Théorie des Graphes
Plus courts chemins

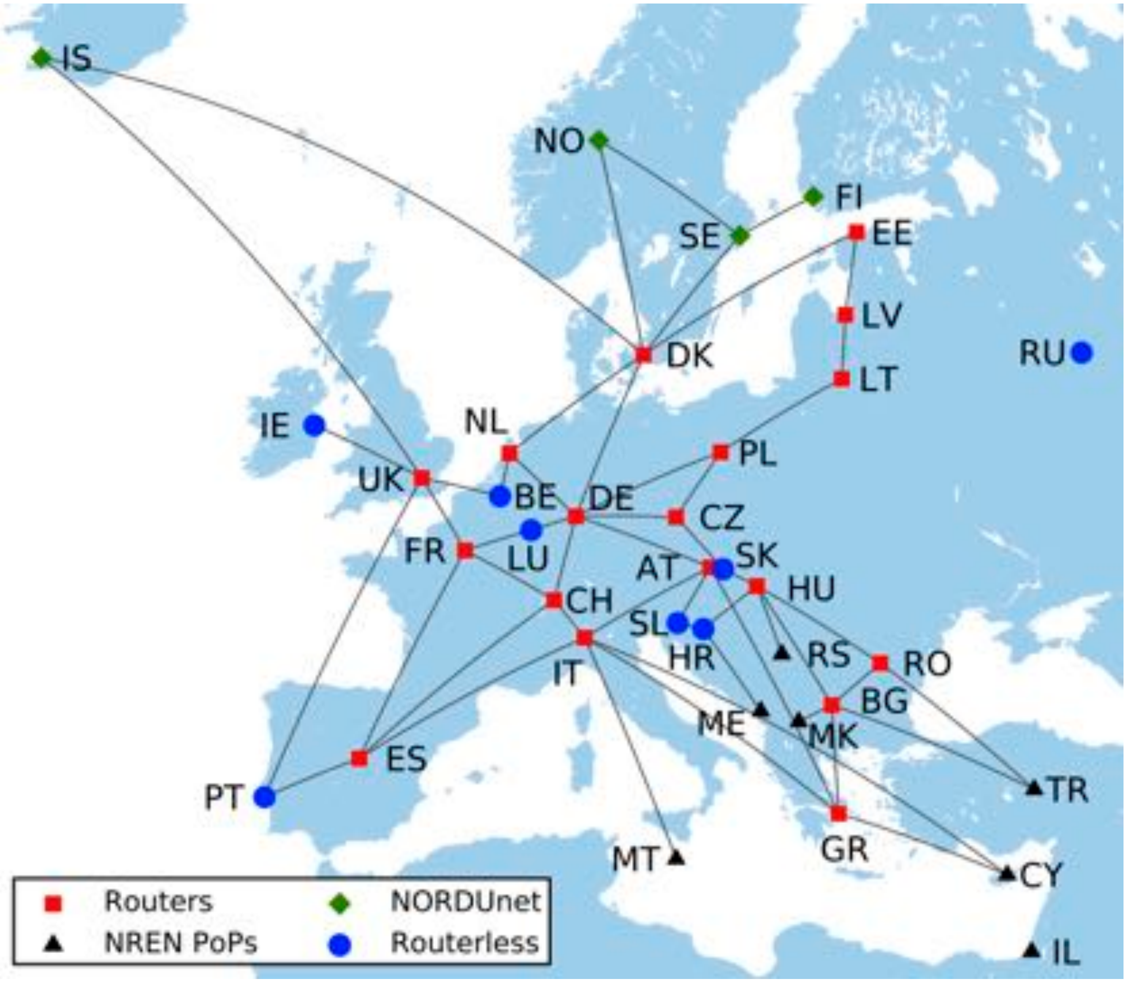
Fabrice Theoleyre

theoleyre@unistra.fr
http://www.theoleyre.eu

Modélisation d’un réseau

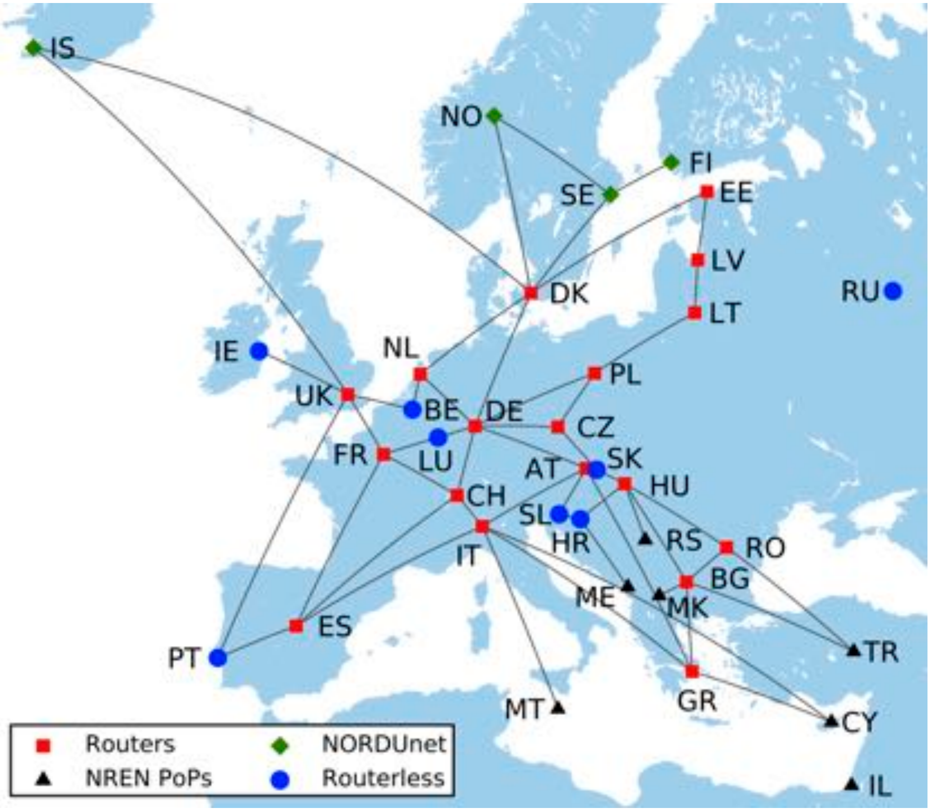
- Graphe G(S,A,L)
  - Sommets = routeurs, stations
  - Arêtes = lien de communication
  - L = longueur d’une arête (poids)

