Algorítmica - Practica 3: Algoritmos Voraces

A. Herrera, A. Moya, I. Sevillano, J.L. Suarez

30 de abril de 2015

Contents

1 Organización de la práctica

La práctica 3 trata sobre el desarrollo de algoritmos greedy que consigan la solución óptima de los problemas propuestos o actúen como heurística sobre los mismos. Uno de los problemas a estudiar es el viajante de comercio, ampliamente conocido en el ámbito de la inteligencia artificial y teoría de algoritmos. Es un problema NP completo pero que será abordable gracias al uso de algoritmos greedy polinomiales que no proporcionarán la solución óptima pero sí una lo suficientemente buena para nuestros objetivos.

Nuestro grupo debe resolver el problema 4 y el viajale de comerntcio. Hemos abordado también el problema opcional (número 5) optentiendo el algoritmo greedy óptimo.

Para cada problema se sigue la siguiente estructursa:

- Enunciado del problema
- Resolución teórica del problema (con una subsección por algoritmo)
- Análisis empírico. Análisis de la eficiencia híbrida

En este último apartado se proporcionan gráficas con los resultados de los algoritmos y un análisis de la eficiencia híbrida para los mismos.

Los algoritmos se han ejecutado sobre un ordenador con las siguientes características:

- **Marca:** Toshiba
- **RAM:** 8 GB
- **Procesador:** Intel(R) Core(TM) i5-3210M CPU @ 2.50GHz

El código, los resultados de las ejecuciones, las gráficas y los pdf asociados se puede encontrar en GitHub.

2 Problema 4

2.1 Enunciado del problema

Consideremos un grafo no dirigido $G = (V, E)$. Un conjunto U se dice que es un recubrimiento de G si $U \subseteq V$ y cada arista en E incide en, al menos, un vértice o nodo de U, es decir,

$$\forall (x, y) \in E : x \in U \text{ o } y \in U$$

Un conjunto de nodos es un recubrimiento minimal de G si es un recubrimiento con el menor número posible de nodos.

- Diseñar un algoritmo greedy para intentar obtener un recubrimiento minimal de G. Demostrar que el algortmo es correcto, o dar un contraejemplo.
- Diseñar un algoritmo greedy que obtenga un recubrimiento minimal para el caso particular de grafos que sean árboles.
- Opcionalmente, realizar un estudio experimental de las diferencias entre los dos algoritmos anteriores cuando ambos se aplican a árboles.

2.2 Solución teórica

2.2.1 Solución para un grafo arbitrario.

En primer lugar, nótese que si tomamos $U = G$ tenemos un recubrimiento. Este será el peor recubrimiento porsible pues utiliza todos los nodos del grafo. Queremos obtener un mejor recubrimiento. Para ello, construyamos uno desde 0:

1. $U = \emptyset$
2. Para cada arista $(x, y) \in E$ tomamos como nodo x o y aleatoriamente y lo añadimos a U

Este algoritmo voraz aleatorio obtiene un recubrimiento del grafo pues para cada arista hay efectivamente un nodo en U sobre el que esta incide. El tiempo de ejecución es lineal sobre el número de aristas, $|E|$. Sin embargo, es claro que las soluciones obtenidas serán manifiestamente mejorables. Podemos mantener este tipo de construcción pero librarnos parcialmente de la aleatoriedad en la elección voraz del algoritmo:

1. $U = \emptyset$
2. Para cada arista $(x, y) \in E$ tomamos un nodo v y lo añadimos a U donde v es:
   - x si $x \in U$.
   - y si $y \in U$.
   - Uno de los dos, elegido aleatoriamente, si $x, y \notin U$.

En efecto, si algún nodo sobre el que incide la arista ya está en U no tenemos por qué añadir uno nuevo.

Sin embargo, este algoritmo lineal y aleatorio no es óptimo. Para obtener un algoritmo voraz óptimo necesitamos construir la solución desde otra perspectiva.

2.2.1.1 Solución óptima