**Diseño y Análisis de Algoritmos - Memoria Práctica 2**

Pareja 05

Borja González Farías

Darío Adrián Hernández Barroso

Cuestiones sobre circuitos y caminos eulerianos:

**1- Tras las funciones anteriores nos hemos dado cuenta de que las mismas no tratan sobre grafos NO dirigidos. Indicar brevemente qué cambios harían falta, si alguno, para que nuestras funciones pudieran buscar caminos y circuitos eulerianos en tales grafos.**

Para detectar caminos y circuitos en grafos no dirigidos habría que cambiar:

En primer lugar, la estructura de guardado. Tal y como usamos actualmente los diccionarios se guardaría mucha información duplicada, dado que en un grafo no dirigido la matriz tiene, de manera simétrica, los mismos datos en la parte superior de la diagonal y la parte inferior. Por lo tanto, la información guardada en el diccionario habría que sacarla sólo de una de las mitades de la matriz.

Por otra parte, habría que cambiar también la forma en la que detectamos la posibilidad de caminos y circuitos eulerianos. En ésta práctica usamos las listas de adyacencia e incidencia, pero para grafos no dirigidos hay que usar una lista con los grados de cada nodo, y usar la correspondiente comprobación.

Por último, la función para elegir el nodo inicial y el nodo final. Para grafos no dirigidos tienen que ser aquellos 2 con grado impar, devolviendo uno como inicial y otro como final.

**2- El criterio sobre existencia de caminos o circuitos eulerianos sólo funciona cuando el grafo no contiene subgrafos disjuntos. ¿Cómo podríamos detectar dicha condición?**

En primer lugar, usar DFS desde un nodo aleatorio e ir guardando los nodos que encuentra en una lista.

Después, comprobar si en tu diccionario existe algún nodo w que no se encuentra dentro de la lista generada. Si hay un w que cumpla esa condición, entonces tienes subgrafos disjuntos.

Cuestiones sobre reconstrucción de secuencias

**1- Si se tiene una secuencia *P* de longitud lp y se usan lecturas de longitud lr para reconstruirla mediante un camino o circuito euleriano sobre un grafo G = (V,E), queremos estudiar cuántos nodos |V| y cuántas ramas |E| hay y cuál sería el sparsity factor *p*.**

**Para ello vamos a hacer un pequeño estudio experimental de estas cantidades, fijando una longitud lp suficientemente grande y estimando el número de vértices, ramas y el sparsity factor para diferentes valores de lr. Escribir una función Python que estime dichos números de vértices, ramas y sparsity factor, describir sus resultados y comentarlos.**

**2- El coste de reconstruir una secuencia no sólo depende de la búsqueda de un camino euleriano sino también del tiempo empleado en la construcción del grafo.**

**Discutir los costes asociados a las funciones *spectrum(sequence, len\_read), spectrum\_2(spectr) y spectrum\_2\_undirected\_graph(spectr)* definidas más arriba de acuerdo a la implementación que se haya seguido y, en consecuencia, el coste total del algoritmo de reconstrucción de secuencias.**

C(spectrum) = O(N), puesto que hay que recorrer la lista e ir partiéndola, y por lo tanto se hace para cada subsecuencia de tamaño len\_read.

C(spectrum\_2) = O(N^2). Hay que recorrer la lista con el espectro, y después volverla a recorrer para quitar los repetidos (crear un set desordena el spectro-1)

C(spectrum\_2\_undirected\_graph) = O(N^2) + O(N^2) = O(N^2). Ésta función llama a spectrum\_2 (primer O(N^2)) y después se realizan dos bucles, recorriendo las keys del diccionario a crear y otro para rellenarlo con la información necesaria. Hay un tercer bucle en ésta función, pero es de tamaño variable, pues sirve para escribir las ramas que hay de un nodo a otro correctamente.

Una vez sacados estos costes observamos el C(Total):

C(Total) = C(spectrum) + C(spectrum\_2\_undirected\_graph) = O(N+N^2)