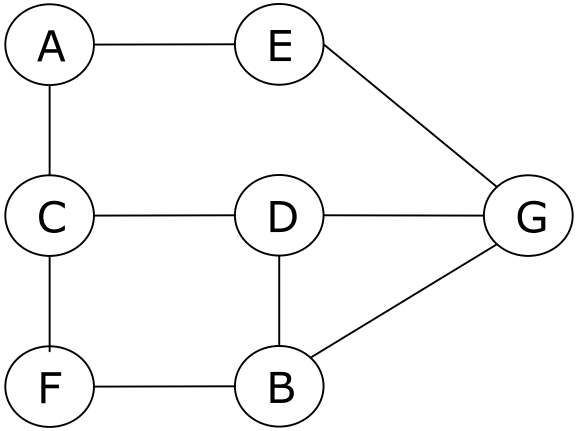
Γ. Κορίλη “Αλγόριθμοι Δρομολόγησης”

http://www.seas.upenn.edu/~tcom501/Lectures/Lecture11.pdf

- Δρομολόγηση σε Δίκτυα Δεδομένων
- Αναπαράσταση Δικτύου με Γράφο
- Μη Κατευθυνόμενοι Γράφοι
- Εκτεταμένα Δέντρα
- Κατευθυνόμενοι Γράφοι
- Αλγόριθμοι Ελαχίστου Δρόμου:
  - Bellman-Ford
  - Dijkstra
  - Floyd-Warshall
- Κατανεμημένος Ασύγχρονος Αλγόριθμος Bellman-Ford

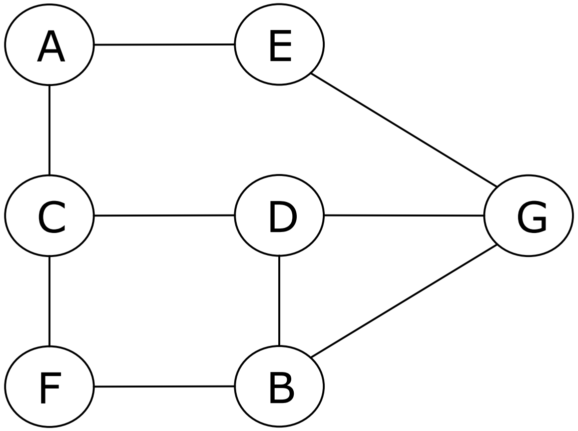
Εισαγωγή στη Δρομολόγηση

- Τι είναι δρομολόγηση;
  - Η δημιουργία πληροφοριών (για τις καταστάσεις) στο δίκτυο που θα επιτρέψει την *αποτελεσματική παράδοση* των πακέτων στους επιθυμητούς προορισμούς τους
- Δυο κύρια στοιχεία
  - Απόκτηση πληροφοριών: Τοπολογία, διευθύνσεις
  - Χρήση πληροφοριών: Υπολογισμός “καλών” δρόμων προς όλους τους προορισμούς
- Ερωτήματα
  - Πού είναι το B;
  - Πώς θα φθάσουμε στο B;
  - Πώς θα φθάσουμε *καλύτερα* στο B;
  - Πώς θα κατανέμουμε καλύτερα όλη την κυκλοφορία (όχι μόνο από το A στο B);



Έννοιες Θεωρίας Γράφων

- Ένας Μη Κατευθυνόμενος Γράφος G = (N, A) αποτελείται από:
  - Ένα πεπερασμένο μη κενό σύνολο κόμβων N και
  - Μια συλλογή “πλευρών” ή “συνδέσεων” A, που συνδέουν ζευγάρια (διακριτών) κόμβων του N.
- Αν i και j είναι κόμβοι στο N και (i, j) πλευρά στο in A, η πλευρά αυτή λέγεται ‘προσπίπτουσα’ (incident) στα i και j
- Διαδρομή: μια ακολουθία κόμβων (n_1, n_2, …, n_k), όπου (n_1, n_2), (n_2, n_3), …, (n_{k-1}, n_k) είναι πλευρές
- Δρόμος: μια διαδρομή χωρίς επαναλαμβανόμενους κόμβους
- Κύκλος: μια διαδρομή (n_1, n_2, …, n_k) με n_1=n_k και χωρίς άλλους επαναλαμβανόμενους κόμβους
- Συνεκτικός Γράφος: για κάθε $i, j \in$ N, υπάρχει δρόμος (n_1, n_2, …, n_k) με i=n_1, j=n_k

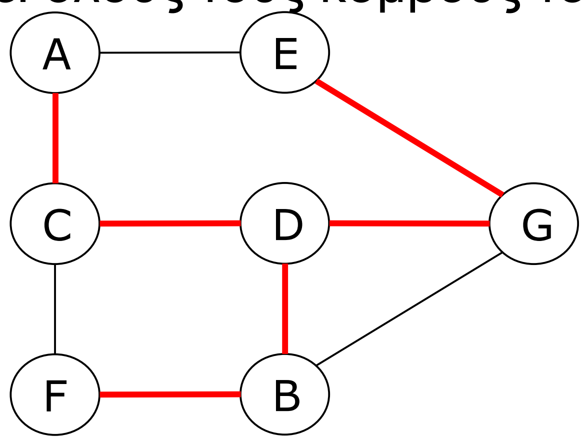


$$\mathcal{N} = \{A, B, C, D, E, F, G\}$$

$$\mathcal{A} = \{(A,E),(A,C),(C,D),(C,F),$$
$$(B,D),(B,G),(E,g)\}$$

Εκτεταμένα Δέντρα (Spanning Trees)

- Ένας γράφος G' = (N′, A), με $N' \subseteq N$ και $A' \subseteq A$ ονομάζεται υπο-γράφος του G = (N, A)
- Δέντρο: συνεκτικός γράφος που δεν περιέχει κύκλους
- Εκτεταμένο δένδρο (spanning tree) του γράφου G: υπο-γράφος του G, που περιλαμβάνει όλους τους κόμβους του G (N′ = N)



- <u>Λήμμα:</u> Έστω ο συνεκτικός γράφος G = (N, A) και S ένα μη κενό γνήσιο υποσύνολο του σύνολο των κόμβων N. Τότε, υπάρχει τουλάχιστον μια πλευρά (i, j) τέτοια ώστε $i \in S$ και $j \notin S$.