Geant



Plus courts chemins

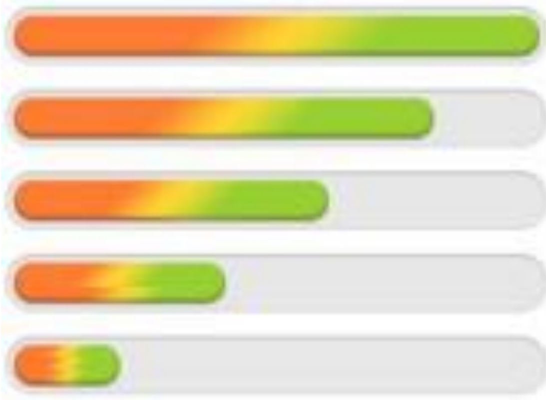
- Longueur d'un chemin
  - Somme de la longueur des arêtes
  - Par exemple:
    - ❖ $L(c) = \sum_{(u,v)\in c} L((u,v))$



- Plus courts chemins
  - c_{court} est le plus court chemin de u à v :
    - ❖ $\forall c \in \mathcal{C}(u,v), L(c_{court}) \leq L(c)$
    - ❖ $\mathcal{C}(u,v)$ l'ensemble des chemins de u à v
    - ❖ Pas d'unicité
    - ❖ Long : $\geq$
  - notion de sous optimalité des meilleurs chemins

Métrique de décision

- Que choisir comme poids d'un chemin / arête?
  - Additive
    - ❖ Délais, longueurs, gigue
    - ❖ min,+ : les plus courts
      - $L(c) = \sum_{(u,v)\in c} L((u,v))$
      - $c_{court} = min_{c \in \mathcal{C}(u,v)}\, L(c)$
    - ❖ max, + : les plus longs
  - Concave/convexe
    - ❖ bande passante, charge du routeur
    - ❖ max,min : les plus gros
      - $L(c) = \boldsymbol{Min}_{(u,v)\in c} L((u,v))$
      - $c_{court} = \boldsymbol{max}_{c \in \mathcal{C}(u,v)}\, L(c)$
    - ❖ min, max / max, max / min,min
  - Multiplicative (taux de perte / de succès)
    - ❖ max,*   / min, *

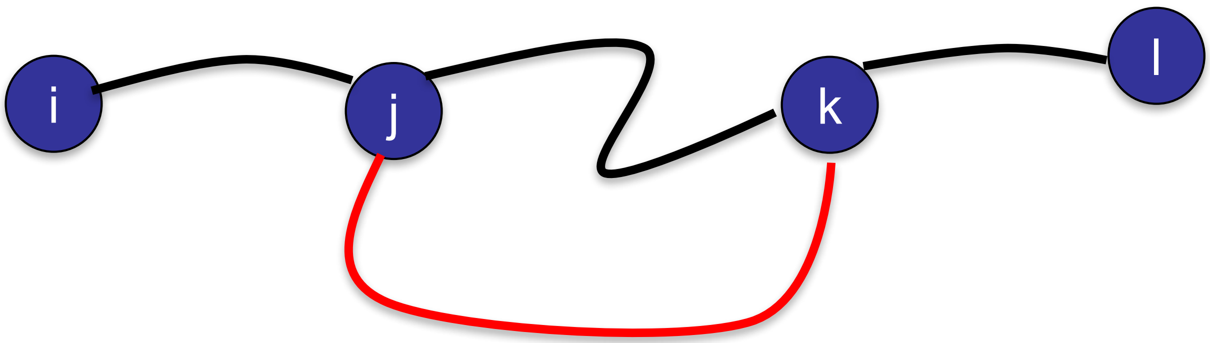


Conditions sur l'existence

- Soit R=(S,A,L) un réseau
  - u,v $\in$ S
- 3 possibilités (exclusives) :
  - il n'existe aucun chemin de u à v; *// graphe non fortement connexe*
  - il existe un chemin mais pas de plus court; *// existence d'un circuit absorbant*
    - ❖ Circuit c absorbant
      - $L(c) < 0$
    - ❖ Si une composante connexe contient un circuit absorbant
      - Il n'existe aucun plus court chemin entre toute paire de sommets de la composante
      - Preuve ?
  - il existe un plus court chemin de u vers tout autre sommet de S\{u} *// u et v sont dans la même composante connexe*
    - ❖ si R est fortement connexe
    - ❖ et si R ne contient aucun circuit absorbant

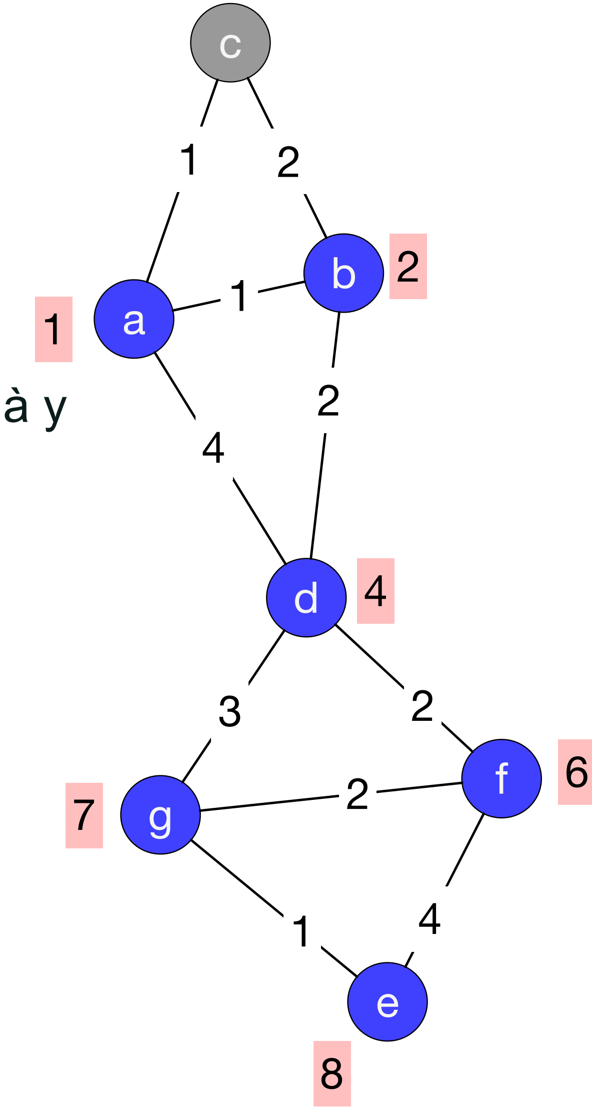
Propriétés des plus courts chemins

- Sous-optimalité des plus courts chemins
  - R=(S,A,L) un réseau
  - $c_{court} = \{(u_i, u_{i+1})\}_{i \in [0,l-1]}$ un plus court chemin
  - $\forall j, k \in [0, l-1] |\ 0 \leq j \leq k \leq l,$ alors $\{(u_i, u_{i+1})\}_{i \in [j,k-1]}$ est également un plus court chemin entre $u_j\ et\ u_k$
    - ❖ Fondamental : on découpe en sous-problèmes
  - démonstration par l'absurde



Propriétés des plus courts chemins

- Graphe (acyclique - sans circuits) des plus courts chemins
  - Soit R=(S,A,L) un réseau sans circuit absorbant
  - soit u $\in$ S un sommet de R
  - Soit $\pi_u : S \rightarrow \mathbb{R} \cup \{+\infty\}$ l'application définie par
    - ❖ $\pi_u(v) = L(C)$ s'il existe un plus court chemin C de x à y dans R
      - – Dénote la *distance* d'un sommet à u
    - ❖ $+\infty$ sinon
  - Soit R'=(S,A') le graphe partiel de (S,A)
    - ❖ $A' = \{(v,w) \in A \,|\, L(v,w) = \pi_u(w) - \pi_u(v)\}$
    - ❖ il s'agit du graphe des plus courts chemins
      - – tous ses arcs appartiennent à un (ou plusieurs) plus court chemin(s) de u à un sommet dans R
    - ❖ Exemple de droite ?



