus. The exercises on this assignment will be discussed in the tutorial session on November 30.