Ejercicios de Grafos - Tercera Parte

1. Modificar el algoritmo BFS para que calcule la distancia mínima entre dos vértices dados, y que además nos permita recuperar un camino mínimo.

2. Diseñar un algoritmo para resolver el problema del laberinto en su versión más general con una complejidad *razonable*.

   Nota: En `http://www.spoj.pl/problems/ESJAIL` se puede consultar el enunciado.

3. Diseñar un algoritmo para resolver el problema de camino mínimo/máximo en un DAG (grafo dirigido acíclico, es decir, sin ciclos orientados).

4. Una persona dispone de un departamento que desea alquilar durante el verano. Luego de publicarlo en los clasificados obtiene N ofertas. Cada oferta consta de 3 datos: el monto ofrecido, el día de inicio y el día de finalización.

   Esta persona desea maximizar sus ganancias. Modelar el problema como un problema de grafos y resolverlo con un algoritmo conocido.

5. Calcular la complejidad del algoritmo de Dijkstra en el caso en que π se guarda en una parva (heap) en lugar de en una lista.

6. Escribir en pseudocódigo los algoritmos de Floyd y de Dantzig.

7. Sea $G = (V, E)$ un grafo con costos en las aristas. Notemos c_{ij} el costo de la arista que une v_i y v_j.

   (a) Dado π un vector de n coordenadas consideremos unos nuevos costos $c^{\pi}_{ij} = c_{ij} + \pi_i - \pi_j$. Mostrar que los caminos mínimos con estos nuevos costos son los mismos que antes (el costo total de cada camino puede cambiar).

   (b) Supongamos que G no tiene ciclos negativos y sea s un nodo distinguido en G, tomemos π_i como la longitud del camino minimo (con los costos originales) desde s hasta v_i. Demostrar que $\pi_j \leq \pi_i + c_{ij}$.

   (c) Probar que si vale $\pi_j \leq \pi_i + c_{ij}$ entonces los nuevos costos son no-negativos.

   (d) Supongamos que los vértices de G son puntos del plano y los costos son las distancias euclideas. Demostrar que tomando $\pi_i = ||s - v_i||_2$ se tiene $\pi_j \leq \pi_i + c_{ij}$.

8. Mostrar cómo resolver el problema de matching máximo bipartito, considerándolo como un problema de flujo máximo en una red conveniente.

Nota: El problema de matching máximo bipartito consiste en: dado un grafo bipartito elegir un conjunto de aristas del mayor cardinal posible con la propiedad de que en cada vértice incide a lo sumo una arista del conjunto.

9. ¿Cómo se puede resolver el problema de flujo máximo en una red si los vértices también tienen capacidades máximas?.

10. Sea $G = (V, E)$ un grafo y $v, w \in V$. ¿Cuántos caminos disjuntos en aristas se pueden encontrar simultáneamente entre v y w? ¿Y disjuntos en vértices?.

11. Resolver el problema

    `http://acmicpc-live-archive.uva.es/nuevoportal/data/p4479.pdf`

    considerándolo como un problema de flujo máximo en redes.

12. Sea $a_1, a_2, \cdots, a_{2n}$ una lista de números naturales sin repeticiones. Se quieren armar parejas con estos números, de forma que en cada pareja la suma de ambos números sea primo. ¿Cuál es la máxima cantidad de parejas que se pueden armar?. Modelar este problema como un problema de flujo.

    Ejemplo: Si los números son 2,3,4,5 se pueden armar 2 parejas 2-5, 3-4. Pero si son 2,3,4,6, sólo se puede armar una pareja.

13. Dadas d_{in} y d_{out} dos n-tuplas de naturales se desea saber si existe un grafo dirigido G tal que los grados de entrada y salida de vértices v_i sean $d_{in}(i)$ y $d_{out}(i)$ respectivamente. Modelar como un problema de flujo en redes.

14. Modelar los problemas de flujo máximo y de flujo de mínimo costo como problemas de programación lineal. Implementarlos en algunos ejemplos usando ZIMPL.

    Opcional: Implementar algunos de los algoritmos vistos en clase y comparar los tiempos de ejecución con los programas hechos en ZIMPL.¿Cuál es más rápido? ¿Cuál debería ser mas rápido?