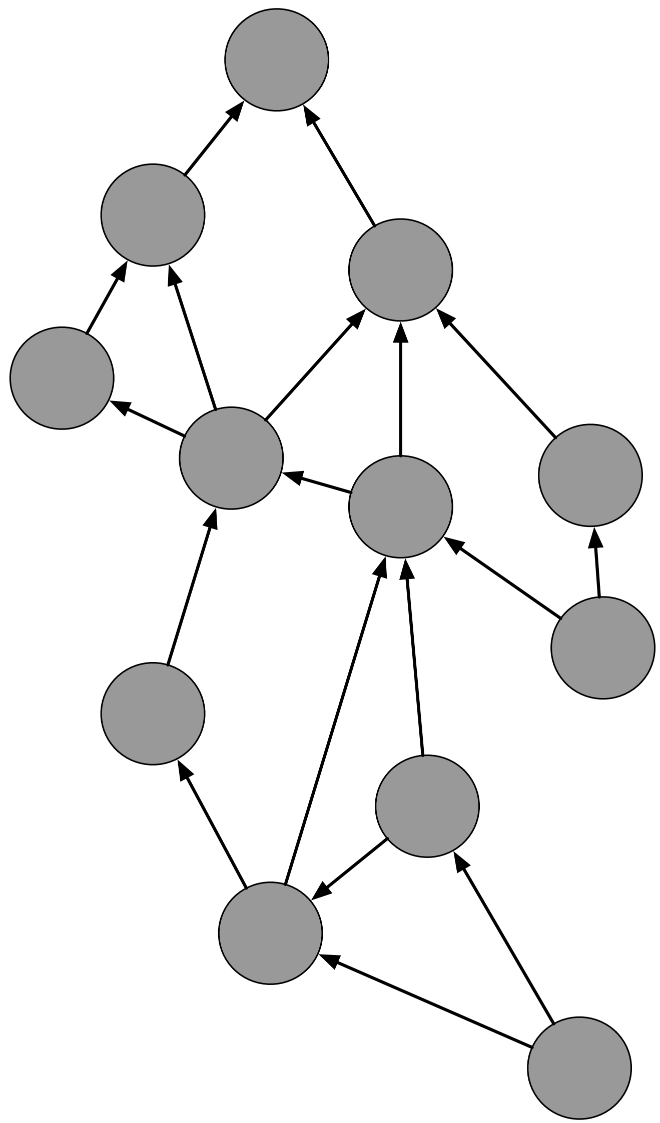
Propriétés des plus courts chemins

- R'=(S,A',L) le graphe (partiel) des plus courts chemin d'un réseau R=(S,A,L) sans circuit absorbant
  - un plus court chemin de u à v dans R est également un plus court chemin de u à v dans R'

- Arborescence des plus courts chemins
  - Réseau R=(S,A,L) sans circuit absorbant et u un sommet de R
  - (S,A') le graphe des plus courts chemins depuis u dans R
  - $S' = \pi_u^{-1}(R) = \{v \in S | \pi_u\,(v) \neq \infty\}$
  - $A'' \subset A \mid (S', A'', u)$ *est une arborescence des plus courts chemins*
  - Pas forcément d'unicité !
    - ❖ Plusieurs plus courts chemins

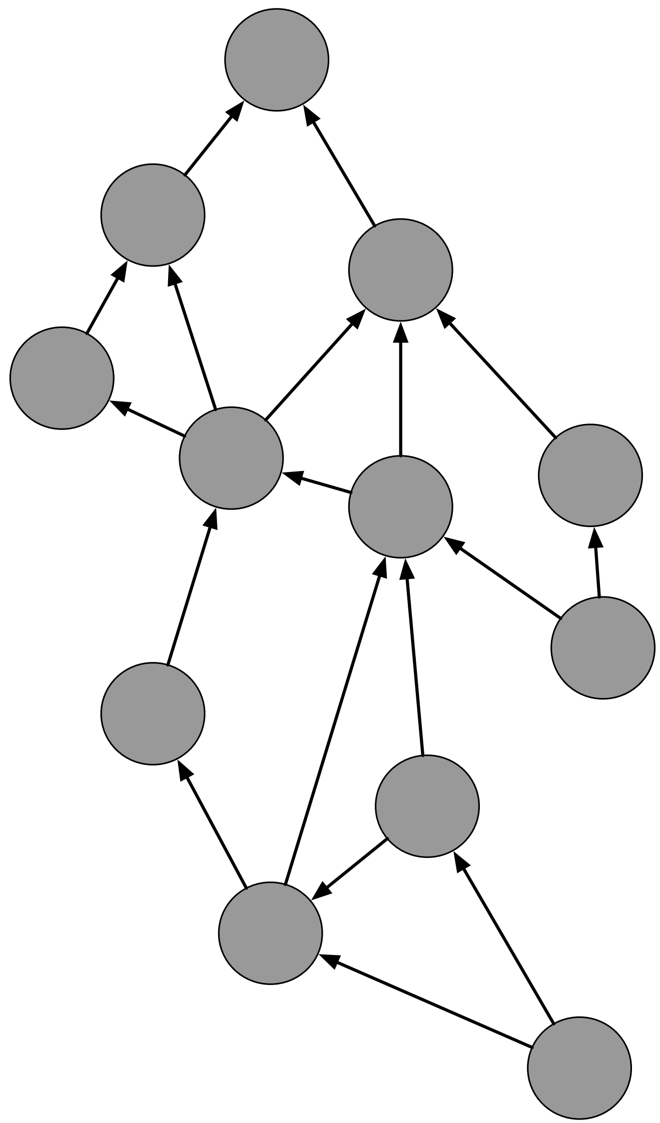
Graphe orienté acyclique

- Soit un Graphe orienté G(S,A)
- Graphe orienté acyclique (DAG) ou Graphe orienté sans cycle
  - $\forall u \in S, \nexists \{(\mathrm{v_i}, \mathrm{v_{i+1}})\}_{\mathrm{i} \in [0,\mathrm{k}], \mathrm{k}>0} \mid \mathrm{v_0} = \mathrm{u} \wedge \mathrm{v_{k+1}} = \mathrm{u} \wedge (\mathrm{v_i}, \mathrm{v_{i+1}}) \in A$
    - G est sans circuit (même une boucle)
  - Exemple ?
- R'=(S,A',L) le graphe (partiel) des plus courts chemin d'un réseau R=(S,A,L) sans circuit absorbant
  - $A' = \{(v, w) \in A \mid L(v, w) = \pi_u(w) - \pi_u(v)\}$
  - R' forme un DAG
    - Preuve ?
  - toutes ses composantes fortement connexes sont des singletons
  - R: Arbre → pour les graphes non orientés



