University Côte d'Azur — Graphs
Master 1 — June 2019

Exam : Graphs and Approximation algorithms

3 hours

Documents are allowed but no electronic device is allowed.
You may answer in french if you prefer.
Each of your answers must be explained.

The goal of the following is to analyze approximation algorithms for two problems : Maximum Cut Problem and Minimum Steiner Tree Problem. The two problems are independent.

Notations : In this document, n will always denote the number of vertices of a graph and m will denote the number of its edges.

Recall that, for any $c \geq 1$, a c-approximation algorithm for a maximization problem is an algorithm that computes **in polynomial-time** a feasible solution such that

$$OPT/c \leq value(solution) \leq OPT$$

where OPT is the optimal value.

Question 1 *Let $c \geq 1$. Define a c-approximation algorithm for a minimization problem.*

1 Maximum Cut Problem

Let $G = (V, E)$ be a graph. A *cut* in G is a partition of V into two sets. Let $S \subseteq V$ be a subset of vertices. The *cost of the cut* $(S, V \setminus S)$, denoted by $cost(S)$, equals the number of edges between S and $V \setminus S$, i.e., the size of the set $\{\{x, y\} \in E \mid x \in S, y \in V \setminus S\}$.

The Maximum Cut Problem takes a graph $G = (V, E)$ as input and the objective is to find a cut with maximum cost.

Question 2 *Let $G = (A \cup B, E)$ be a bipartite graph (i.e., A and B are stable sets). Give a maximum cut of G. What is its cost ? (prove that the solution is optimal)*

Question 3 *Give an exponential-time algorithm that computes a maximum cut in arbitrary graphs. Prove that its time-complexity is $O(m \cdot 2^n)$.*

The Maximum Cut Problem is NP-hard, meaning that it does not admit a polynomial-time algorithm unless $P = NP$. The goal of next questions is to analyze an approximation algorithm for it.

Definition : Let $(S, V \setminus S)$ be a cut. A vertex v is *movable* if
— either $v \in S$ and $cost(S) < cost(S \setminus \{v\})$;
— or $v \in V \setminus S$ and $cost(S) < cost(S \cup \{v\})$.
That is, v is movable if moving v on the other side strictly increases the cost of the cut.

Question 4 *To simplify, only in this question, we assume that the Line 1 of Algorithm 1 is replaced by $S = \{g, d\}$. Apply Algorithm 1 on the example depicted in Figure 1.*

Algorithm 1 2-approximation algorithm for MAXIMUM CUT

Require: A graph $G = (V, E)$

Ensure: A cut $(S, V \setminus S)$

1: $S = \emptyset$
2: **while** There is a movable vertex v **do**
3: Move the vertex v on the other side, that is :
— if $v \in S$ then replace S by $S \setminus \{v\}$
— if $v \in V \setminus S$ then replace S by $S \cup \{v\}$
4: **return** S

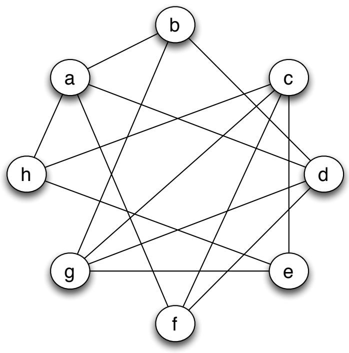

FIGURE 1 – A graph with 8 nodes and 14 edges

Question 5 *What is the maximum number of iterations of the While loop of Algorithm 1 ?*

Let us assume that checking if a vertex is movable can be done in constant time. What is the order of magnitude of the time-complexity of Algorithm 1 ?

Notation : Let $G = (V, E)$ be a graph, $v \in V$ and $X \subseteq V$. Let $deg_X(v)$ denote the degree of v in X, that is the size of the set $\{w \in X \mid \{v, w\} \in E\}$. Let $d(v)$ denote the (classical) degree of v, i.e., $d(v) = deg_V(v)$.