Αλγόριθμος Εκτεταμένου Δέντρου

1. Επιλέξτε αυθαίρετα έναν κόμβο $n \in N$ και αρχίστε με $G' = (N', A')$

$$\mathcal{N}' = \{n\}, \ \ \mathcal{A}' = \emptyset$$

2. Αν $N' = N$, ΤΕΛΟΣ: $G' = (N', A')$ είναι ένα εκτεταμένο δέντρο
ΑΛΛΙΩΣ: πάτε στο βήμα 3
3. Έστω $(i, j) \in A$ με $i \in N'$ και $j \in N - N'$
   - Αλλάξτε:

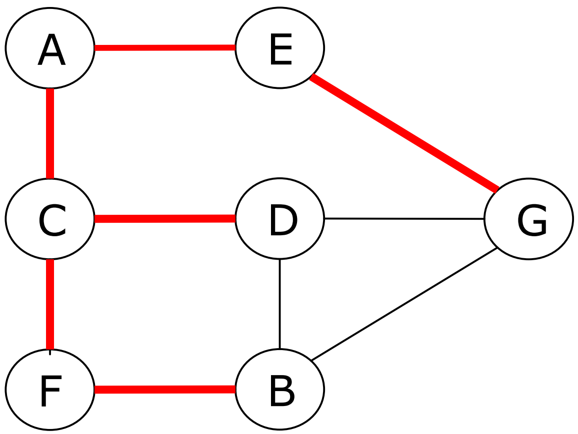
     $$\mathcal{N}' := \mathcal{N}' \cup \{j\}, \ \ \mathcal{A}' := \mathcal{A}' \cup \{(i, j)\}$$

   - Πάτε στο βήμα 2

- <u>Απόδειξη:</u> Δείξτε με επαγωγή ότι μετά την προσθήκη ενός νέου κόμβου i, ο *G* παραμένει συνεκτικός και δεν περιέχει κανένα κύκλο – δηλαδή, είναι δένδρο.

Κατασκευή Εκτεταμένου Δέντρου

- $V' = \{A\}$; $E' = \varnothing$
- $V' = \{A,E\}$; $E' = \{(A,E)\}$
- $V' = \{A,E,C\}$;
  $E' = \{(A,E),(A,C)\}$
- $V' = \{A,E,C,D\}$;
  $E' = \{(A,E),(A,C),(CD)\}$
- $V' = \{A,E,C,D,F\}$;
  $E' = \{(A,E),(A,C),(CD),(CF)\}$
- $V' = \{A,E,C,D,F,B\}$;
  $E' = \{(A,E),(A,C),(CD),(CF),(F,B)\}$
- $V' = \{A,E,C,D,F,B,G\}$;
  $E' =$
  $\{(A,E),(A,C),(CD),(CF),(F,B),(E,G)\}$



Εκτεταμένα Δέντρα (συνέχεια)

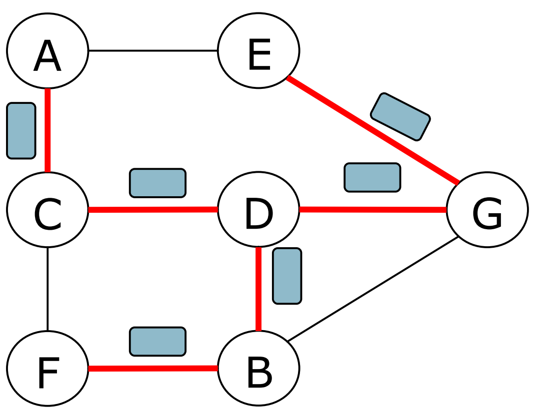
- **Πρόταση:** Έστω G συνεκτικός γράφος με N κόμβους και A συνδέσεις. Τότε:

1. G περιέχει ένα εκτεταμένο δέντρο
2. $A \geq N-1$
3. G είναι δέντρο αν και μόνον αν A=N-1

- **Απόδειξη:** Ο αλγόριθμος κατασκευής του εκτεταμένου δέντρου αρχίζει με ένα μόνο κόμβο και σε κάθε επανάληψη αυξάνει το δέντρο κατά έναν κόμβο και μια πλευρά. Επομένως, το δέντρο κατασκευάζεται μετά από N-1 επαναλήψεις κι έχει N-1 συνδέσεις, δηλαδή, $A \geq N-1$.
  - Αν A=N-1, το εκτεταμένο δέντρο περιέχει όλες τις πλευρές του G, δηλαδή, G είναι δέντρο.
  - Αν A>N-1, υπάρχει μια σύνδεση (i, j) που δεν ανήκει στο δέντρο. Θεωρώντας το δρόμο που συνδέει τα i και j μέσα στο εκτεταμένο δέντρο, ο δρόμος αυτός (i, j) σχηματίζει κύκλο κι, έτσι, το G δεν είναι δέντρο.

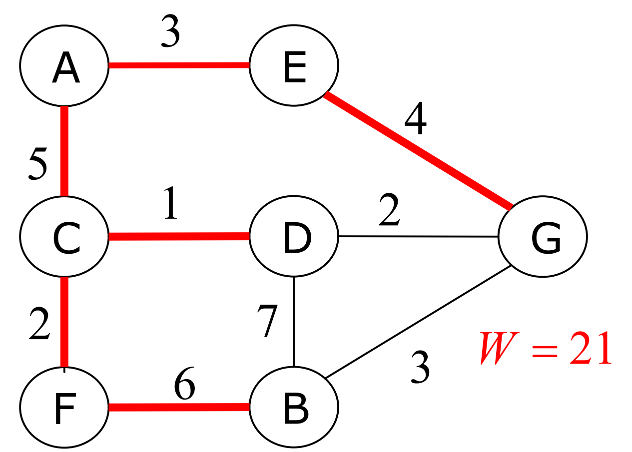
Η Χρήση των Εκτεταμένων Δέντρων

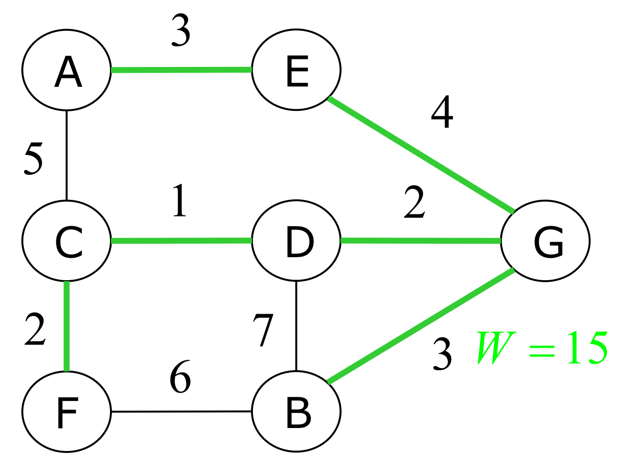
- Πρόβλημα: πώς να διανέμουμε τις πληροφορίες σε *όλους* τους κόμβους σ’ ένα γράφο (δίκτυο) – π.χ., πληροφορίες για διευθύνσεις και τοπολογία
- Πλημμύρα: όταν κάθε κόμβος προωθεί τις πληροφορίες προς όλους τους γείτονές του.
- Εκτεταμένο δέντρο: οι κόμβοι προωθούν τις πληροφορίες μόνο κατά την κατεύθυνση των συνδέσεων που ανήκουν στο εκτεταμένο δέντρο. Πιο αποτελεσματικά:
  - Οι πληροφορίες φθάνουν σε κάθε κόμβο μόνο μια φορά και διασχίζουν κάθε σύνδεση μόνο μια φορά
  - Παρατηρήστε ότι το εκτεταμένο δέντρο είναι *διπλής κατεύθυνσης*