Graphe des plus courts chemin

- Définition / contexte
  - u est une source si $d^-(u)=0$
  - u est un puits si $d^+(u)=0$
    - ❖ au moins une source et un puits



- Rang, hauteur
  - Le rang r(u) d'un sommet u dans un graphe orienté est la longueur maximale d'un chemin élémentaire se terminant par u
  - La hauteur h(u) d'un sommet u dans un graphe orienté est la longueur maximale d'un chemin élémentaire débutant par u
  - u est une source (respectivement puits) ssi r(u)=0 (resp. ssi h(u)=0)

Détection de circuits

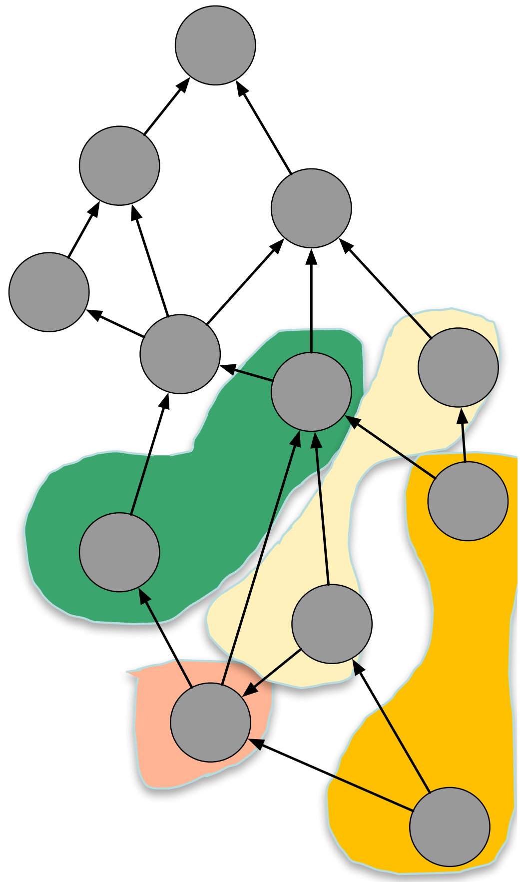
- Parcours en profondeur
  - On part de la racine
  - On explore chaque sommet tant
    - ❖ qu'on ne reboucle pas
    - ❖ On aboutit à un sommet impasse (tous ses voisins sortant ont été explorés)

- Complexité linéaire avec le nb. d'arêtes
  - On ne teste pas deux fois la même arête

- Application
  - Un graphe orienté sans circuit à 10 sommets (DAG)
  - Donner le rang et la hauteur de chacun
    - ❖ p (puits) et s (source)

Chemins dans les réseaux sans circuits

- Partition en niveaux
  - Un graphe orienté G=(S,A) admet une partition en niveaux s'il existe une partition $\{S_i\}^t_{i=0}$ (t>=0) de S telle que S_0 est l'ensemble des sources de G et pour tout i $\in$ [1,t], S_i est l'ensemble des sources du sous graphe de G induit par $S \setminus \cup_{j=1}^{i-1} S_j$



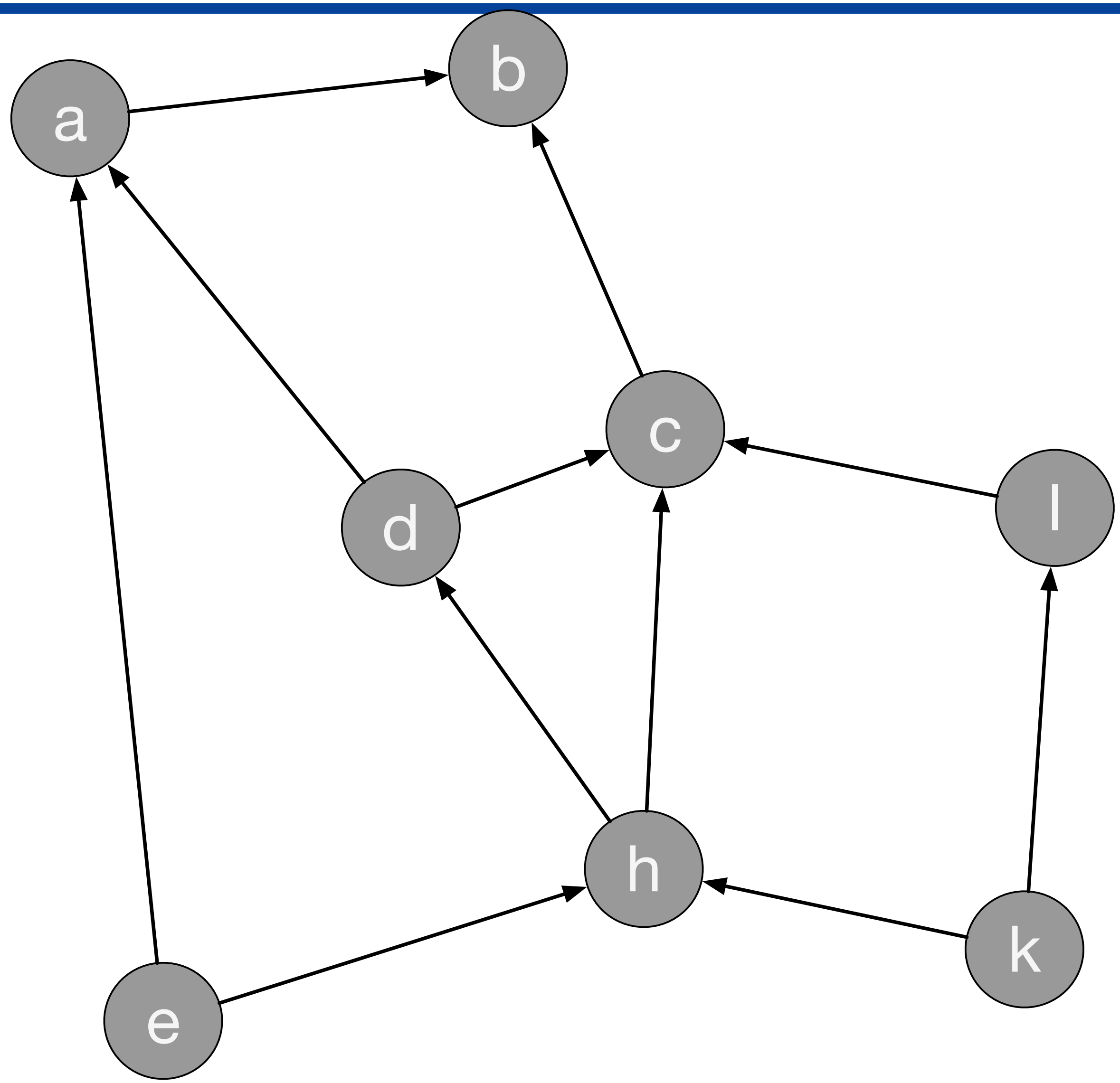
- Un graphe orienté G=(S,A) admet une partition en niveaux si et seulement s'il est sans circuit
  - Dans ce cas, une telle partition en niveaux $\{S_i\}^t_{i=0}$ (t>=0) est unique et vérifie $\forall$x $\in$ S_i, r(x)=i
- Preuve en exercice

