MA6150. Algoritmos de Aproximación. 2023.
Profesor: José Soto
Última Actualización: 8 de agosto de 2023.

Ingeniería Matemática
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
UNIVERSIDAD DE CHILE

Guía de ejercicios.

Esta guía se irá actualizando con ejercicios para el curso. En cada tarea se pedirá entregar cierto puntaje mínimo de ellos, de ciertas secciones. Se podría solicitar entregar algún problema específico adicionalmente. Se recomienda trabajar y resolver ejercicios de la guía de manera sistemática.

1. Generales

En todos los algoritmos que proponga debe justificar el factor de aproximación objenido.

Ejercicio 1.1 (1 punto)**.** En el problema MÁXIMO SUBDIGRAFO ACÍCLICO, la entrada es un digrafo $G = (V, E)$ y la salida debe ser un conjunto de arcos $F \subseteq E$ acíclico (en el sentido dirigido) de cardinalidad máxima.

(a) Encuentre una 2-aproximación a este problema. **Indicación:** Enumere $V = [n]$ y llame a un arco $e = (i, j) \in E$ *directo* si $i < j$ o *reverso* si $j < i$. Estudie ambos conjuntos.

(b) Suponga ahora que el grafo $\overline{G}$ no dirigido subyacente de G es conexo y simple. Llame $\rho = |E|/(|V| - 1)$ a la densidad de G. Encuentre una $\text{mín}(\rho, 2)$-aproximación para el problema del *maximo subdigrafo acíclico*.

Ejercicio 1.2 (1 punto)**.**

(a) En el problema del MÍNIMO MATCHING MAXIMAL (MMM) el objetivo es encontrar un matching maximal M de cardinalidad mínima. Demuestre que el algoritmo glotón que encuentra cualquier matching maximal es una 2-aproximación para este problema.

(b) Considere ahora el mismo problema anterior pero en un hipergrafo k-uniforme. ¿Es o no cierto que el mismo algoritmo glotón es una k-aproximación para MMM? (justifique o dé un contraejemplo)

Ejercicio 1.3. (1 punto) Sea $k \geq 1$ cualquier constante. Demuestre que si $\mathbf{P} \neq \mathbf{NP}$ no existe algoritmo para MIN VERTEX COVER que devuelva un vertex cover W con garantía aditiva de k, es decir, tal que $|W| \leq \text{OPT} + k$.

Ejercicio 1.4 (1 punto)**.**
Encuentre una $O(n/\log n)$ aproximación para MAX CLIQUE siguiendo el siguiente esquema: Para $k = k(n)$ cierta función de n, elija una partición cualquiera de V en grupos de tamaño k. Revisar *todos* los subconjuntos de V completamente contenidos en alguno de los grupos y devolver de todos ellos el clique más grande. Primero, demuestre que si $k(n) = O(\log n)$ entonces el algoritmo anterior se puede implementar en tiempo polinomial. Luego demuestre que la solución obtenida es una $n/k(n)$ aproximación. Finalmente use $k(n) = \log n$ para concluir.

Ejercicio 1.5. (1 punto) Considere el siguiente algoritmo para Min Vertex Cover

Algoritmo 1: Para encontrar un vertex-cover.

Entrada: Un grafo G.
Salida: Un vertex cover ALG
ALG $\leftarrow \emptyset$.
$G' \leftarrow G$
mientras $E(G') \neq \emptyset$ **hacer**
  Sea $v \in V(G')$ el vértice de mayor grado de G'.
  Añadir v a ALG.
  Borrar de G', el vértice v y todas las aristas incidentes a v.
fin
devolver ALG.

(a) Muestre que el algoritmo anterior es correcto (devuelve un vertex cover en tiempo polinomial).

(b) Muestre una familia infinita de instancias donde el algoritmo anterior devuelva una solución ALG con alg $\geq \Omega(\log n)$opt. **Indicación:** Considere el grafo G_n bipartito con lado izquierdo $[n]$ y lado derecho $\mathcal{P}([n])$, donde para cada $j \in [n]$ y $A \in \mathcal{P}([n])$, jA es arista si y solo si $j \in A$. ¿Qué podría pasar en este grafo?

Ejercicio 1.6. (1 punto) El problema MAX CUT se define como sigue. Dado un grafo G, encontrar un conjunto $S \subseteq V(G)$ (un corte) tal que el número de aristas con un extremo en S y otro en $V(G) \setminus S$ se maximice, es decir estamos maximizando $|\delta(S)| = |E(S : V(G) \setminus S)|$. Considere el siguiente algoritmo para MAX CUT.

Algoritmo 2: Para encontrar un corte.

Entrada: Un grafo G.
Salida: Un corte ALG
ALG $\leftarrow \emptyset$.
mientras *verdadero* **hacer**
  si *Existe* $v \in V(G) \setminus$ ALG *tal que* $|\delta(\text{ALG} + v)| \geq |\delta(\text{ALG})|$ **entonces**
    ALG $\leftarrow$ ALG $+ v$.
  si no, si *Existe* $v \in$ ALG *tal que* $|\delta(\text{ALG} - v)| \geq |\delta(\text{ALG})|$ **entonces**
    ALG $\leftarrow$ ALG $- v$.
  en otro caso
    devolver ALG.
  fin
fin

Muestre que el algoritmo anterior es correcto (devuelve un corte en tiempo polinomial) y que es una 2-aproximación para MAX CUT.