Εκτεταμένα Δέντρα Ελάχιστου Βάρους

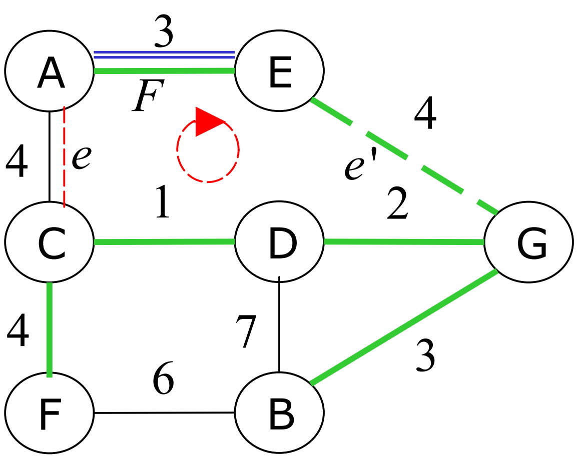
- Το βάρος w_{ij} σε μια πλευρά χρησιμοποιείται για να δώσει το “κόστος” χρήσης της σύνδεσης (i, j)
- Παραδείγματα: καθυστέρηση, φορτίο, απόσταση, κλπ.
- Το βάρος (κόστος) ενός δέντρου είναι το άθροισμα των βαρών όλων των συνδέσεών του (τα πακέτα διασχίζουν τις συνδέσεις μόνο μια φορά)
- Ορισμός: Ένα *Εκτεταμένο Δέντρο Ελάχιστου Βάρους* ή *Ελάχιστο Εκτεταμένο Δέντρο (ΕΕΔ)* είναι ένα εκτεταμένο δέντρο με ελάχιστο άθροισμα βαρών των συνδέσεων
- Ορισμός: Ένα υπο-δέντρο ενός ΕΕΔ ονομάζεται *τεμάχιο*. Μια πλευρά που έχει έναν κόμβο σε κάποιο τεμάχιο και τον άλλο κόμβο όχι σ’ αυτό το τεμάχιο ονομάζεται *εξερχόμενη πλευρά* από το τεμάχιο.





Ελάχιστα Εκτεταμένα Δέντρα

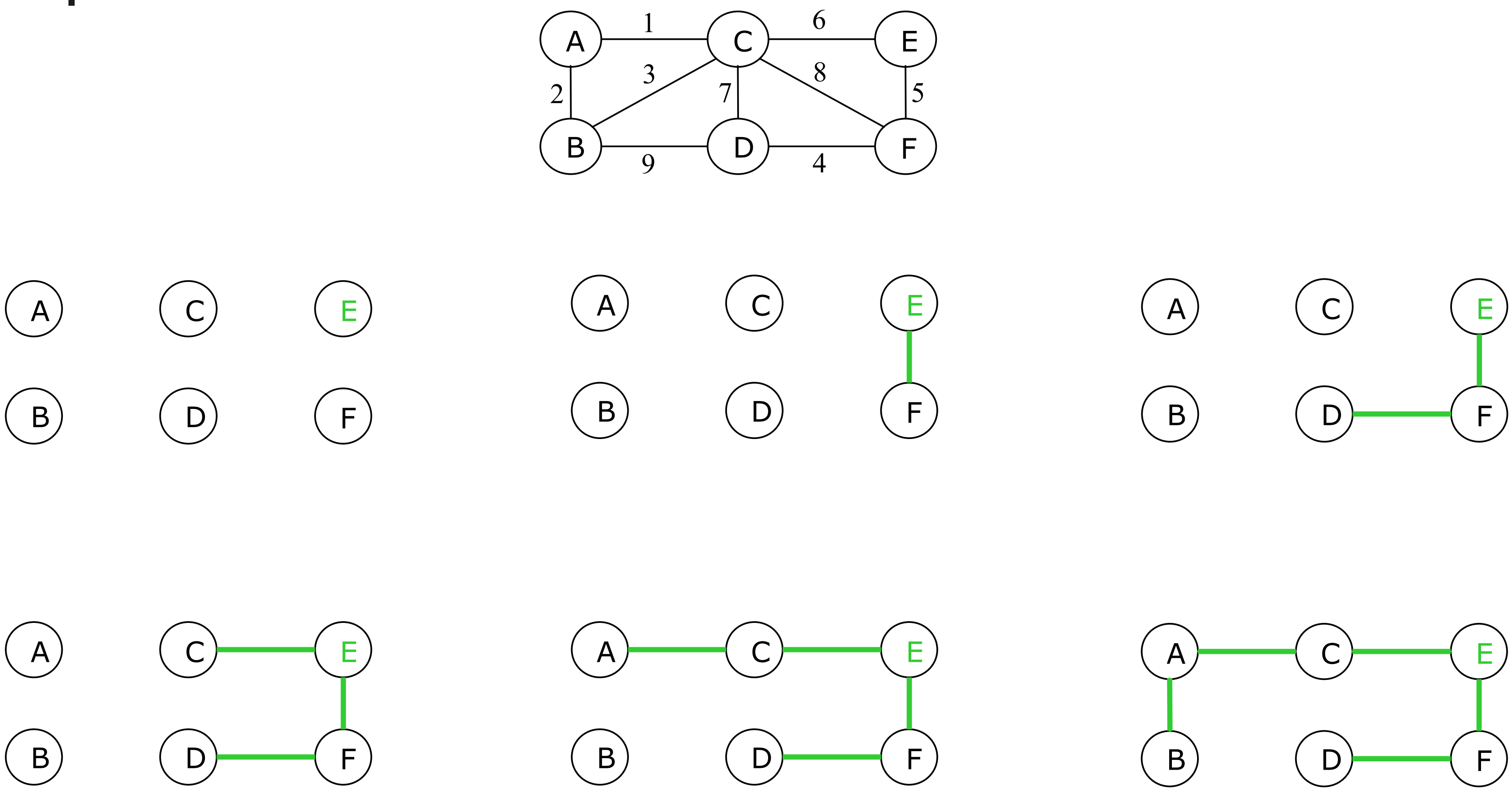
- <u>Λήμμα:</u> Δοθέντος ενός τεμαχίου *F*, έστω *e*=(*i*, *j*) μια εξερχόμενη πλευρά ελάχιστου βάρους, όπου $j \notin F$. Τότε το *F* επεκτεταμένο με την πλευρά *e* και τον κόμβο *j* είναι ένα τεμάχιο.
- <u>Απόδειξη:</u>
- Έστω *T* το ΕΕΔ που περιέχει το τεμάχιο *F*. Αν $e \in T$, τελειώσαμε.
- Έστω $e \notin T$: τότε σχηματίζεται ένας κύκλος από την *e* και τις πλευρές του *T*
  - Αφού $j \notin F$. Υπάρχει πλευρά $e' \neq e$ που ανήκει στον κύκλο και το *T* και είναι εξερχόμενη από το *F*.
  - Έστω $T' = (T - \{e'\}) \cup \{e\}$. Αυτός είναι υπο-γράφος με N-1 πλευρές και χωρίς κύκλους, δηλαδή, εκτεταμένο δέντρο.
  - Επειδή $w_e \leq w_{e'}$, το βάρος του *T'* είναι μικρότερο ή ίσο του βάρους του *T*
  - Τότε *T'* είναι ένα ΕΕΔ και το *F* επεκτεταμένο από την πλευρά *e* και τον κόμβο *j* είναι τεμάχιο.



