Ελάχιστα διασυνδετικά δένδρα: Ο αλγόριθμος του Kruskal

Είσοδος: Ένα συνδεδεμένο μη κατευθυνόμενο γράφημα $G = (V, E)$ με μία συνάρτηση βάρους $w : E \to \mathcal{R}$.

Έξοδος: Ένα ελάχιστο διασυνδετικό δένδρο για το G.

- Υπενθυμίζεται ο **γενικός** αλγόριθμος:

Generic_MST(G, w)
$A = \emptyset$
while A does not form a spanning tree **do**
    Find an edge (u, v) that is safe for A.
    $A = A \cup \{(u, v)\}$
return A

Σταθερά (invariant) του αλγορίθμου: "Το A είναι πάντοτε ένα υποσύνολο κάποιου ελάχιστου διασυνδετικού δένδρου".

- Μια ακμή (u, v) είναι μία *ασφαλής ακμή* για το A εάν μπορεί να προστεθεί στο A χωρίς να διαταραχθεί η σταθερά του αλγορίθμου.

MST_Kruskal(G, w)
1. $A = \emptyset$
2. **for** each vertex $v \in V(G)$
3. **do** $MAKE\_SET(v)$
4. Sort the the edges of E in non-decreasing weight w.
5. **for** each edge $(u, v) \in E$, in order by non-decreasing weight,
6. **do if** $FIND\_SET(u) \neq FIND\_SET(v)$
7. **then** $A = A \cup \{(u, v)\}$
8. $UNION(u, v)$
9. **return** A

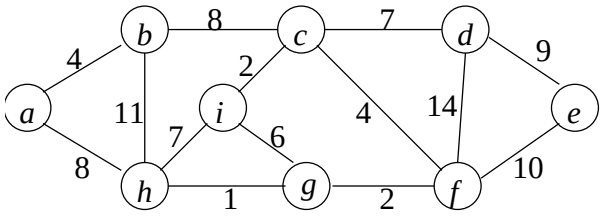
- Χρησιμοποιήσαμε μια ΑΔΔ που υποστηρίζει πράξεις σε ξένα μεταξύ τους (disjoint) σύνολα.
  - $MAKE\_SET(v)$: Δημιουργεί ένα νέο σύνολο που περιέχει το στοιχείο v.
  - $FIND\_SET(v)$: Επιστρέφει το σύνολο στο οποίο ανήκει το v.
  - $UNION(u, v)$: Δημιουργεί ένα νέο σύνολο που είναι η ένωση των: $FIND\_SET(u)$ και $FIND\_SET(v)$.

- **Ανάλυση** Βήματα 2-3: $O(V)$
  Βήμα 4: $O(E \log E)$
  Βήματα 5-8: $O(E + V \log V)$
  ΣΥΝΟΛΙΚΑ: $O(E \log E)$

Παράδειγμα

Το E είναι ταξινομημένο σε αύξουσα τάξη:

(h, g)	1
(i, c)	2
(g, f)	2
(a, b)	4
(c, f)	4
(i, g)	6
(c, d)	7
(h, i)	7
(a, h)	8
(b, c)	8
(d, e)	9
(f, e)	10
(b, h)	11
(d, f)	14



