4. Aristas Eficientes, Daniel Bustos

Sea G un digrafo con pesos positivos que tiene dos vértices especiales s y t. Para una arista $e \notin E(G)$ con peso positivo, definimos $G + e$ como el digrafo que se obtiene de agregar e a G. Decimos que e mejora el camino de s a t cuando $d_G(s, t) > d_{G+e}(s, t)$. Diseñar un algoritmo eficiente que, dado un grafo G y un conjunto de aristas $E \notin E(G)$ con pesos positivos, determine cuáles aristas de E mejoran el camino de s a t en G. Demostrar que el algoritmo es correcto.

Idea del algoritmo

Similar al ejercicio 3, usamos Dijkstra sobre s y luego sobre t en el grafo transpuesto. Luego vamos agregando las aristas una a una y nos fijamos si mejoran el camino usando que una arista vw es st-eficiente $\leftrightarrow d(s, v) + c(v, w) + d(w, t)$. Si al poner alguna arista esto mejora la mejor distancia, entonces esa arista mejora el camino.

Algoritmo

Algorithm 1 Determinar qué aristas mejoran el camino de s a t

Ejecutar Dijkstra desde s y desde t en el grafo transpuesto.
distancia_actual $\leftarrow d(s, t)$ {Obtenido del Dijkstra anterior}
for arista (v, w) **in** E **do**
    if $d(s, v) + c(v, w) + d(w, t) <$ distancia_actual **then**
        Marcar (v, w) como una arista que mejora el camino
    end if
end for
return Todas las aristas que mejoran el camino de s a t

Demostración

Como G es un digrafo con pesos positivos, sabemos por la propiedad del problema 1 que una arista ab es s-t eficiente $\leftrightarrow d(s, a) + c(a, b) + d(b, t) = d(s, t)$.

Sea G nuestro digrafo, con P un camino mínimo. Consideremos ahora $G+e$. Sabemos por la propiedad 1 que si en $G + e$, $d(s, t) = d(s, e) + c(e) + d(e, t) = d(s, t)$, entonces necesariamente forma parte de un camino mínimo. Si esto se cumple sobre $G + e$ Tenemos dos casos:

- e no cambia el camino mínimo, en cuyo caso nuestro algoritmo comparará con $d(s, t)$ en G y no tomará a e como eficiente.
- e mejora el camino mínimo anterior, luego nuestro algoritmo la marcará como eficiente.