1. Dado un digrafo D con pesos $c\colon E(D) \to \mathbb{N}$ y dos vértices s y t, decimos que una arista $v \to w$ es *st-eficiente* cuando $v \to w$ pertenece a algún camino mínimo de s a t. Sea $d(\cdot,\cdot)$ la función que indica el peso de un camino mínimo entre dos vértices.

   a. Demostrar que $v \to w$ es st-eficiente si y sólo si $d(s,v) + c(v \to w) + d(w,t) = d(s,t)$.

   b. Usando el inciso anterior, proponga un algoritmo eficiente que encuentre el mínimo de los caminos entre s y t que no use aristas st-eficientes. Si dicho camino no existe, el algoritmo retorna $\perp$.

a)

$\Leftarrow$ $d(s,t)$ es el peso minimo de un camino entre s y t. Si $d(s,t) = d(s,v) + c(v \to w) + d(w,t)$ entonces la arista $v \to w$ pertenece a un camino minimo y por def es st-eficiente.

$\Rightarrow$ Si la arista $v \to w$ es st-eficiente por def pertenece a un camino minimo entre s y t.
Esto quiere decir que existe un camino minimo entre s y t que pasa por los vèrtices v y w.
~~Por lema 22.1 (cormen): subcaminos de caminos minimos son tambien caminos minimos.~~ Pag 606
Sea $P = s \rightsquigarrow v \to w \rightsquigarrow t$ un camino minimo entre s y t.
$d(s,v) + c(v \to w) + d(w,t)$ es el costo de P, y por ser un camino minimo entre s y t, sigue que:
$d(s,v) + c(v \to w) + d(w,t) = d(s,t)$.

b) Opción 1

1) Corremos Dijkstra desde s manteniendo una lista de padres en vez de uno solo. Al revisar las aristas, si la distancia mínima se mantiene igual extendemos la lista de padres. Si mejora, la reseteamos y colocamos el nuevo padre encontrado que mejora la distancia mínima.
Esto genera un digrafo con todos los caminos mínimos desde s, lo llamamos D_s.

2) Recorremos todos los caminos en reversa en D_s partiendo desde t. Durante este proceso eliminamos todas las aristas st-eficientes en D.

3) Corremos nuevamente Dijkstra en D desde s. Como borramos todas las aristas st-eficientes, si Dijkstra encuentra un camino mínimo, este no contiene aristas st-eficientes. Si no hay camino mínimo retornamos $\perp$.

b) Opción 2 (mejor)

1) Corremos Dijkstra desde s para calcular $d(s,v)$ $\forall v \in V(D)$.

2) invertimos todas las aristas y volvemos a correr Dijkstra pero esta vez desde t. Así calculamos $d(t,v) = d(v,t)$ $\forall v \in V(D)$.

3) Iteramos por todas las aristas de D y verificamos si son st-eficientes, utilizando las distancias mínimas previamente calculadas.
$v \to w$ st-eficiente $\iff d(s,v) + c(v,w) + d(w,t) = d(s,t)$.
Si $v \to w$ es st-eficiente la marcamos como tal con un booleano.

4) Corremos Dijkstra una última vez desde s, pero ahora ignorando todas las aristas marcadas como st-eficiente. Si hay camino mínimo retornamos su costo, sino $\bot$.