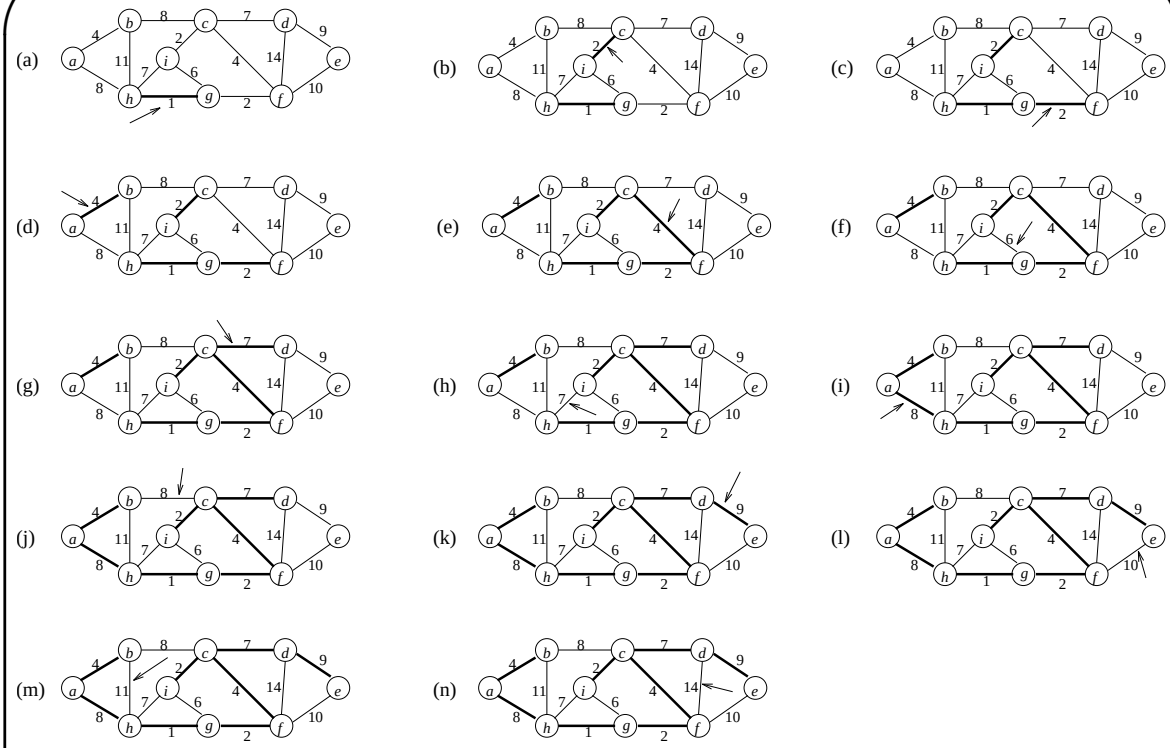
(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

(m) (n)

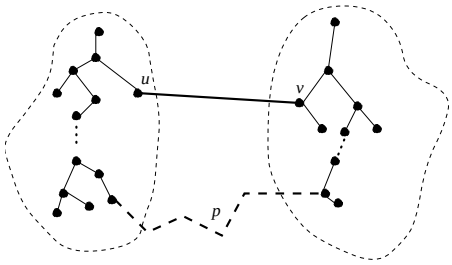


Απόδειξη της ορθότητας του αλγορίθμου του Kruskal

Θεώρημα Εάν αποφασίσουμε να προσθέσουμε την ακμή (u, v) στο A, τότε η (u, v) είναι ασφαλής ακμή.

Απόδειξη Προσθέτουμε την ακμή (u, v) στο A εάν και μόνο εάν οι u και v ανήκουν σε διαφορετικά δένδρα του G (που έχουν ήδη σχηματιστεί από την εκτέλεση του αλγορίθμου του Kruskal).

- Εάν η (u, v) δεν είναι ασφαλής, τότε δεν ανήκει στο ελάχιστο διασυνδετικό δένδρο. Άρα, υπάρχει ένα μονοπάτι p από το δένδρο που περιέχει την u προς το δένδρο που περιέχει την v το οποίο ανήκει στο ελάχιστο διασυνδετικό δένδρο του G.



- Αφού το μονοπάτι p αποτελείται από ακμές με βάρη μεγαλύτερα ή ίσα του $w(u, v)$, μπορούμε να αντικαταστήσουμε οποιαδήποτε ακμή του p με την (u, v) και να λάβουμε ένα ελάχιστο διασυνδετικό δένδρο με το ίδιο βάρος του "υποθετικού "ελάχιστου διασυνδετικού δένδρου του G.

Άρα, η ακμή (u, v) είναι ασφαλής.

□

Ελάχιστα μονοπάτια από κοινή αφετηρία

Είσοδος: • Ένα κατευθυνόμενο, με βάρη, γράφημα $G = (V, E)$ και μία συνάρτηση βάρους $w : E \to \mathcal{R}$ που αντιστοιχεί σε κάθε ακμή ένα πραγματικό αριθμό (βάρος).
• Ένας κόμβος αφετηρία s.

Έξοδος: Τα ελάχιστα μονοπάτια από το s προς κάθε άλλο κόμβο του G.

Επανεκτίμηση - Relaxation

- Για κάθε κορυφή $v \in V$, το $d[v]$ είναι ένα άνω φράγμα για το βάρος του ελάχιστου μονοπατιού από την αφετηρίαs στην v.