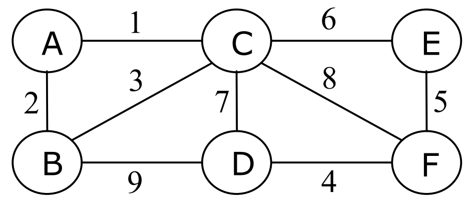
Αλγόριθμοι Ελάχιστων Εκτεταμένων Δέντρων

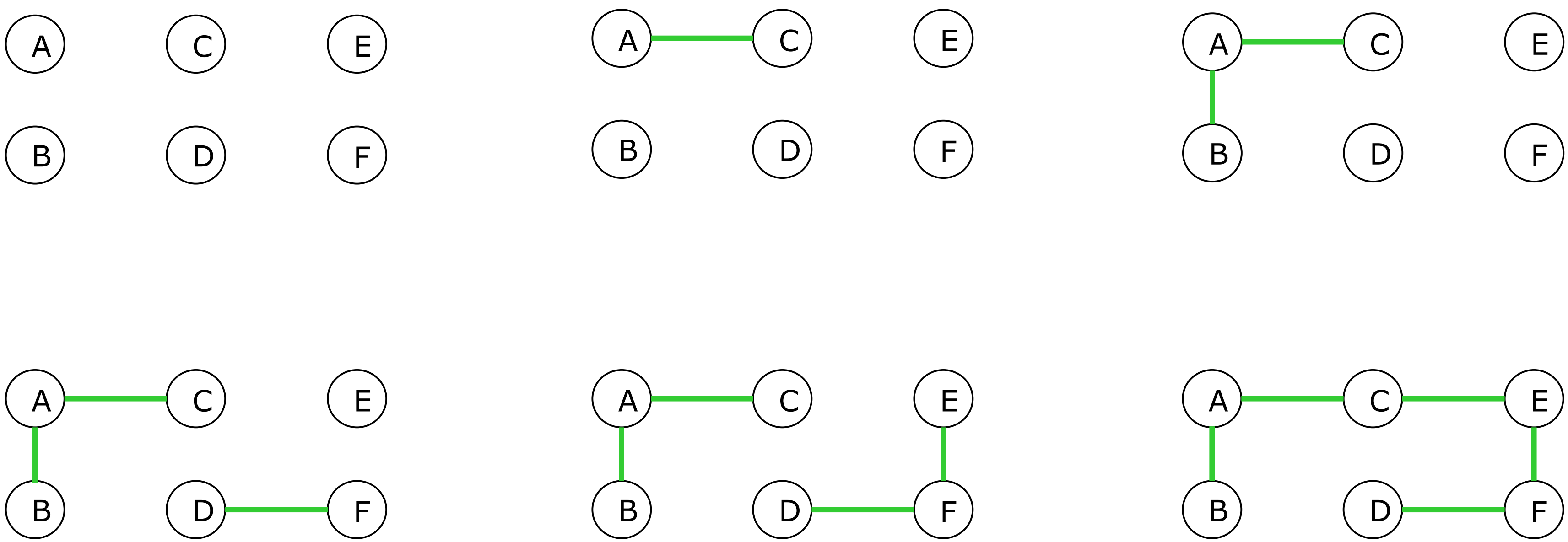
- Επαγωγικοί αλγόριθμοι που βασίζονται στο προηγούμενο λήμμα
- Ο Αλγόριθμος του Prim:
  - Ξεκινήστε με έναν τυχαίο κόμβο ως το αρχικό τεμάχιο
  - Επεκτείνετε το τεμάχιο με τη διαδοχική πρόσθεση εξερχόμενων πλευρών ελάχιστου βάρους
- Ο Αλγόριθμος του Kruskal:
  - Όλες οι κορυφές είναι τα αρχικά τεμάχια
  - Διαδοχικά συνδυάστε τα τεμάχια χρησιμοποιώντας πλευρές ελάχιστου βάρους που δεν δημιουργούν κύκλους

Αλγόριθμος του Prim



Ο Αλγόριθμος του Kruskal





Αλγόριθμοι Ελάχιστου Δρόμου

- **Πρόβλημα:** Δοθέντων των κόμβων A και B, βρείτε την “καλύτερη” διαδρομή για να περάσει η κυκλοφορία από το A στο B
- “Καλύτερη:” ελάχιστου κόστους – όπου τυπικά το κόστος μιας διαδρομής ισούται προς το άθροισμα των κοστών των συνδέσεων της διαδρομής
- Σημαντικό πρόβλημα για διάφορες δικτυακές εφαρμογές
- Δρομολόγηση της κυκλοφορίας σε δικτυακές συνδέσεις $\Rightarrow$ χρειάζεται να λάβουμε υπόψη την κατεύθυνση της ροής
- Κατάλληλο δικτυακό μοντέλο: Κατευθυνόμενος Γράφος

Κατευθυνόμενοι Γράφοι

- Ένας *Κατευθυνόμενος Γράφος* (ή Δι-γραφος) G = (N, A) αποτελείται από:
  - Ένα πεπερασμένο μη κενό σύνολο κόμβων N και
  - Μια συλλογή “τόξων” A, δηλαδή, διατεταγμένων ζευγαριών (διακριτών) κόμβων του N.
- Κατευθυνόμενες διαδρομές, κατευθυνόμενοι δρόμοι και κατευθυνόμενοι κύκλοι ορίζονται όπως (αντίστοιχα) με τους μη κατευθυνόμενους γράφους
- Δοθέντος του κατευθυνόμενου γράφου G = (N, A), υπάρχει ένας αντίστοιχος μη κατευθυνόμενος γράφος G' = (N′, A′), με N′=N και $(i, j)\in$ A′ αν είτε $(i, j)\in$ A ή $(j, i)\in$ A
- Ένας κατευθυνόμενος γράφος G = (N, A) ονομάζεται συνεκτικός αν ο αντίστοιχος μη κατευθυνόμενος γράφος G' = (N ′, A′) είναι συνεκτικός
- Ένας κατευθυνόμενος γράφος G = (N, A) ονομάζεται *ισχυρά συνεκτικός* αν, για κάθε $i, j\in$ N, υπάρχει ένας κατευθυνόμενος δρόμος $(n_1, n_2, \ldots, n_k)$, με $i=n_1$, $j=n_k$

Αλγόριθμοι Ελάχιστου Δρόμου: Γενική Διατύπωση

- Έστω ο γράφος $G = (V,E)$ με N κόμβους και με μήκη πλευρών d_{ij} για την πλευρά (i,j) (θεωρούμε $d_{ij} = \infty$ αν $(i,j) \notin E$)
- Πρόβλημα: Βρείτε τους δρόμους ελάχιστης απόστασης από όλους τους κόμβους του V προς τον κόμβο 1
  - Αλλιώς διατυπωμένο, βρείτε τους δρόμους ελάχιστης απόστασης από τον κόμβο 1 προς όλους τους κόμβους του V
- Πάλι η γενική προσέγγιση θα είναι *επαναληπτική*:

$$D_i^{(n+1)} = \min\{D_j^{(n)} + d_{ij}\}$$

  - Διαφορές για το πώς προχωρούμε στις επαναλήψεις κι έτσι υπάρχουν
  - Τρεις κύριοι αλγόριθμοι