Partition en niveaux via l'algorithme circuits-niveaux

Algo CIRCUIT-NIVEAUX (Données : E, Γ, Γ^{-1} ; Résultat : $E_i \subset E, CIRC$)

$E_0 = \emptyset, N = 0, i = 0$ // N est le nombre de sommets traités
for all $x \in E$ **do**
    $d^-(x) = |\Gamma^{-1}(x)|$ — *Degré entrant*
end for
for all $x \in E$ tel que $d^-(x) = 0$ **do**
    $E_0 = E_0 \cup \{x\}$ — *Les sources*
    $N = N + 1$
end for
while $N < |E|$ et $E_i \neq \emptyset$ **do** — *Traitement de tous les sommets*
    $E_{i+1} = \emptyset$
    for all $x \in E_i$ **do**
        for all $y \in \Gamma(x)$ **do** — *Voisins des sommets précédemment traités*
            $d^-(y) = d^-(y) - 1$
            if $d^-(y) = 0$ **then** — *Si plus d'arcs entrants -> à traiter*
                $E_{i+1} = E_{i+1} \cup \{y\}$
                $N = N + 1$
            end if
        end for
    end for
    $i = i + 1$
end while
if $N < |E|$ **then** — *A-t-on tous les sommets ou non ? = est-on revenu au point de départ*
    $CIRC = VRAI$
else
    $CIRC = FAUX$
end if

Exemple



Bellman

- Algorithme de Bellman
  - Partition en circuits niveaux / tri topologique puis calcul des distances
  - Entrées : réseau R=(S,A,L) sans circuit, tel que les sommets de S soit triés par rangs croissants, A les arcs
    - ❖ x_1 désigne un sommet de rang 0 (au moins un)
  - Sorties : longueurs $\boldsymbol{\pi}_{x1}$: S → $\mathbb{R} \cup \{+\infty\}$
    - ❖ Un sommet peut être à une distance infinie de x1 (non connexe)

$\forall i \in [2, |S|], \pi_{x1}(i) = +\infty, \ \pi_{x1}(x_1) = 0$

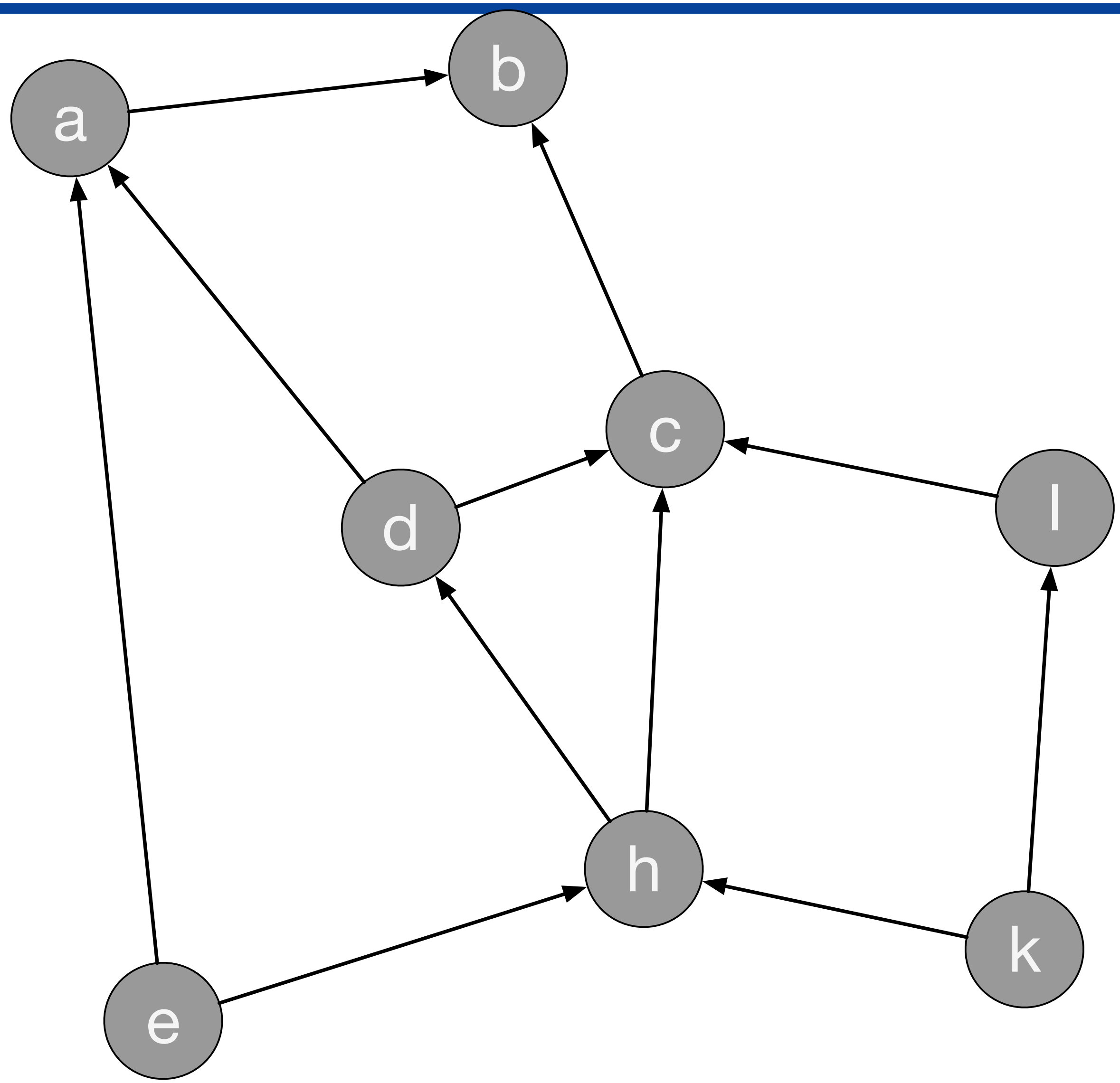
j=2

Tant que j $\leq n$

$$\pi_{\mathrm{x}1}(\mathrm{x_j}) = \min\{\pi_{x1}(y) + L\left(\left(y, x_j\right)\right), y \in \mathrm{A}^{-1}(x_j)\}$$

