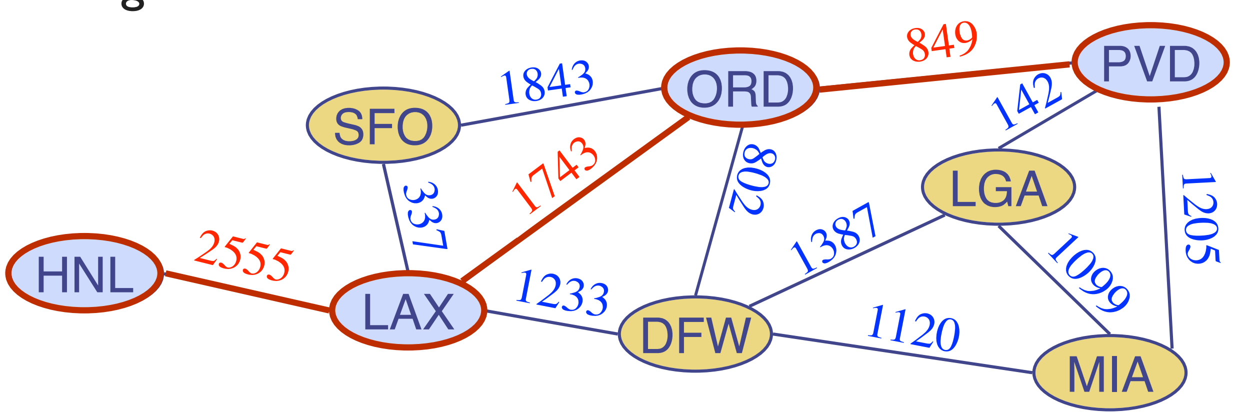
Chemin de poids minimal

- Étant donné un graphe et deux sommets u et v, on veut trouver le chemin de poids total minimal entre u et v.
  - on va appeler la longueur d'un chemin la somme des poids des arêtes faisant parties de ce chemin
  - le chemin de poids minimal (ou longueur minimal) entre u et v sera appelé la distance entre u et v

- Exemple: Chemin de longueur minimale entre PVD et HNL:
  Longueur = 5147



© adapté de Goodrich et Tamassia 2004

Propriétés

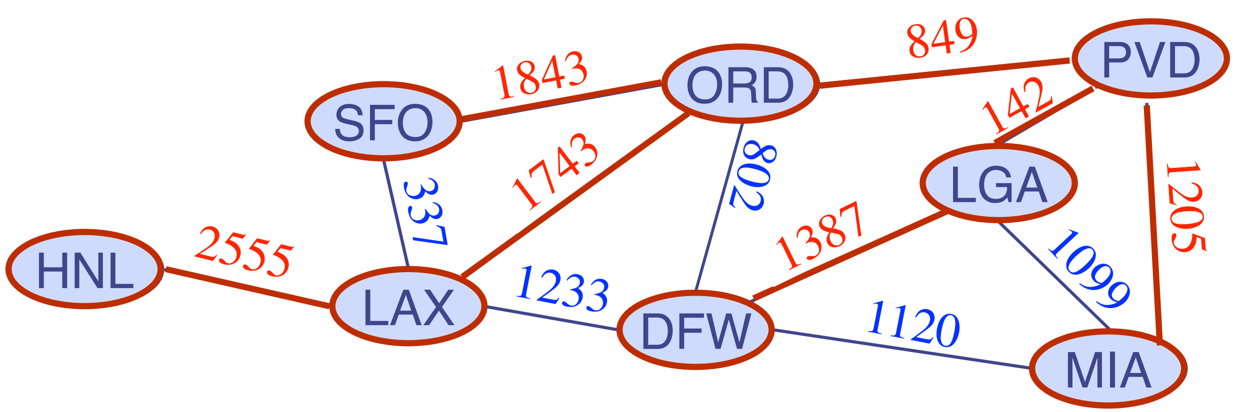
- **Propriété 1:**

  Un sous-chemin d'un chemin de coût minimal est un chemin de coût minimal

- **Propriété 2:**

  Étant donné un sommet v d'un graphe non-orienté et connexe G, il existe un arbre couvrant composé des chemins de coût minimal entre v et tous les autres sommets du graphe

- **Exemple:** Arbres des chemins de coût minimal de PVD



Algorithme de Dijkstra

- La distance entre un sommet v et un sommet s est le poids total minimal d'un chemin entre v et s

- L'algorithme de Dijkstra est un algorithme glouton qui calcule les distances entre un sommet v et tous les autres sommets d'un graphe

- Ici, on va assumer:
  - Le graphe est connexe
  - Les poids des arêtes sont non-négatifs