Ο Αλγόριθμος Bellman-Ford

- Σε κάθε επαναληπτικό βήμα αυξάνει το πλήθος των πλευρών κατά ένα
- Ορίζουμε το D_i^h σαν το μήκος της ελάχιστης διαδρομής από το i στο 1 που περιέχει το πολύ h πλευρές
  - $D_1^h = 0$, εξ ορισμού, για κάθε h
- Αλγόριθμος Bellman-Ford:
  - Ορίζουμε
$$D_i^{h+1} = \min_j \{D_j^h + d_{ij}\}, \ \forall i \neq 1$$
  - Θέτουμε αρχικά $D_i^0 = \infty$ για κάθε $i \neq 1$

  a Τα D_i^h είναι τα μήκη των ελάχιστων διαδρομών κατά μήκος το πολύ h πλευρών από το i στο 1

  b Ο αλγόριθμος τερματίζει μετά από πεπερασμένο πλήθος επαναλήψεων αν και μόνον αν όλοι οι κύκλοι, που δεν περιέχουν τον 1, έχουν μη αρνητικό μήκος. Επιπλέον, αν ο αλγόριθμος τερματίσει, το κάνει σε N το πολύ επαναλήψεις και τερματίζει στο μήκος του ελάχιστου δρόμου από το i στο 1.

Απόδειξη του Αλγόριθμου Bellman-Ford (1)

- Με επαγωγή στο πλήθος των πλευρών *h*
  - Για *h* = 1, έχουμε
    - $D_i^1 = d_{i1}$, για κάθε $i \neq 1$, οπότε ισχύει το ζητούμενο, για $i = 1$
  - Υποθέτουμε ότι αυτό ισχύει για όλα τα $k \leq h$. Πρέπει να δείξουμε ότι ισχύει και για h+1
- Αρκεί να εξετασθούν δυο περιπτώσεις
  1. Η ελάχιστη ($\leq h$+1) διαδρομή από το *i* στο 1 έχει $\leq h$ πλευρές
     - Τότε το μήκος της είναι D_i^h
  2. Η ελάχιστη ($\leq h$+1) διαδρομή από το *i* στο 1 έχει (h+1) πλευρές
     - Αποτελείται από τη συνένωση της πλευράς (*i*,*j*) με την ελάχιστη διαδρομή σε *h* πλευρές από το *j* στο 1
  - Η δεύτερη περίπτωση είναι αυτή που θα εστιασθούμε

Απόδειξη του Αλγόριθμου Bellman-Ford (2)

- Από τις Περιπτώσεις 1 και 2, έχουμε

$$\text{shortest } (\leq h+1) \text{ walk length} = \min\left\{D_i^h, \min_{j\neq 1}\left[D_j^h + d_{ij}\right]\right\}$$

- Από την επαγωγική υπόθεση και τις αρχικές συνθήκες
  - $D_i^k \leq D_i^{k-1}$ για όλα τα $k \leq h$ έτσι ώστε

$$D_i^{h+1} = \min_j\left[D_j^h + d_{ij}\right] \leq \min_j\left[D_j^{h-1} + d_{ij}\right] = D_i^h$$

  - $D_i^h \leq D_i^1 = d_{i1} = d_{i1} + D_1^h$
- Συνδυάζοντας τα παραπάνω παίρνουμε

$$\text{shortest } (\leq h+1) \text{ walk length} = \min\left\{D_i^h, \min_j\left[D_j^h + d_{ij}\right]\right\}$$

$$= \min\left\{D_i^h, D_i^{h+1}\right\} = D_i^{h+1}$$

Που ολοκληρώνει την απόδειξη του μέρους a

Απόδειξη του Αλγόριθμου Bellman-Ford (3)

- Υποθέτουμε ότι ο αλγόριθμος τερματίζει μετά από h βήματα
  - Αυτό συνεπάγεται ότι $D_i^k = D_i^h$ για όλα τα i και $k \geq h$
  - Άρα με την προσθήκη περισσότερων πλευρών δεν μπορούν να μειωθούν τα μήκη αυτών των διαδρομών
  - Επομένως, δεν μπορούν να υπάρξουν κύκλοι αρνητικού μήκους που δεν περιέχουν τον κόμβο 1
    - Αυτοί θα μπορούσαν να επαναλαμβάνονταν πολλές φορές για να έκαναν τα μήκη των διαδρομών αυθαίρετα μικρά
  - Αφαιρούμε όλους αυτούς τους κύκλους
    - Τότε παίρνουμε δρόμους μικρότερου ή ίσου μήκους
    - Έτσι, για όλα τα i έχουμε ένα δρόμο h ή λιγότερων πλευρών προς τον κόμβο 1 με μήκος D_i^h
    - Αφού αυτοί οι δρόμοι δεν περιέχουν κύκλους, περιλαμβάνουν το πολύ N-1 πλευρές κι, άρα, $D_i^N = D_i^{N-1}$ για όλα τα i
- Ο αλγόριθμος τερματίζεται σε N το πολύ βήματα, που αποδεικνύει το μέρος b

Πολυπλοκότητα Αλγόριθμου Bellman-Ford

- Πλήθος υπολογισμών στη χειρότερη περίπτωση για την εύρεση των ελάχιστων διαδρομών
  - Ο αλγόριθμος επαναλαμβάνεται το πολύ N φορές
  - Κάθε επανάληψη γίνεται για N-1 κόμβους (όλα τα $i \neq 1$)
  - Το βήμα της ελαχιστοποίησης απαιτεί την εξέταση το πολύ N-1 ενδεχομένων

  $\Rightarrow$ Η υπολογιστική πολυπλοκότητα είναι $O(N^3)$
  - Πιο λεπτομερής υπολογισμός δίνει υπολογιστική πολυπλοκότητα τάξεως $O(hM)$

Κατασκευή Ελάχιστων Δρόμων

- Ο αλγόριθμος B-F algorithm δίνει τα *μήκη* των ελάχιστων δρόμων αλλά μας ενδιαφέρει να βρούμε και *ποιες είναι* οι διαδρομές αυτές
- Ξεκινάμε από την εξίσωση B-F

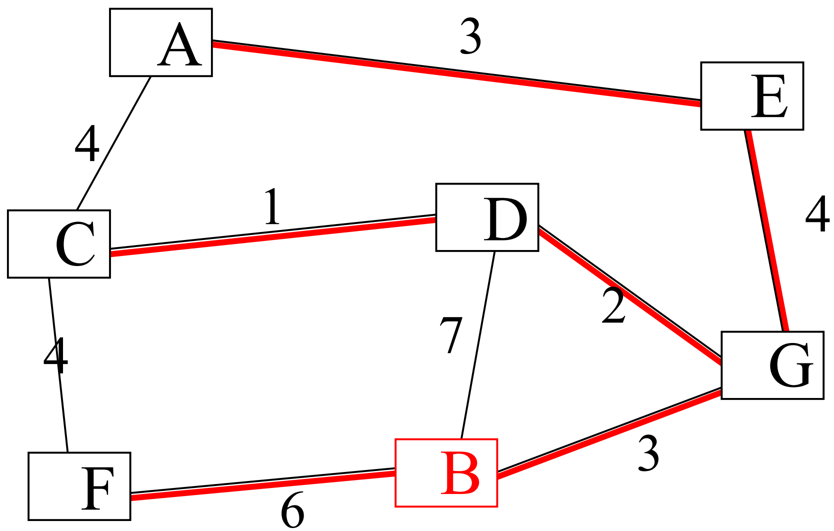
$$D_i = \min_{j \in G}\left[D_j + d_{ij}\right], \ \forall i \neq 1, \ \text{ and } \ D_1 = 0$$