- $A^{-1}(x)$ dénote les sommets ayant un arc pointant vers x
- Complexité en O(|A|+|S|)

Exemple



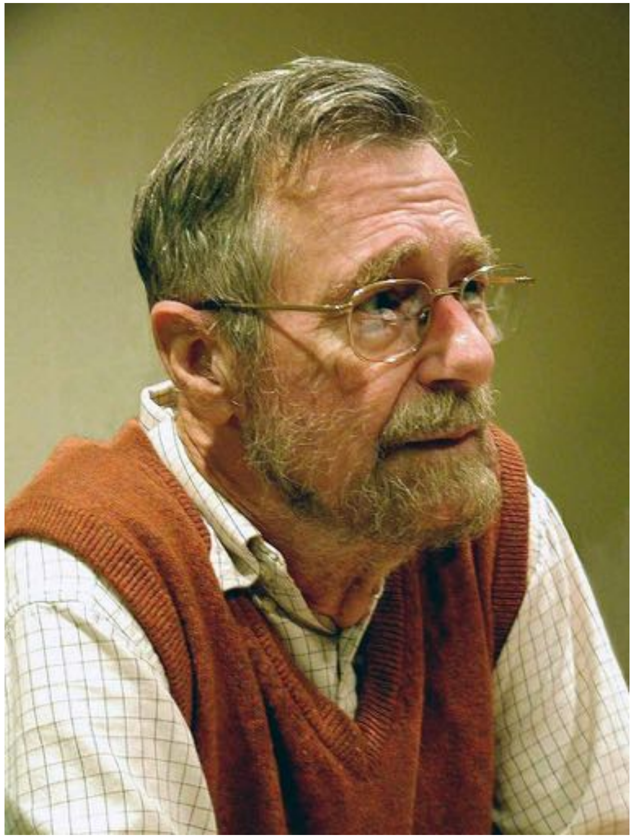
Chemins dans les réseaux à longueur positive

- Définition / contexte
  - Un réseau R=(S,A,L) est à longueurs positives si L(x) >= 0 pour tout x dans S
  - Les assertions suivantes sont équivalentes :
    - ❖ il existe un chemin de x à y dans R;
    - ❖ il existe un plus court chemin de x à y dans R.
  - Autrement dit : il n'existe pas de cycle

- Propriété
  - R=(S,A,L) un réseau à longueurs positives
  - x un sommet de R
  - S'⊆S un ensemble de sommets y pour lesquels on connait la longueur $\pi_x(y)$ d'un plus court chemin depuis x
  - S'' ⊆ S\S' l'ensemble des sommets de S\S' successeurs d'un sommet dans S'
  - Pour tout sommet z ∈ S'', on note $\boldsymbol{\pi}'_x(z)$ la longueur d'un plus court chemin de x à z dont tous les sommets (sauf z) sont dans S'.
  - Alors, si z ∈ S'' est tel que $\pi'_x(z) = \min_{w \in S''}\{\pi'_x(w)\}$
    - ❖ on a $\pi'_x(z) = \pi_x(z)$
    - ❖ Preuve ?

L’algorithme de Dijkstra

- Edsger Dijkstra (1930 – 2002)
  - Prix Turing 1972
- Contributions
  - 1950 : participation au développement d’Algol
  - 1959 : redécouverte (indépendante) de l’algo de Prim
  - 1959 : algo de plus court chemin
    - Utilisé dans A*, OSPF, etc.
  - 60’s : système d’exploitation et les bases du software design
  - 1965 : problème de l’exclusion mutuelle (sémaphores, etc.) avec la première solution correcte
  - 1968 : instruction GOTO
  - 1972 : auto-stabilisation
  - 1976 : vérification formelle



Dijkstra

Algo DIJKSTRA (<u>**Données :**</u> $E, \Gamma, l, i \in E$; <u>**Résultats :**</u> π, S)

$S = \{i\}$, $\pi(i) = 0$, $k = 1$, $x_1 = i$
for all $x \in E \setminus \{i\}$ **do**
    $\pi(x) = \infty$
end for

Initialisation : distance infinie

while $k < n$ et $\pi(x_k) < \infty$ **do**
    for all $y \in \Gamma(x_k)$ tel que $y \notin S$ **do**