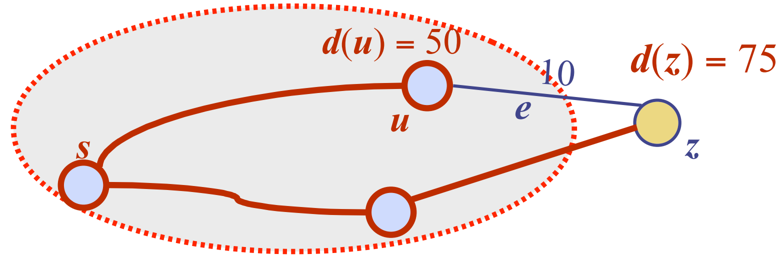
- On va faire grossir un "nuage" de sommets, contenant au départ v et couvrant éventuellement tous les sommets

- On va donner une étiquette d(u) à chaque sommet, représentant la distance entre v et u dans le sous-graphe constitué des sommets dans le nuage et des arêtes adjacentes

- À chaque étape:
  - On ajoute au nuage le sommet u extérieur au nuage qui a la plus petite étiquette d(u)
  - On met à jour les étiquettes des sommets adjacents à u

Relaxation des arêtes

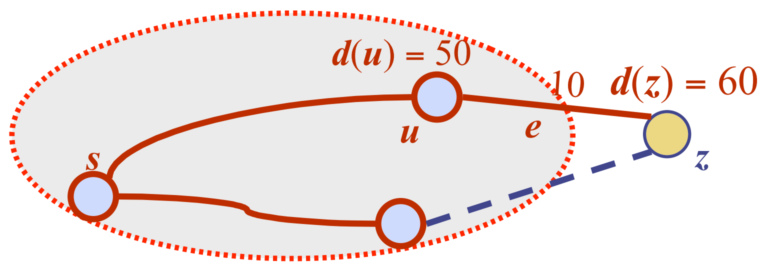
- On appelle la mise à jour des étiquettes des sommets, après l'ajout d'un sommet u dans le nuage, la relaxation des arêtes



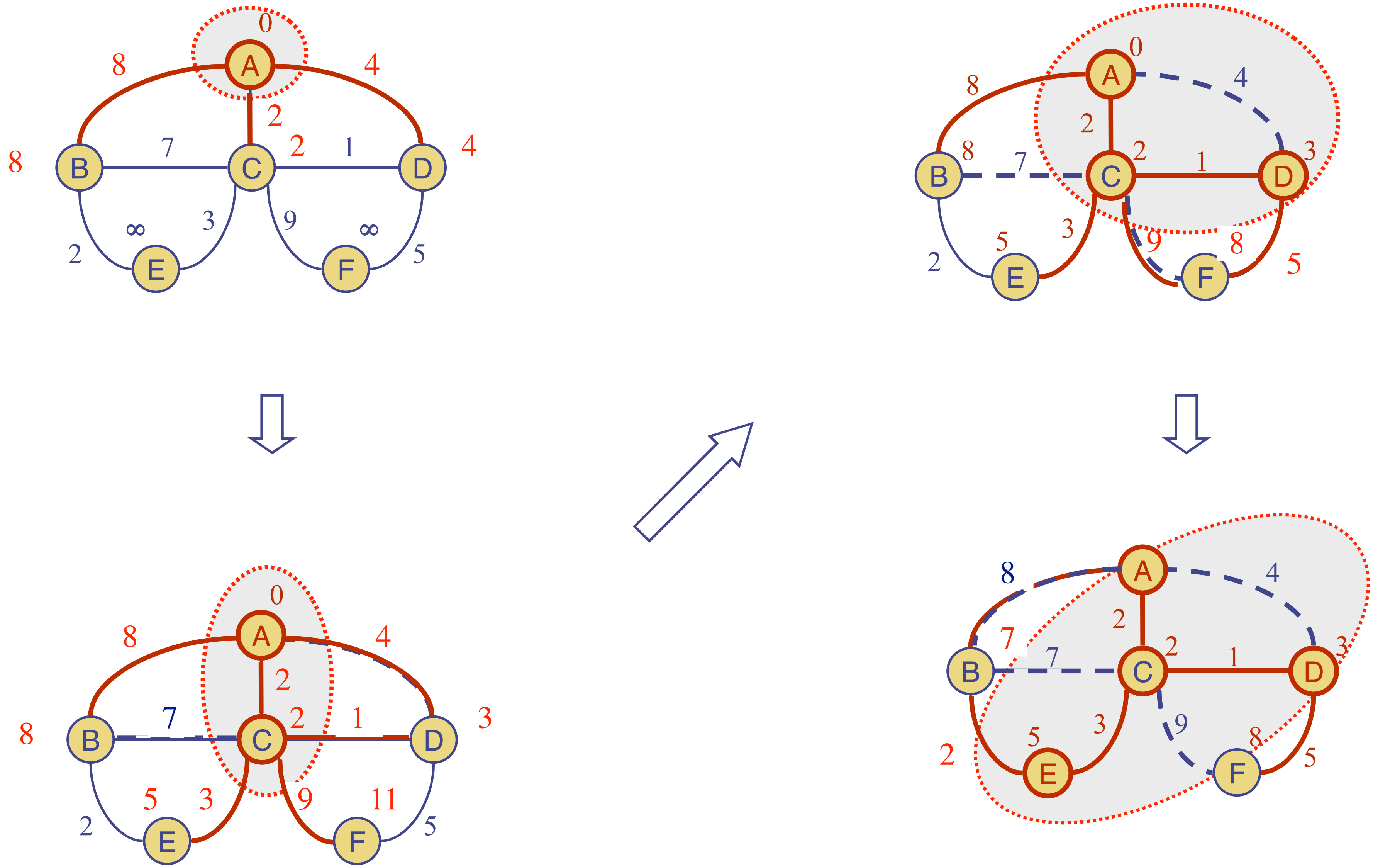
- Considérons une arête e=(u,z) telle que
  - u est le sommet qu'on vient d'ajouer au nuage
  - z n'est pas dans le nuage