- Για κάθε κόμβο $i \neq 1$, διαλέγουμε την πλευρά (*i,j*) που ελαχιστοποιεί την εξίσωση B-F
  - Αυτό παράγει έναν υπο-γράφο με *N*-1 πλευρές (ένα δέντρο)
  - Για κάθε κόμβο *I,* ακολουθούμε τις πλευρές από τον *i* κατά μήκος αυτού του υπο-γράφου μέχρι να φθάσουμε στον 1
- Παρατήρηση: Για γράφους χωρίς κύκλους μηδενικού ή αρνητικού μήκους, η εξίσωση B-F ορίζει ένα σύστημα *N*-1 εξισώσεων που έχουν μια *μοναδική* λύση

Παράδειγμα Κατασκευής Ελάχιστου Δρόμου με τον Αλγόριθμο B-F

- $D_A = 3 + D_E$
- $D_E = 4 + D_G$
- $D_G = 3 + D_B = 3$
- $D_D = 2 + D_G$
- $D_C = 1 + D_D$
- $D_F = 6 + D_B = 6$

- *AB = A-E-G-B*



Ο Αλγόριθμος του Dijkstra (1)

- Διαφορετικά κριτήρια στις επαναλήψεις
  - Ο αλγόριθμος προχωρά αυξάνοντας το *μήκος δρόμου* αντί του *πλήθους πλευρών*
    - Ξεκινάμε με τον "πλησιέστερο" κόμβο στον προορισμό, τον χρησιμοποιούμε για να βρούμε τον΄επόμενο πλησιέστερο κόμβο κοκ.
  - Για τη σύγκλιση απαιτείται να έχουμε *μη αρνητικά* βάρη πλευρών
- Διακρίνονται δυο κατηγορίες κόμβων
  - *L*: Αποδεκτοί κόμβοι (στον ελάχιστο δρόμο)
  - *C*: Υποψήφιοι κόμβοι (εκτός ελάχιστου δρόμου ως τότε)
  - Σε κάθε επανάληψη ένας κόμβος μετακινείται από το *C* στο *L*

Ο Αλγόριθμος του Dijkstra (2)

- Έναρξη
  - $L = \{1\}$ and $C = G - L$ (ο κόμβος 1 είναι ο προορισμός)
  - $D_1 = 0$ και $D_j = d_{j1}$ για $j \neq 1$
- Επαναληπτικά βήματα
  1. Βρείτε τον επόμενο *πλησιέστερο* κόμβο εκτός *L,* δηλαδή,
     - Βρείτε τον κόμβο $i \notin L$ τέτοιον ώστε $$D_i = \min_{j \notin L} D_j$$
     - Τροποποιείστε τα *C* και *L*: $L = L \cup \{i\}$ και $C = C - \{i\}$
  2. Τροποποιείστε τα μήκη δρόμων των κόμβων που παραμένουν στο *C*

$$D_j := \min[D_j, D_i + d_{ji}], \ \forall j \in C$$

- Ο αλγόριθμος τερματίζει όταν $L = G$

Απόδειξη του Αλγόριθμου Dijkstra (1)

- Στην αρχή κάθε βήματος 1, έχουμε
  - a $D_i \le D_j$ για όλα τα $i \in L$ και $j \notin L$
  - b Για κάθε κόμβο j, D_j είναι η ελάχιστη απόσταση του j από το 1 για οποιοδήποτε δρόμο χρησιμοποιώντας κόμβους (πιθανόν εκτός του j) μόνον από το L

Μη αρνητικά βάρη πλευρών

- Απόδειξη της συνθήκης (a)
  - Η (a) ικανοποιείται αρχικά
  - Έχουμε $d_{ji} \ge 0$ και $D_i = \min_{j \notin L} D_j$ έτσι ώστε η συνθήκη (a) να διατηρείται από την τροποποιημένη εξίσωση

$$D_j := \min[D_j, D_i + d_{ji}], \ \forall j \in C$$

Απόδειξη του Αλγόριθμου Dijkstra (2)

- Μένει να αποδείξουμε τη συνθήκη (b), που θα το κάνουμε με επαγωγή
  - Η (b) ικανοποιείται αρχικά
  - Επαγωγική υπόθεση (Η)
    - Η συνθήκη (b) ισχύσει στην αρχή κάποιου βήματος 1, όπου *i* είναι ο κόμβος που προστίθεται στο *L* σ’ αυτό το βήμα
  - Από την (Η), η συνθήκη (b) ισχύει για κάθε κόμβο *i* καθώς και $\forall\ j \in L$ λόγω της (a)
  - Θεωρήστε στη συνέχεια έναν κόμβο $j \notin L \cup \{i\}$

Απόδειξη του Αλγόριθμου Dijkstra (3)

- Έστω D'_j το μήκος του ελάχιστου δρόμου από $j \notin L \cup \{i\}$ στο 1 με όλους τους κόμβους του εκτός του j στο $L \cup \{i\}$
  - Ο δρόμος αυτός αποτελείται από μια πλευρά (j,k), $k \in L \cup \{i\}$, συνενωμένη με τον ελάχιστο δρόμο από k σε 1 με κόμβους στο $L \cup \{i\}$
    - Από την (H), D_k είναι το μήκος αυτού του ελάχιστου δρόμου από k στο 1
  - Έχουμε

$$D'_j = \min_{k \in L \cup \{i\}} [D_k + d_{jk}] = \min\left[\min_{k \in L}[D_k + d_{jk}], D_i + d_{ji} \right]$$

  - Κι από την (H) έχουμε επίσης

$$D_j = \min_{k \in L}[D_k + d_{jk}]$$

- Συνδυάζοντας τα δυο αυτά παίρνουμε

$$D'_j = \min[D_j, D_i + d_{ji}] = D_j$$

- Κάτι που ολοκληρώνει την απόδειξη της επαγωγής

Υπολογιστική Πολυπλοκότητα Αλγόριθμου Dijkstra

- Όπως με τον αλγόριθμο Bellman-Ford, έχουμε μέχρι N επαναλήψεις, όπου $N = |G|$
- Η κύρια εξοικονόμηση είναι ότι σε κάθε επανάληψη θεωρούμε κάθε κόμβο μόνο *μια φορά*
  - Στα βήματα 1 και 2 κοιτάμε πρώτα τους κόμβους στο L (βήμα 1) και μετά στο $C = G\text{-}L$ (βήμα 2)
  - Το κόστος των πράξεων σε κάθε βήμα είναι το ίδιο (μια μονάδα κόστους)
- Το πολύ N επαναλήψεις και N πράξεις μιας μονάδας κόστους ανά επανάληψη
  - Το συνολικό κόστος είναι τάξης $O(N^2)$ (αλλά μπορεί να βελτιωθεί)