Tous les voisins faisant partie du graphe non exploré

        $\pi(y) = \min[\pi(y), \pi(x_k) + l(x_k, y)]$

distance de y : celle qu'il avait avant ou en passant via x_k

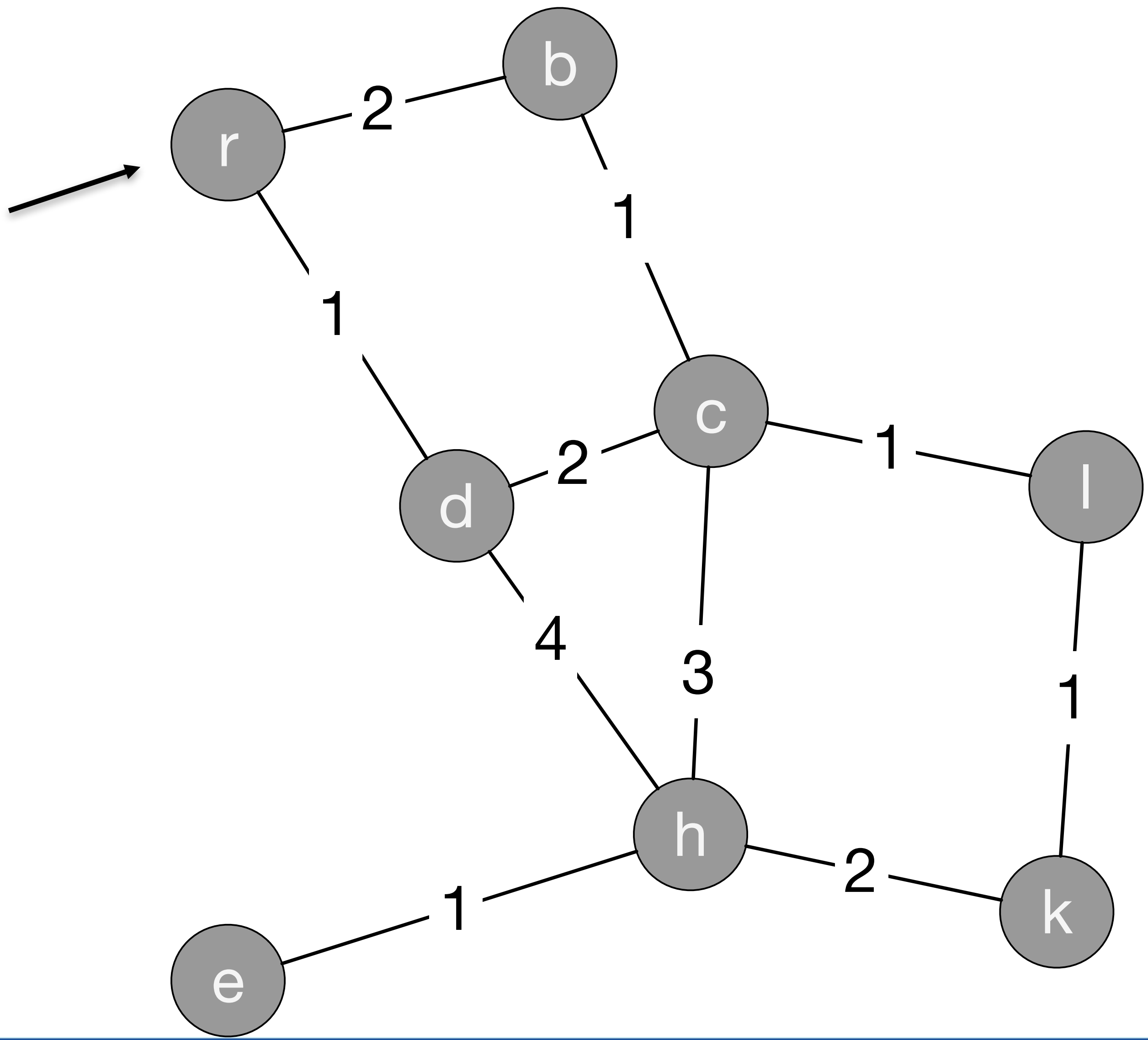
    end for
    Extraire un sommet $x \notin S$ tel que $\pi(x) = \min\{\pi(y), y \notin S\}$
    $k = k + 1$, $x_k = x$, $S = S \cup \{x_k\}$.

Le prochain sommet traité est celui non-traité ayant la + petite distance

end while

Michel COUPRIE ©

Exemple



Chemins dans les réseaux à longueur quelconque

- **Propriété fondatrice**
  - Soit R=(S,T,L) un réseau et x, y deux sommets de R tels qu'il existe un plus court chemin de x à y dans R.
  - On a alors $\pi_x(y) = \min_{z \in T^{-1}(y)} \{\pi_x(z) + L(z,y)\}$
- **k-chemins et propriétés générales (n=|S|)**
  - On appelle k-chemin (k $\in \mathbb{N}$) de x à y dans R tout chemin comprenant au maximum k sommets
    - ❖ Un plus court k-chemin C de x à y dans R est un k-chemin de x à y tel que pour tout k-chemin C' on a L(C) <= L(C').
  - On définit $\boldsymbol{\pi}_x^k$: S -> $\mathbb{R} \cup \{+\infty\}$ de telle sorte que pour tout y $\in$ S, on a $\boldsymbol{\pi}_x^k(y)$=L(C) s'il existe un plus court k-chemin de x à y dans R et +∞ sinon.
- Il existe toujours un plus court n-1-chemin de x à y dans R s'il existe un chemin entre x et y
- Pour tout y $\in$ S, si $\boldsymbol{\pi}_x^{n-1}(y) = +\infty$ alors il n'existe pas de chemin de x à y dans R
- Pour tout y $\in$ S, si $\exists k \geq 0 \mid \pi_x^{n-k}(y) > \pi_x^n(y)$, alors il existe un circuit absorbant dans R
- s'il existe k $\in \mathbb{N}$ tel que $\forall$y$\in$S on a $\boldsymbol{\pi}_x^k(y) = \boldsymbol{\pi}_x^{k+1}(y)$ alors $\forall$z$\in$S tel que $\boldsymbol{\pi}_x^k(z) \neq +\infty$, il existe un plus court chemin de x à z de longueur $\boldsymbol{\pi}_x(z) = \boldsymbol{\pi}_x^k(z)$

Bellman-Ford

Algo BELLMAN (**Données :** $E, \Gamma^{-1}, l, i \in E$; **Résultats :** $\pi, CIRCABS$)

$CIRCABS = FAUX$; $\pi^0(i) = 0$; $\pi^1(i) = 0$; $k = 1$
for all $x \in E \setminus \{i\}$ **do**
  $\pi^0(x) = \infty$
  $\pi^1(x) = \infty$

Initialisation : distance infinie

end for
for all $x \in E$ **tel que** $i \in \Gamma^{-1}(x)$ **do**
  $\pi^1(x) = l(i,x)$

Liens sortants de x → cout du lien = cout de la route

end for
while $k < n+1$ **et** $\exists x \in E$ **tel que** $\pi^k(x) \neq \pi^{k-1}(x)$ **do**

Tant que la lg de la route n'est pas max
Et changement d'une distance au -

  $k = k+1$
  for all $x \in E$ **do**
    $\pi^k(x) = \min\left[\pi^{k-1}(x), \pi^{k-1}(y) + l(y,x); y \in \Gamma^{-1}(x)\right]$

