



1. Dados dos árboles generadores  $T$  y  $R$  de una gráfica  $G = (V, E)$ , muestra cómo encontrar la secuencia más corta de árboles generadores  $T_0, T_1, \dots, T_k$  tal que  $T_0 = T$ ,  $T_k = R$  y cada árbol  $T_i$  difiere del árbol anterior  $T_{i-1}$  agregando y borrando una arista.
2. Sea  $G$  una gráfica cuyas aristas tienen asignados pesos positivos. Sea  $T$  un árbol generador de peso mínimo de  $G$ . Pruebe que existen aristas  $e \in T$  y  $e' \notin T$  tales que  $T - \{e \cup e'\}$  forman un árbol de peso mayor o igual que  $T$ , pero menor o igual a cualquier otro árbol generador de  $G$ , i.e. un segundo árbol generador de peso mínimo.
3. Una empresa está planeando una fiesta para sus empleados. Los organizadores de la fiesta quieren que sea una fiesta divertida, por lo que han asignado una calificación de “diversión” a cada empleado. Los empleados están organizados en una estricta jerarquía, es decir, un árbol enraizado en el presidente. Sin embargo, hay una restricción en la lista de invitados a la fiesta: tanto un empleado como su supervisor inmediato (padre en el árbol) no pueden asistir a la fiesta (porque eso no sería divertido). Diseñe un algoritmo de tiempo lineal que haga una lista de invitados para la fiesta y que maximice la suma de las calificaciones de “diversión” de los invitados.
4. Supongamos que usted quiere marcar un número de  $n$  dígitos  $\{r_1, r_2, \dots, r_n\}$  en un teléfono normal en el que los números están en un arreglo normal de  $4 \times 3$  teclas utilizando sólo dos dedos. Supongamos que al comenzar a marcar, sus dedos están en las teclas “\*” y “#”. Encuentre un algoritmo de tiempo lineal (programación dinámica) que minimiza la distancia Euclídeana que tienen que recorrer sus dedos.
5. Sea  $S$  un conjunto de  $n$  puntos en el plano y en posición general, tales que  $\forall (x_i, y_i) \in S$  se tiene que  $x_i, y_i \in \mathbb{N}$  y  $x_i, y_i \in [0, \dots, n^2]$ . Describe un algoritmo que encuentre el cierre convexo de  $S$  en tiempo  $O(n)$ .

---

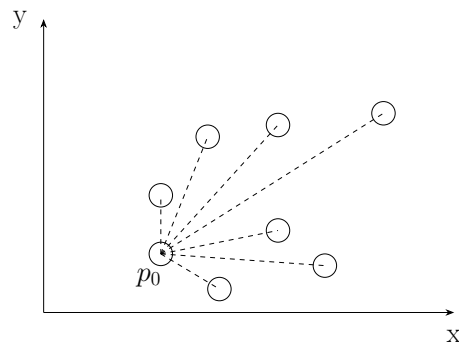
Lo primero a notar es que no podemos aplicar metodos como el de Graham o el de Jarvis, ya que estos tienen una complejidad de  $O(n \log n)$  y en este caso se pide un algoritmo de complejidad  $O(n)$ .

### Graham Scan + Radix Sort

Como bien dice el titulo, vamos a usar el algoritmo de Graham Scan para encontrar el cierre convexo de  $S$ . Ademas, vamos a usar el algoritmo de Radix Sort para ordenar los puntos de  $S$  en tiempo  $O(n)$ . A continuación se describe el algoritmo:

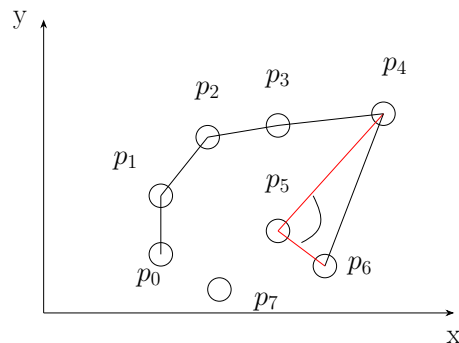
Primero, vamos a buscar el mas chico en la coordenada x, lo vamos a llamar  $p_0$ . Luego, vamos a ordenar los puntos de  $S$  en orden decreciente de angulo abarcado entre el segmento que une a  $p_0$  con el punto y el eje x, se puede utilizar la cotangente para agilizar este proceso, ademas por la reestriccion de que los puntos estan en el rango  $[0, \dots, n^2]$  podemos usar Radix Sort para ordenar los puntos en tiempo  $O(n)$ .