Ο Αλγόριθμος Floyd-Warshall (1)

- Σκοπεύει στον υπολογισμό των ελάχιστων δρόμων μεταξύ *όλων των ζευγαριών* των κόμβων
  - Ο αλγόριθμος αυτός επιτρέπει θετικά κι αρνητικά βάρη πλευρών αλλά απαιτεί να μην υπάρχουν κύκλοι με αρνητικό μήκος
- Οι επαναλήψεις γίνονται ως προς το *σύνολο των κόμβων* που αφήνονται να είναι ενδιάμεσοι κόμβοι σ' ένα δρόμο
  - Η αρχική συνθήκη είναι όπως για Bellman-Ford και Dijkstra,δηλαδή, δρόμοι μιας πλευράς για όλους τους κόμβους
  - Η επόμενη επανάληψη αφήνει μόνο τον κόμβο 1 να γίνει ενδιάμεσος κόμβος
  - Ο αλγόριθμος τερματίζει όταν όλοι οι κόμβοι έχουν χρησιμοποιηθεί ως ενδιάμεσοι κόμβοι

Ο Αλγόριθμος Floyd-Warshall (2)

- Αρχικές συνθήκες

$$D_{ij}^{0} = d_{ij}, \quad \forall i, j \ \text{ και } \ i \neq j$$

- Έστω D_{ij}^{n} το μήκος του ελάχιστου δρόμου μεταξύ των κόμβων *i* και *j*, όταν οι ενδιάμεσοι κόμβοι στο δρόμο περιορίζονται στους κόμβους 1,2,...,*n*
- Το επαναληπτικό βήμα ορίζεται ως

$$D_{ij}^{n+1} = \min[D_{ij}^{n}, D_{i(n+1)}^{n} + D_{(n+1)j}^{n}], \ \forall i \neq j$$

  - Ελέγξτε αν η απόσταση μεταξύ των κόμβων *i* και *j* βελτιώνεται χρησιμοποιώντας τον κόμβο (*n*+1) ως ενδιάμεσο κόμβο

Απόδειξη Αλγόριθμου Floyd-Warshall

- Πάλι επαγωγικά
- Για *n*=0 οι αρχικές συνθήκες προφανώς δίνουν τους ελάχιστους δρόμους χωρίς ενδιάμεσους κόμβους
- Η Επαγωγική Υπόθεση (Η)
  - Για δοθέν *n*, το D_{ij}^{n} δίνει το μήκος του ελάχιστου δρόμου από *i* σε *j* με ενδιάμεσους μόνο τους κόμβους 1 ως *n*
- Ο ελάχιστος δρόμος από *i* σε *j* με ενδιάμεσους μόνο τους κόμβους 1 ως (*n*+1) είτε περιλαμβάνει τον κόμβο (*n*+1) ή δεν τον περιέχει
  - Αν ναι, το μήκος του δρόμου είναι ο 2ος όρος του επαναληπτικού βήματος
  - Αν όχι, το μήκος του δρόμου είναι ο 1ος όρος του επαναληπτικού βήματος

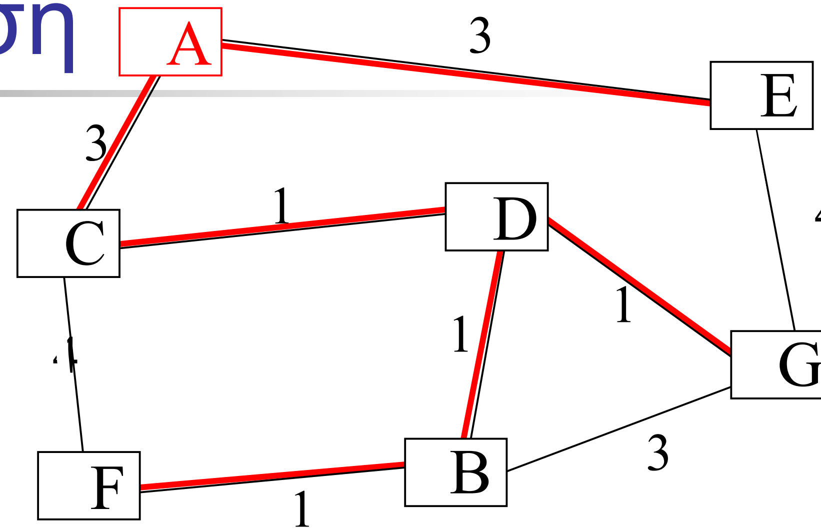
Πολυπλοκότητα Αλγόριθμου Floyd-Warshall

- *N* επαναληπτικά βήματα
  - Ένα για κάθε δυνατό σύνολο ενδιάμεσων κόμβων
- Οι υπολογισμοί σε κάθε επαναληπτικό βήμα περιλαμβάνουν συγκρίσεις για κάθε ζευγάρι κόμβων
  - Υπάρχουν $N(N-1)$ ζευγάρια κόμβων
- Το συνολικό κόστος είναι της τάξης $O(N^3)$
  - Ισοδύναμο με *N* εφαρμογές του αλγόριθμου Dijkstra
    - Κάθε εφαρμογή του Dijkstra δίνει ελάχιστους δρόμους από όλους τους κόμβους προς έναν προορισμό

Ελάχιστοι Δρόμοι Δοθέντων Προελεύσεων

- Η διαφορά είναι τώρα ότι ενδιαφερόμαστε να βρούμε δρόμους που ξεκινούν από δοθέντα κόμβο προέλευσης προς όλους τους άλλους κόμβους αντί από όλους τους κόμβους προέλευσης προς δοθέντα κόμβο προορισμού
- Η βασική προσέγγιση παραμένει η ίδια
  - Δευτερεύουσες διαφορές στους ορισμούς των αρχικών συνθηκών και των επαναληπτικών βημάτων
  - Θα εξετάσουμε και τον αλγόριθμο Bellman-Ford και τον αλγόριθμο Dijkstra με αυτή την εστίαση στον κόμβο προέλευσης