  $d[v] = \text{shortest\_path\_estimate}$

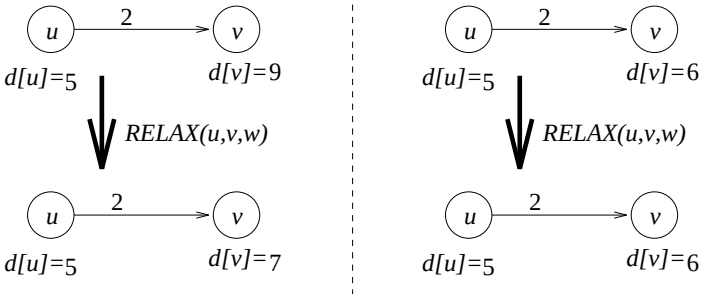
- Η αρχικοποίηση του $d[v],\ v \in V$, γίνεται ως εξής:

  INITIALIZE_SINGLE_SOURCE(G, s)
  for each vertex $v \in V$
  do $d[v] \leftarrow +\infty$
  $\pi[v] \leftarrow NIL$
  $d[s] \leftarrow 0$

Κατά *την επανεκτίμηση της ακμής* (u, v), ελέγχουμε εάν μπορούμε να βελτιώσουμε το ελάχιστο μονοπάτι από τον s στον v πηγαίνοντας μέσω της ακμής (u, v).

RELAX(u, v, w)
if $d[v] > d[u] + w(u, v)$
  then $d[v] \leftarrow d[u] + w(u, v)$
    $\pi[v] \leftarrow u$

Παράδειγμα



- Αμέσως μετά την επανεκτίμηση της ακμής (u, v) μέσω της $RELAX(u, v, w)$, ισχύει: $d[v] \leq d[u] + w(u, v)$.

- Ο αλγόριθμος του Dijkstra μπορεί να γραφεί και ως:

Dijkstra(G, w, s)
1. ***INITIALIZE_SINGLE_SOURCE***(G, s)
2. $S \leftarrow \emptyset$
3. $Q \leftarrow V[G]$
4. **while** $Q \neq \emptyset$
5. **do** $u = extract\_min(Q)$
6. $S \leftarrow S \cup \{u\}$
7. **for** each vertex $v \in Adj[u]$
8. **do** ***RELAX***(u, v, w)

Παρατήρηση 1: Στον αλγόριθμος του Dijkstra, κάθε ακμή επανεκτιμάτε *ακριβώς μία φορά.*

Παρατήρηση 2: Ο αλγόριθμος του Dijkstra δουλεύει σωστά μόνο για γραφήματα με μη αρνητικά βάρη στις ακμές.

Ερώτημα: Πώς αντιμετωπίζουμε τα αρνητικά βάρη στις ακμές;

Ο αλγόριθμος των BELLMAN-FORD

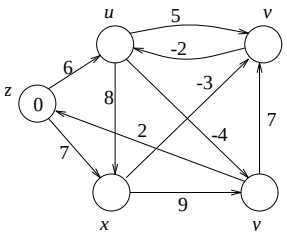
Είσοδος:
- Ένα κατευθυνόμενο, με βάρη, γράφημα $G = (V, E)$ και μία συνάρτηση βάρους $w : E \to \mathcal{R}$ που αντιστοιχεί σε κάθε ακμή ένα πραγματικό αριθμό (βάρος).
- Ένας κόμβος αφετηρία s.

Έξοδος:
- Μία τιμή τύπου boolean που θα δηλώνει την ύπαρξη κύκλου αρνητικού βάρους προσπελάσιμου από την αφετηρία s.
- Εάν δεν υπάρχουν αρνητικοί κύκλοι, τα ελάχιστα μονοπάτια (και τα βάρη τους) από τον s προς κάθε άλλο κόμβο του G.

BELLMAN_FORD(G, w, s)
1. ***INITIALIZE_SINGLE_SOURCE***(G, s)
2. **for** $i = 1$ **to** $|V(G)| - 1$
3. **do for** each edge $(u, v) \in E(G)$
4. **do** ***RELAX***(u, v, w)
5. **for** each edge $(u, v) \in E(G)$
6. **do if** $d[v] > d[u] + w(u, v)$
7. **then return** $FALSE$
8. **return** $TRUE$

Πολυπλοκότητα $O(VE)$ (Κάθε ακμή επανεκτιμάται ακριβώς $|V| - 1$ φορές.)

Παράδειγμα



Επανεκτιμούμε τις κορυφές με την ακόλουθη σειρά:

(z, u)
(z, x)
(u, v)
(u, y)
(u, x)
(v, u)
(x, v)
(x, y)
(y, z)
(y, v)

- Η σειρά της επανεκτίμησης των κόμβων είναι αυθαίρετη. Σε κάθε "πέρασμα"μπορεί να επανεκτιμήσουμε τις ακμές σε διαφορετική σειρά.

