Práctica 8

Optimización Combinatoria

8.1 Probar que en cualquier grupo de personas hay 2 que tienen la misma cantidad de amigos dentro del grupo.

8.2 ¿Es posible que haya un grupo de 7 personas tal que cada persona conozca exactamente otras 3 personas del grupo?

8.3 Dado el algoritmo de *Kruskal* para determinar un árbol generador mínimo de un grafo $G = (V, X)$ y una función $l(e) : X \to \mathbb{R}$

`PASO 1:` `poner` $i = 1, T = \{e\}$`, con` e `de longitud mínima`

`PASO 2:` `mientras` $i < n - 1$ `hacer:`

`PASO 3:`   `elegir` e `tal que` $l(e)$ `sea mínima entre los que no forman circuito con las aristas que ya están en` T

`PASO 4:`   `poner` $T = T \cup \{e\}$

`PASO 5:`   `poner` $i = i + 1$

a. Probar que el algoritmo de *Kruskal* construye un árbol generador mínimo.

b. Determinar la complejidad del algoritmo de *Kruskal*.

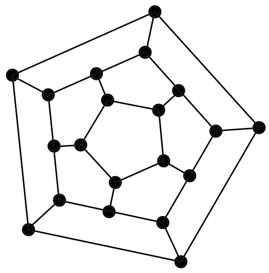
c. ¿Qué técnica utiliza este algoritmo?

8.4 Sea K_n el grafo completo de n vértices.

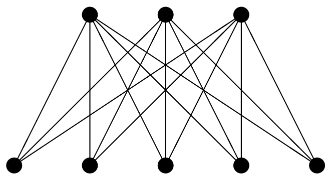
a. ¿Para qué valores de n, K_n tiene circuito euleriano?

b. ¿Hay algún K_n que tenga camino euleriano pero no circuito?

8.5 El siguiente es el grafo original en el cual Hamilton basó su juego. Encontrar un circuito hamiltoniano en el mismo. Una versión del juego original consistía en que uno de los jugadores elegía un camino con 5 vértices y el otro debía extender el camino a un circuito hamiltoniano. ¿Hay algún camino simple de 5 vértices que no pueda ser extendido a un circuito hamiltoniano?



8.6 Decir por qué el grafo de la figura no tiene camino ni circuito hamiltoniano.



8.7 Dado un grafo $G = (V, X)$ donde a cada arista $e \in X$ se le ha asignado una longitud $l(e)$. El problema conocido como problema del viajante de comercio consiste en determinar el circuito hamiltoniano de longitud total mínima. Analizar el siguiente algoritmo para resolver el problema del viajante en forma aproximada:

```
PASO 1: Dar un grafo completo G con longitudes asignadas a las aristas.
PASO 2: Determinar un árbol generador mínimo T de G.
PASO 3: Construir un multigrafo D(T) reemplazando cada arista de T por
        dos aristas.
PASO 4: Usar el algoritmo de Euler para determinar un circuito Euleriano
        en D(T). Suponer que el circuito queda formado por las aristas
        {(x_1, y_1), (x_2, y_2), ..., (x_{2n-2}, y_{2n-2})}
PASO 5: Poner C ← {x_1} y marcar x_1.
PASO 6: Para i ← 1, 2n − 3 hacer
PASO 7:     Si y_i no está marcado, marcarlo y poner C ← C ∪ {y_i}.
PASO 8:     Poner C ← C ∪ {x_1}.
PASO 9: Informar C y parar.
```

a. Mostrar un ejemplo en que el resultado del algoritmo anterior no sea un circuito hamiltoniano de longitud mínima.
b. Analizar la complejidad del algoritmo.
c. Probar que si $l(C)$ es la longitud del circuito determinado por el algoritmo y $l(H)$ es la longitud de un circuito hamiltoniano de longitud mínima y además en el grafo se cumple que vale la desigualdad triangular para las distancias, entonces $l(C) \leq 2\ l(H)$. Comentar este resultado.

8.8 Analizar el siguiente algoritmo goloso (método del vecino más cercano) para resolver el problema del viajante de comercio en un grafo completo. Elegir v_1 un vértice cualquiera. Poner $C_1 = \{v_1\}$. Mientras C_j no contenga todos los vértices de V, elegir v_{j+1} como el vértice más cercano a v_j y poner $C_{j+1} = C_j \cup \{v_{j+1}\}$. Agregar la arista entre v_n y v_1.

a. Mostrar que este no es un algoritmo exacto para resolver el problema del viajante de comercio. Determinar su complejidad.

b. ¿Es una buena heurística?

c. Escribir un nuevo algoritmo para el mismo problema, que elija en primer lugar el vértice más lejano a v_1, y después vaya eligiendo e insertando en cada paso el vértice que aún no está en el camino ya construido C_j y que está mas cerca de algún vértice de C_j. ¿Es este algoritmo exacto? ¿Es bueno? ¿Es mejor que al anterior? ¿Peor?

Clases de complejidad

8.9 Mostrar que los siguientes problemas pertenecen a P:

a. Ordenar una lista de n números.

b. Chequear si un grafo es conexo.

c. Chequear si un grafo tiene un ciclo.

8.10 Decir si las siguientes afirmaciones son verdaderas, falsas o no se sabe:

a. $P \subseteq NP$

b. $P \subseteq \textit{co-NP}$

c. $P = NP \cap \textit{co-NP}$

d. $P = NP \Rightarrow \textit{co-NP} = NP$

e. $\textit{co-NP} = NP \Leftrightarrow \text{SAT} \in \textit{co-NP}$

f. $\textit{co-NP} \subseteq NP \Rightarrow NP = \textit{co-NP}$

g. $\textit{co-NP} \subseteq NP \Rightarrow P = NP$

8.11 ¿Es cierto que si dos problemas X e Y son *NP-completos* entonces $X \leq_p Y$, y también $Y \leq_p X$? Justificar.