Αλγόριθμος Bellman-Ford με Δοθείσα Προέλευση



- Η προέλευση είναι ο κόμβος A
- Έναρξη
  - $D_A^h = 0, \forall h$
  - $D_i^0 = \infty$ για $i \neq$ A
- Ορίζουμε

$$D_i^{h+1} = \min_j \{D_j^h + d_{ji}\}, \ \forall i \neq \text{A}$$

Την απόσταση από τον A προς τον i μέσω το πολύ h+1 πλευρών

- Όπως πριν, οι επαναλήψεις τερματίζουν όταν επιτυγχάνεται σύγκλιση

$D^1_E = 3$, $D^1_C = 3$

$D^2_E = 3$, $D^2_C = 3$, $D^2_D = 4$, $D^2_G = 7$, $D^2_F = 7$

$D^3_E = 3$, $D^3_C = 3$, $D^3_D = 4$, $D^3_G = 5$, $D^3_F = 7$, $D^3_B = 5$

$D^4_E = 3$, $D^4_C = 3$, $D^4_D = 4$, $D^4_G = 5$, $D^4_F = 6$, $D^4_B = 5$

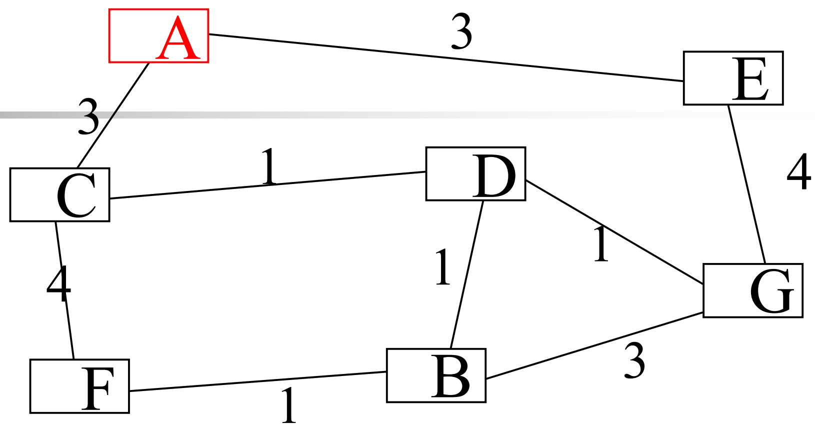
Αλγόριθμος Dijkstra με Δοθείσα Προέλευση

- Έναρξη
  - L = {A} και $C = G - L$ (ο κόμβος A είναι η προέλευση)
  - $D_A = 0$ και $D_i = d_{Aj}$ για $j \neq$ A
- Επαναληπτικά βήματα
  1. Βρείτε τον επόμενο *πλησιέστερο* κόμβο στο C
     - Βρείτε τον κόμβο $i \notin L$ τέτοιο ώστε

$$D_i = \min_{j \notin L} D_j$$

     - Τροποποιήστε το C και L: $L = L \cup \{i\}$ και $C = C - \{i\}$
  2. Τροποποιήστε τα μήκη των δρόμων που απομένουν στο C

$$D_j := \min[D_j, D_i + d_{ij}],\ \forall j \in C$$



{A}; D_E=3,D_C=3

{A,E}; D_E=3; D_C=3,D_G=7

{A,E,C}; D_E=3,D_C=3; D_D=4,D_G=7,D_F=7

{A,E,C,D}; D_E=3,D_C=3,D_D=4; D_G=5,D_F=7,D_B=5

{A,E,C,D,B}; D_E=3,D_C=3,D_D=4, D_B=5; D_G=5,D_F=6

{A,E,C,D,B,G}; D_E=3,D_C=3, D_D=4,D_B=5,D_G=5; D_F=6

{A,E,C,D,B,G,F}; D_E=3,D_C=3, D_D=4,D_B=5,D_G=5,D_F=6

Δέντρα Ελάχιστων Δρόμων

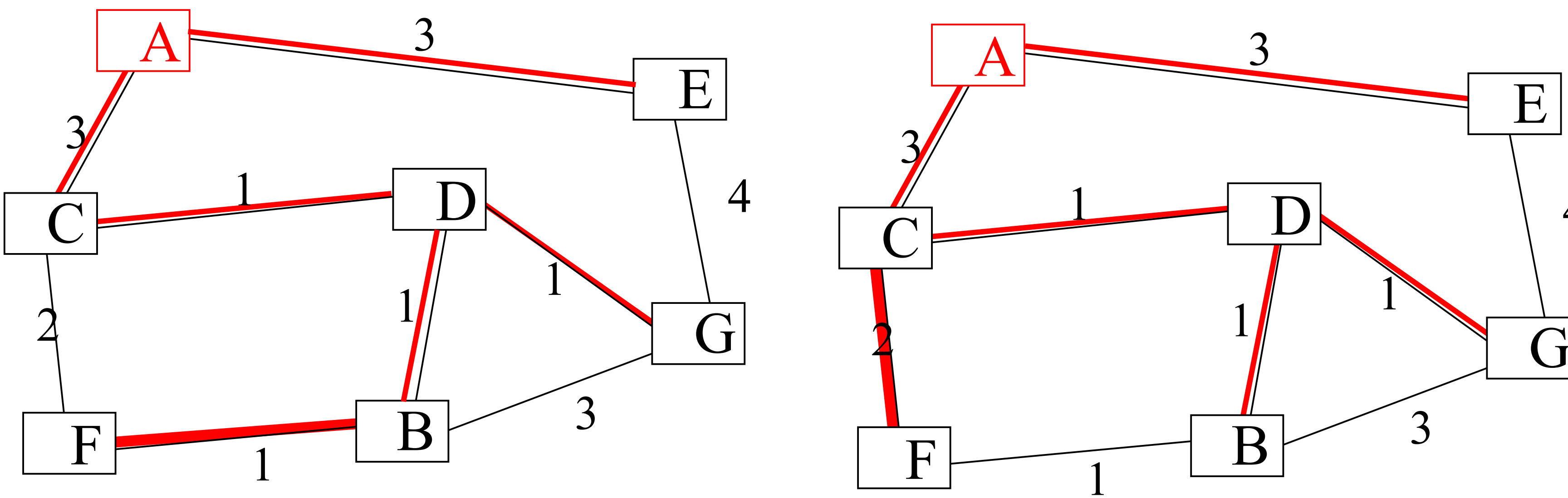
- Για την προσθετική συνάρτηση απόστασης έχουμε υποθέσει:
  - Αν ο δρόμος $p=i_1,i_2,\ldots,i_h$, με $i=i_1$ and $j=i_h$ είναι ένας ελάχιστος δρόμος από i σε j, τότε για κάθε k, $0\leq k\leq h-1$, ο υπο-δρόμος $s=i_1,i_2,\ldots,i_k$ είναι ένας ελάχιστος δρόμος από i σε i_k
    - Αν αυτό δεν ίσχυε, θα μπορούσαμε να βελτιώναμε τον ελάχιστο δρόμο μεταξύ i και j διαλέγοντας τον ελάχιστο δυνατό δρόμο που υπάρχει μεταξύ i και i_k
  - Έστω D διάνυσμα των αποστάσεων από τον κόμβο l. Τότε, ένας κατευθυνόμενος δρόμος p από i σε j είναι ελάχιστος δρόμος αν και μόνον αν $D_k = D_l + d_{kl}$ για όλες τις πλευρές $(k,l)\in p$

$\Rightarrow$ Μπορούμε να κατασκευάσουμε ένα δέντρο ελάχιστου δρόμου από τον κόμβο i προς όλους τους άλλους κόμβους

Δέντρο Ελάχιστου Δρόμου και Ελάχιστο Εκτεταμένο Δέντρο

- Τα δυο δέντρα **ΔΕΝ** είναι τα ίδια

Ελάχιστο Εκτεταμένο Δέντρο Δέντρο Ελάχιστου Δρόμου



Συνολικό Βάρος: 10
$D_{AF} = 6$

Συνολικό Βάρος: 11
$D_{AF} = 5$