- La relaxation de e, consiste à mettre à jour la distance d(z) comme suit:

$$d(z) \leftarrow \min\{d(z), d(u) + poids(e)\}$$

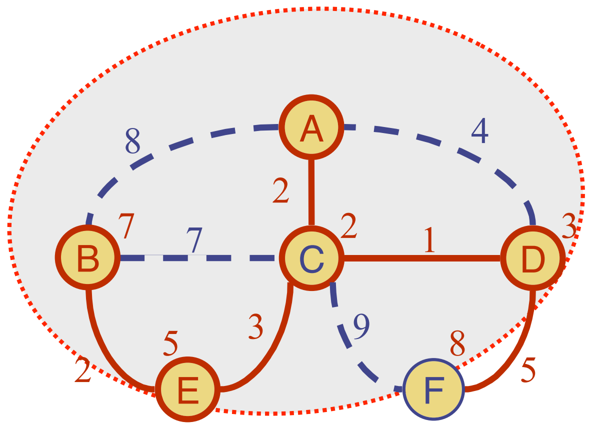


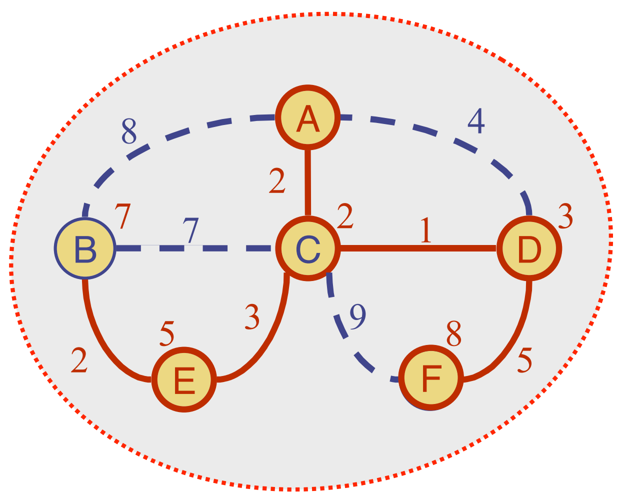
Exemple:



A B C D E F
0 8 4 2 7 2 1 8 4 ∞ 3 9 ∞ 2 5
0 8 4 2 7 2 1 8 3 5 3 9 11 2 5
0 8 4 2 8 7 2 1 3 5 3 9 8 2 5
8 4 2 7 7 2 1 3 5 3 9 8 2 5

Exemple (suite):





Analyse de complexité de l'algorithme de Dijkstra

- Insérer les sommets et leur étiquette correspondante dans une liste avec priorités prend un temps O(nlogn) si on insère les éléments un à un, ou un temps O(n) si on utilise une construction de monceau de bas en haut

- À chaque tour de boucle TANT QUE, on prend un temps O(logn) pour enlever un sommet u du monceau, et un temps O (deg(u) log n) pour exécuter la procédure de relaxation

- Le temps d'exécution total de la boucle TANT QUE est donc de

$$\sum_{v \in G} (1 + deg(v)) \log n$$

- La complexité en temps de Dijkstra est de O((n+m)logn)