Question 6 *Let $(S, V \setminus S)$ be a solution computed by Algorithm 1.*

1. *Let $v \in S$. Show that $deg_S(v) \leq \lfloor d(v)/2 \rfloor$.*
2. *Similarly, show that $deg_{V \setminus S}(v) \leq \lfloor d(v)/2 \rfloor$ for any $v \in V \setminus S$.*

Question 7 *Let $(S, V \setminus S)$ be a solution computed by Algorithm 1. Let $X = \{\{u, v\} \in E \mid u, v \in S\}$ be the set of edges between nodes in S. Let $Y = \{\{x, y\} \in E \mid x, y \notin S\}$ be the set of edges between nodes in $V \setminus S$. Let $Z = E \setminus (X \cup Y)$ be the set of edges between S and $V \setminus S$.*

1. *By summing the degree of the nodes in S, and using previous question, show that $2|X| \leq |Z|$.*
2. *Similarly, show that $2|Y| \leq |Z|$.*
3. *Deduce that $|Z| \geq |E|/2$.*

Question 8 *Prove that Algorithm 1 is a 2-approximation algorithm for the* MAXIMUM CUT *problem.*

Algorithm 2

Require: A graph $G = (V, E)$, $w : E \to \mathbb{R}^+$ and $R = \{a, b, c\} \subseteq V$

Ensure: ? ? ?

$W \leftarrow \min\{dist(a, b) + dist(b, c); dist(a, b) + dist(a, c); dist(a, c) + dist(b, c)\}$.

for $v \in V \setminus R$ **do**

  $W \leftarrow \min\{W; dist(v, a) + dist(v, b) + dist(v, c)\}$.

return W

2 Minimum Steiner Tree Problem

Let $G = (V, E)$ be a graph with a function $w : E \to \mathbb{R}^+$ on the edges. The *weight* of a subgraph is the sum of the weights of its edges. For any $u, v \in V$, the *distance* $dist(u, v)$ between u and v is the minimum weight of a path between the vertices u and v. In what follows, you can use the Dijktra's algorithm that, for each $v \in V$, computes $\{dist(v, u) \mid u \in V\}$ in time $O(n \log n + m)$.

Given a graph $G = (V, E)$ and a set $R \subseteq V$ of vertices, the Steiner Tree problem consists in computing a **connected** subgraph H of G that spans (contains) all vertices of R and such that H has minimum weight. An example is given on Figure 2.

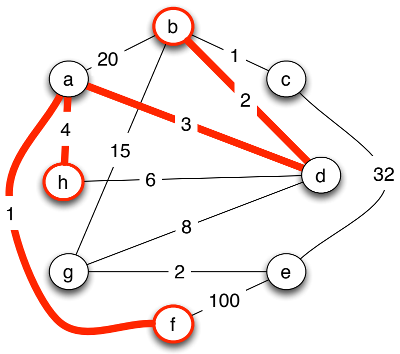

Figure 2 – The bold-red edges represent a minimum weight subgraph spanning the vertices in $R = \{b, f, h\}$

Question 9 *On the example of Figure 2, give a minimum Steiner Tree for the set $R = \{c, e, f\}$. What is its weight ?*

Question 10 *Let $G = (V, E)$ be a graph with weight function $w : E \to \mathbb{R}^+$, $R \subseteq V$ and H a connected subgraph spanning R with minimum weight. Prove that H is a tree.*

Next questions consider the Steiner Tree Problem for small sets R.

Question 11 *Give a polynomial-time algorithm for solving the* Steiner Tree *Problem in the case when $|R| \leq 2$.*

Question 12 *Draw all trees with at most 3 leaves and no vertices with degree 2.*

In the case $|R| = 3$, we can "guess" the shape of a minimum Steiner Tree to compute it efficiently.

Question 13 *Explain what is the goal of Algorithm 2. How does it proceed ?*

Question 14 *What is the time-complexity of Algorithm 2 ?*

The Steiner Tree Problem is NP-hard when the size of R is not bounded. The goal of next questions is to design and to analyze an approximation algorithm for it.

Let $G = (V, E)$ be a graph, $w : E \to \mathbb{R}$ and $R \subseteq V$. The weighted graph (G_R, w_R) is the complete graph with R as set of vertices and, for every two vertices $u, v \in R$, $w_R(\{u, v\}) = dist_G(u, v)$ (i.e., the weight of an edge $\{u, v\}$ in G_R is the distance between u and v in G).

Question 15 *Give the definition of an Hamiltonian cycle.*

Question 16 *Describe an algorithm that computes an Hamiltonian cycle of G_R whose weight is at most twice the weight of an minimum Hamiltonian cycle in G_R.*

Question 17 *Describe (and prove) a* 2*-approximation algorithm for the* Steiner Tree *Problem.*