Eso nos va a dar algo de este estilo:



Entonces buscar el mas chico en una coordenada nos tomo  $O(n)$  y ordenar los puntos nos tomo  $O(n)$ , por lo que hasta ahora llevamos  $O(n)$ . Ahora, vamos a aplicar el algoritmo de Graham Scan para encontrar el cierre convexo de  $S$ .

Ahora Graham va a tomar una pila, meter a  $p_0$  y a  $p_1$ , luego va a ir tomando los puntos de  $S$  de a uno y va a ir viendo si el giro que forma el punto actual con los dos ultimos puntos de la pila es a la izquierda o a la derecha. Si el movimiento es a la derecha entonces continua y mete a el siguiente punto en la pila, si el movimiento es a la izquierda entonces saca el ultimo punto de la pila y vuelve a hacer el giro con el nuevo ultimo punto de la pila. Algo asi:

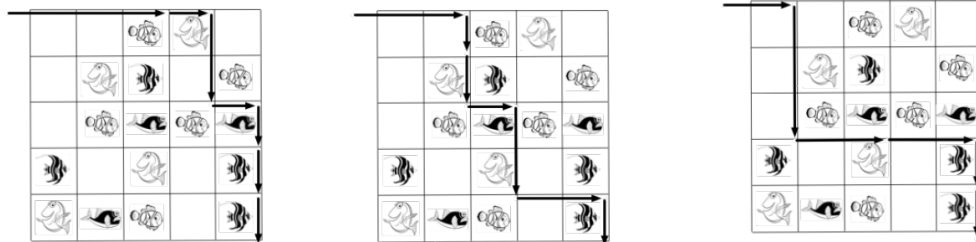


Para calcular si un giro es a la izquierda o a la derecha, se puede usar el producto vectorial, que tiene una complejidad de  $O(1)$   $((x_2 - x_1)(y_3 - y_1) - (y_2 - y_1)(x_3 - x_1))$  si es 0 entonces es colineal (este caso no pasa en posición general), si es positivo es a la izquierda y si es negativo es a la derecha).

Acabamos cuando regresamos a  $p_0$  y la pila tiene  $n$  elementos, por lo que la complejidad de Graham Scan es  $O(n)$  (porque solo procesa en la pila el elemento una vez). Por lo tanto, la complejidad total del algoritmo es  $O(n)$ .

□

6. Considera que un río fluye de norte a sur con caudal constante. Suponga que hay  $n$  ciudades en ambos lados del río, es decir  $n$  ciudades a la izquierda del río y  $n$  ciudades a la derecha. Suponga también que dichas ciudades fueron numeradas de 1 a  $n$ , pero se desconoce el orden. Construye el mayor número de puentes entre ciudades con el mismo número, tal que dos puentes no se intersecten.
7. Un pescador está sobre un océano rectangular. El valor del pez en el punto  $(i, j)$  está dado por un arreglo  $A$  de dimensión  $2 \times n \times m$ . Diseña un algoritmo que calcule el máximo valor de pescado que un pescador puede atrapar en un camino desde la esquina superior izquierda a la esquina inferior derecha. El pescador solo puede moverse hacia abajo o hacia la derecha, como se ilustra en la siguiente figura.



8. Sean tres cadenas de caracteres  $X, Y$  y  $Z$ , con  $|X| = n$ ,  $|Y| = m$  y  $|Z| = n + m$ . Diremos que  $Z$  es un *shuffle* de  $X$  y  $Y$  si  $Z$  puede ser formado por caracteres intercalados de  $X$  y  $Y$  manteniendo el orden de izquierda a derecha de cada cadena.
  - (a) Muestra que *cchocohilaptes* es un *shuffle* de *chocolate* y *chips*, pero *chocochilatspe* no lo es.
  - (b) Diseña un algoritmo de programación dinámica eficiente que determine si  $Z$  es un *shuffle* de  $X$  y  $Y$ . *Hint*: Los valores de la matriz de programación dinámica que construyas, podrían ser valores booleanos y no numéricos.

