Tarea 6.
Fecha de entrega: 3 de diciembre de 2021.

1. Tarea

Un árbol es un tipo de gráfica muy simple: es conexo y no tiene ciclos, lo cual permite que en ellos se puedan resolver problemas de forma más simple que en las gráficas en general. Una trayectoria es un tipo de árbol con estructura aún más simple: una gráfica es una *trayectoria*, si todos sus vértices, excepto a lo más dos, tienen grado exactamente dos (intuitivamente, se pueden dibujar con todos los vértices sobre una línea, y las aristas conectan vértices consecutivos sobre la línea). Esta estructura aún más simple permite resolver problemas de forma aún más fácil que en árboles.

Definición: En una gráfica G, un conjunto S de vértices es *independiente*, si para todo par de vértices $v_i, v_j \in V(G)$, $(v_i, v_j) \notin E(G)$.

Considera el siguiente problema (sobre gráficas en general se sospecha de complejidad exponencial, pero sobre árboles se puede resolver de forma relativamente simple, y sobre trayectorías aún más).

1. 10pt. Problema *conjunto independiente de peso máximo (en trayectorias)*. Entrada: Una secuencia de n pesos $W = [w_1, w_2, \ldots, w_n]$. Salida: un conjunto independiente (representado como la lista de índices de sus vértices) de peso máximo en la trayectoria de vértices $v_1, v_2, \ldots, v_n$, con v_i adyacente a v_{i+1}, y siendo w_i el peso del vértice v_i.

   Ejemplos de instancias:

   a) Entrada: [2,4,1]. Salida: [2] (un conjunto independiente de peso máximo es $\{v_2\}$.

   b) Entrada: [8,2,4,6]. Salida: [1,4] (un conjunto independiente de peso máximo es $\{v_1, v_4\}$.

   c) Entrada: [6,7,5]. Salida: [1,3].

   Tips: Observa que en una trayectoria, hay dos tipos de conjuntos independientes: i) los que no contienen al último vértice y ii) los que sí contienen al último vértice. Observa que los de tipo i) son exactamente los conjuntos independientes de la trayectoria que resulta de eliminar el último vértice; y observa que los de tipo ii) son exactamente los conjuntos independientes de la trayectoria que resulta de eliminar el último *y el penúltimo* vértice. Esto sugiere considerar los subproblemas de la forma: ¿cuál es el peso máximo de algún conjunto independiente en la subtrayectoria de vértices $v_1, v_2, \ldots, v_i$?. Si definimos entonces $opt(i)$ como la solución del subproblema que corresponde a la subtrayectoria $v_1, \ldots, v_i$, la observación ya mencionada nos permite establecer la ecuación de Bellman (la que expresa el valor de la solución óptima de una subinstancia, en función de los valores óptimos de sus subinstancias recursivas), para este problema:

$$opt(i) = \begin{cases} 0 & (i == 0) \\ W[i] & (i == 1) \\ \text{máx}(opt(i-1), opt(i-2) + W[i]) & (i > 1) \end{cases}$$

a) 3pt. Propón un algoritmo recursivo (de complejidad exponencial) que resuelva el problema, en su versión de cálculo del valor óptimo, pero no de la estructura óptima. Analiza su corrección y su complejidad.

b) 3pt. Propón la versión recursiva con memorización del ejercicio anterior. Asegúrate que devuelve el mismo valor que el algoritmo anterior, y por eso se transfiere su correción, y argumenta su complejidad contando el número máximo de subinstancias distintas en el árbol de recursión.

c) 3pt. Propón la versión iterativa de programación dinámica del algoritmo anterior (declara las matrices M_{opt} y $M_{choices}$, establece la semántica de cada una de sus celdas, y propón un orden de llenado correcto, describe la función de llenado). Asegúrate que devuelve el mismo valor que el algoritmo anterior, y por eso se transfiere su correción, y demuestra que su complejidad es $O(n)$.

d) 1pt. Propón el algoritmo que, a partir de la tabla M_{choice} generada por el algoritmo anterior, calcula el conjunto independiente de peso máximo.