Απόδειξη της ορθότητας του αλγορίθμου των BELLMAN-FORD

- Έστω v ένας κόμβος προσπελάσιμος από την αφετηρία s.
- Έστω $p = \langle s, v_1, v_2, \ldots, v_{k-1}, v \rangle$ το ελάχιστο μονοπάτι μήκους k από την s στον v, $k \leq |V| - 1$.
- Θα αποδείξουμε με επαγωγή ότι $d[v_i] = \delta(s, v_i)$ ύστερα από το i-οστό "πέρασμα" των ακμών.

Βάση $d[s] = 0$.

Επαγωγικό βήμα

◇ **Υποθέτουμε** ότι $d[v_{i-1}] = \delta(s, v_{i-1})$ ύστερά από το $(i-1)$-οστό πέρασμα.

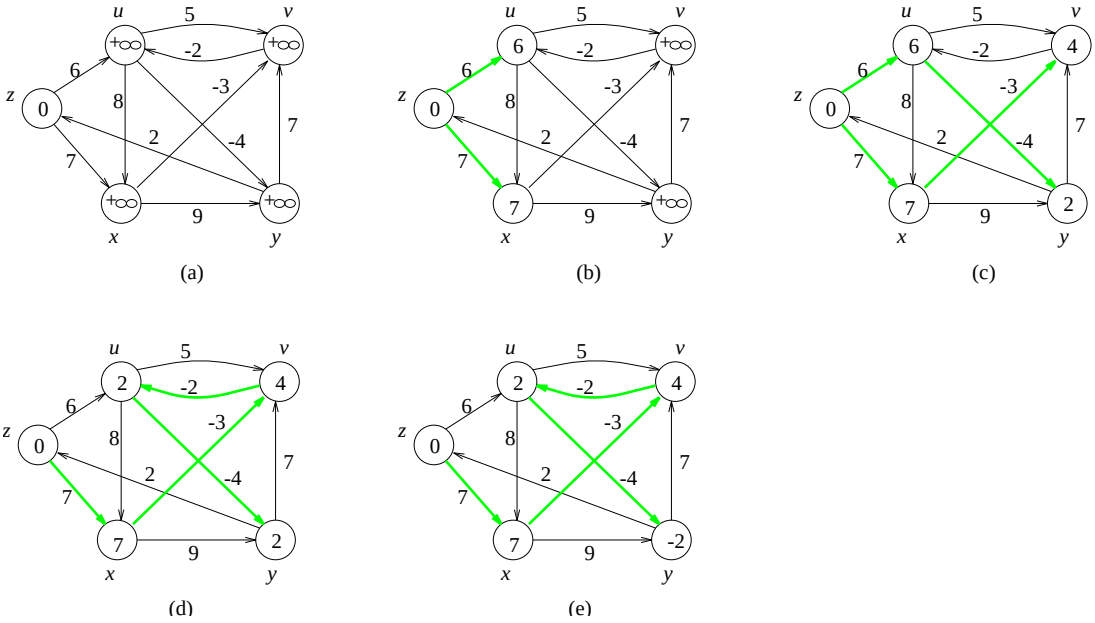
◇ **Θα αποδείξουμε** ότι $d[v_i] = \delta(s, v_i)$ ύστερα από το i-οστό πέρασμα.

Αυτό έπεται από τα παρακάτω:

1. Επανεκτιμούμε την ακμή (v_{i-1}, v_i) κατά το i-οστό πέρασμα.
2. Τα υπό - μονοπάτια των ελάχιστων μονοπατιών είναι ελάχιστα μονοπάτια.

□

Παράδειγμα Οι ακμές επανεκτιμούνται με λεξικογραφική σειρά.



Ελάχιστα μονοπάτια από κοινής αφετηρίας σε DAGs

(DAGs: Κατευθυνόμενα ακυκλικά γραφήματα)

DAGs_Shortest_Paths(G, w, s)

1. Topologically sort the vertices of G
2. ***INITIALIZE_SINGLE_SOURCE***(G, s)
3. **for** each vertex u (in the topologically sorted order)
4. **do for** each vertex $v \in Adj[u]$
5. **do** $RELAX(u, v)$

Πολυπλοκότητα: $O(V + E)$