8.12 Sean X e Y dos problemas de decisión tales que $X \leq_p Y$. ¿Qué se puede inferir?

a. Si X está en P entonces Y está en P.

b. Si Y está en P entonces X está en P.

c. Si Y es *NP-completo* entonces X también.

d. Si X es *NP-completo* entonces Y también.

e. Si Y es *NP-completo* y X está en *NP* entonces X es *NP-completo*.

f. Si X es *NP-completo* e Y está en *NP* entonces Y es *NP-completo*.

8.13 Decir si las siguientes afirmaciones son verdaderas o falsas:

a. Si $P = NP$ entonces todo problema *NP-completo* es polinomial.

b. Si $P = NP$ entonces todo problema *NP-hard* es polinomial.

c. Las clases *NP-completo* y *co-NP-completo* son disjuntas si y solamente si $P \neq NP$ (un problema de decisión es *co-NP-completo* cuando pertenece a *co-NP* y todo otro problema perteneciente a *co-NP* es polinomialmente reducible a él).

8.14 ¿Qué se puede inferir del hecho de que el problema del viajante de comercio (TSP) es *NP-completo*, suponiendo $P \neq NP$?:

a. No existe un algoritmo que resuelva instancias arbitrarias de TSP.

b. No existe un algoritmo que eficientemente resuelva instancias arbitrarias de TSP.

c. Existe un algoritmo que eficientemente resuelve instancias arbitrarias de TSP, pero nadie lo ha encontrado.

d. TSP no está en P.

e. Todos los algoritmos que resuelven TSP corren un tiempo polinomial para algunas entradas.

f. Todos los algoritmos que resuelven TSP corren un tiempo exponencial para todas las entradas.

8.15 a. Sea un grafo $G = (V, X)$ y $W \subseteq V$. Demostrar la equivalencia de las siguientes afirmaciones:

- $V - W$ es un recubrimiento de arcos de G.
- W es un conjunto independiente de G.
- W es un clique de $\overline{G}$ (el grafo complemento de G).

b. A partir del punto anterior, encontrar reducciones polinomiales entre los problemas Mínimo Recubrimiento de Arcos, Máximo Conjunto Independiente y Máximo Clique.

c. ¿A qué clase de complejidad pertenecen estos 3 problemas? ¿Por qué?

8.16 Encontrar una reducción polinomial del problema *Circuito Hamiltoniano* al problema del viajante de comercio (*TSP*) (sugerencia: construir un grafo completo con pesos en los arcos tales que el menor circuito hamiltoniano corresponda a un circuito hamiltoniano del grafo original). Sabiendo que el primer problema es *NP-completo*, ¿qué se puede decir del segundo?

8.17 Utilizando la NP-completitud de otro problema, demostrar que el problema *Camino mínimo condicionado* es *NP-completo*:
Entrada: un grafo $G = (V, X)$ con un peso positivo asociado a cada eje, un par de vértices $s, t \in V$, un subconjunto de nodos $V' \subseteq V$ y un entero $k > 0$.
Pregunta: ¿Existe un camino entre s y t en G que pasa por todos los vértices de V' y de peso total menor o igual que k?

8.18 Mostrar que el problema de decidir si una fórmula booleana tiene al menos dos diferentes asignaciones de variables que la hacen verdadera, es *NP-completo*.

Bibliografía para la práctica 8

Los libros marcados con (a) están en la biblioteca central (Pabellón II), los marcados con (b) en la Hemeroteca del Departamento de Matemática (segundo piso del Pabellón I) y los marcados con (c) en la Infoteca del Departamento de Computación (entrepiso del Pabellón I).

1. R. Ahuja, T. Magnanti, J. Orlin, *Network flows*, Prentice Hall, 1993. (a,c)
2. C. Berge, *Graphs*, North-Holland, 1985. (b)
3. C. Berge, *The theory of graphs and applications*, Wiley, 1958. (a)
4. J. Bondy, U. Murty, *Graph theory with applications*, Macmillan Press, 1976.
5. R. Campello, N. Maculan, *Algoritmos e heurísticas*, Editorial da Universidade Federal Fluminense, 1994. (a,c)
6. S. Even, *Graph algorithms*, Computer Science Press, 1979.
7. L. Ford, D. Fulkerson, *Flows in Networks*, Princeton University Press, 1962. (b)
8. M. Garey, D. Jonhson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. Freeman and Co., 1979. (a,c)
9. M. Gondran, M. Minoux, *Graphs and Algorithms*, John Wiley and Sons, 1984.
10. J. Gross, J. Yellen, *Graph theory and its applications*, CRC, 1999. (c)
11. F. Harary, *Graph theory*, Addison-Wesley, 1969 (hay una reedición de 1996). (a)
12. Mc Hugh James, *Algorithmic Graph Theory*, Prentice-Hall International, 1990.
13. C. Papadimitriou, *Computational Complexity*, Addison-Wesley, 1995.
14. V. Rayward-Smith, I. Osman, C. Reeves, G. Smith, *Modern heuristic search methods*, Wiley, 1996.
15. C. Reeves, *Modern heuristic techniques for combinatorial problems*, Blackwell, 1993.
16. M. Syslo, N. Deo, J. Kowaljk, *Discrete Optimization Algorithms*, Prentice Hall, 1983.
17. M. Swamy, D. Thulasiraman, *Graphs, networks and algorithms*, John Wiley and Sons, 1981.
18. J. Szwarcfiter, *Grafos e algoritmos computacionais*, Editora Campus, Rio de Janeiro, 1987.