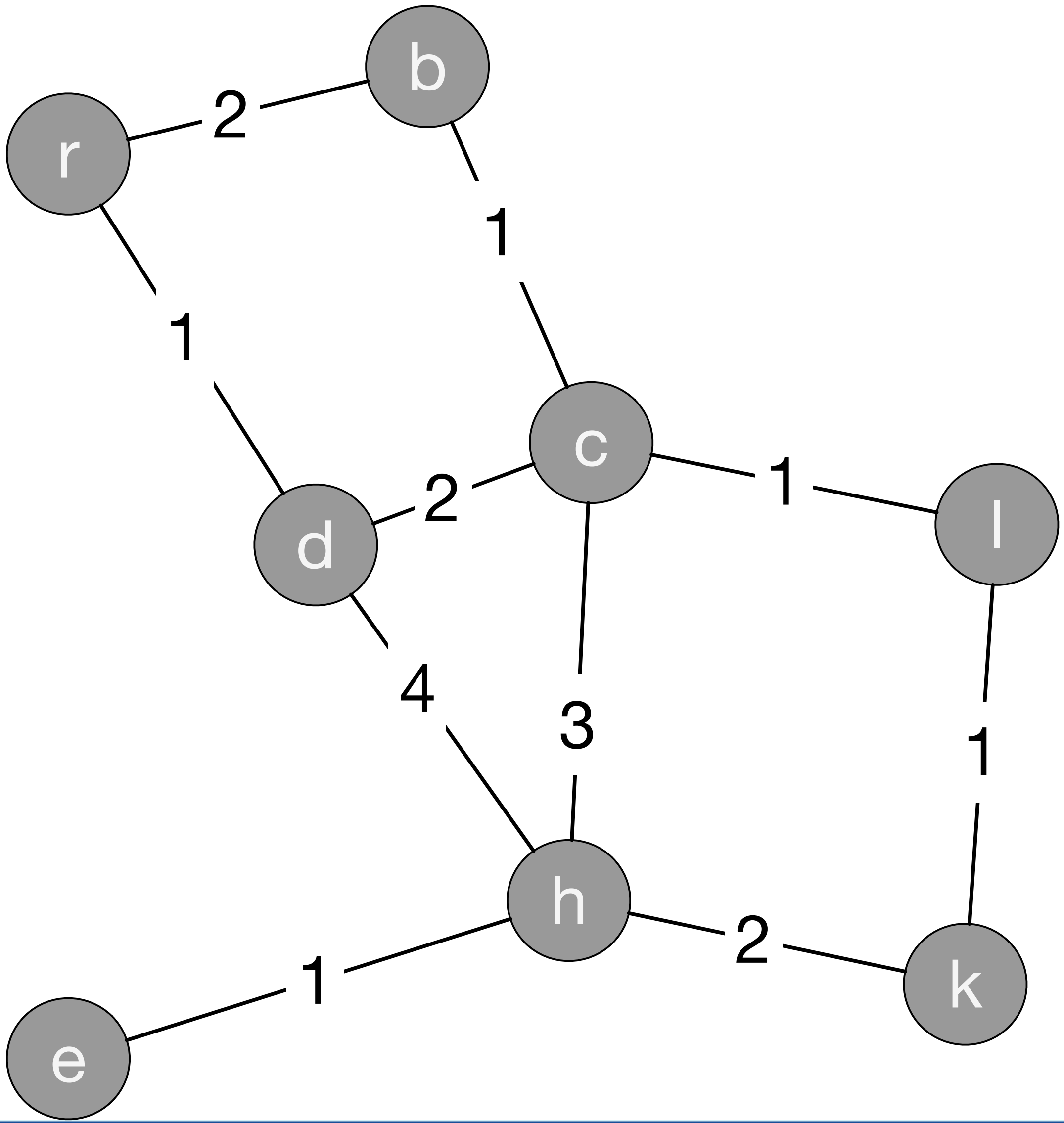
Pour chaque sommet, regarde le voisin
Qui lui offre la plus courte route

  end for
end while
if $k = n+1$ **then**
  $CIRCABS = VRAI$

Aie, il existe un circuit absorbant !

end if
$\pi = \pi^k$

Exemple



Synthèse : plus courts chemins

- Réseaux sans circuits
  - Bellman : linéaire en max(|S|,|A|)
  - ordonnancement systèmes de taches

- Réseaux à longueurs positives
  - Disjktra, gestion des queues
    - ❖ Tas de Fibonnacci : |S|log|S| + |A|
    - ❖ Tas binaire : |S|log|S| + |A|log|S|
    - ❖ Liste simple : |S|^2 + |A| si simple liste
  - Longueur fixe & unique: linéaire en |A| (BFS)

- Réseaux à longueurs quelconques
  - Bellman-Ford : |S||A|, le plus cher
  - application moins fréquente sauf en réseau : routage distribué à vecteur de distance

- Si présence de circuits absorbants → pas de + courte route

ALLONS PLUS LOIN

Quels algorithmes modernes?

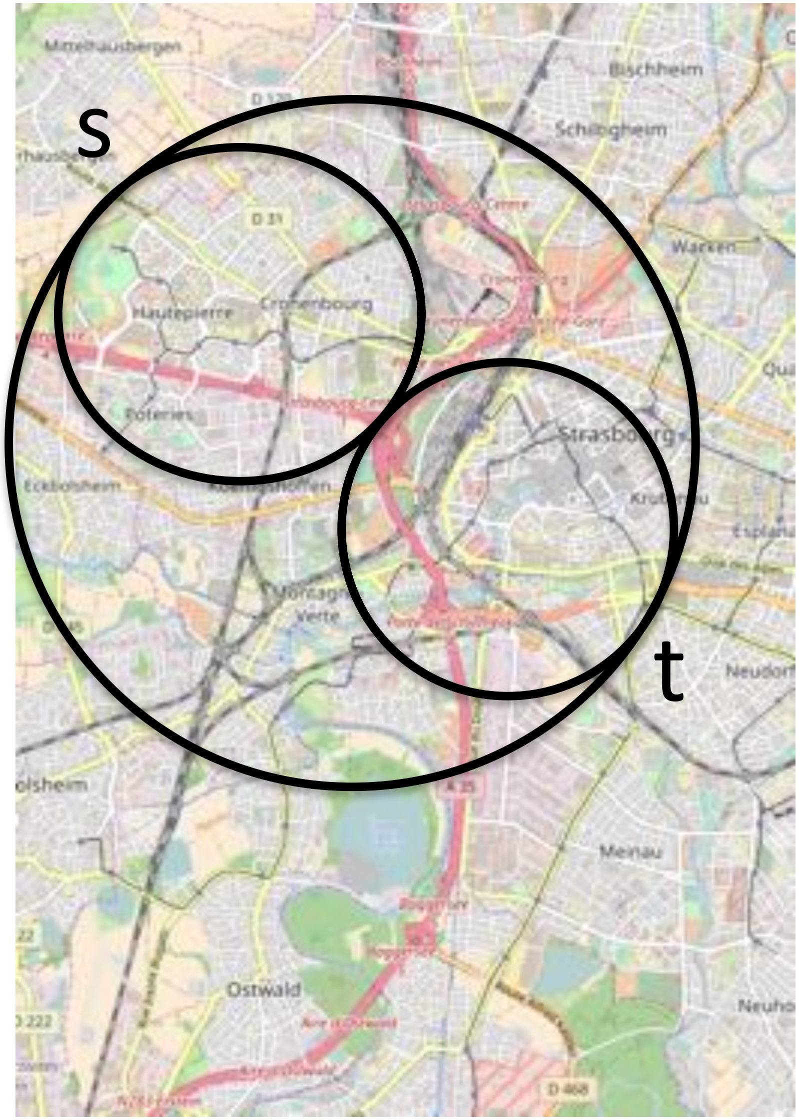
Grands graphes

- Les exemples jusqu'à maintenant : quelques dizaines de sommets /arêtes au plus
  - O(n^k) ou O(m^k) rapide

- Réseaux routiers
  - Berlin : 400k / 1M
  - Allemagne : 5M / 11M
  - Europe : 18M / 42M
- Comment calculer efficacement ?
  - Mémoire
  - Rapidité

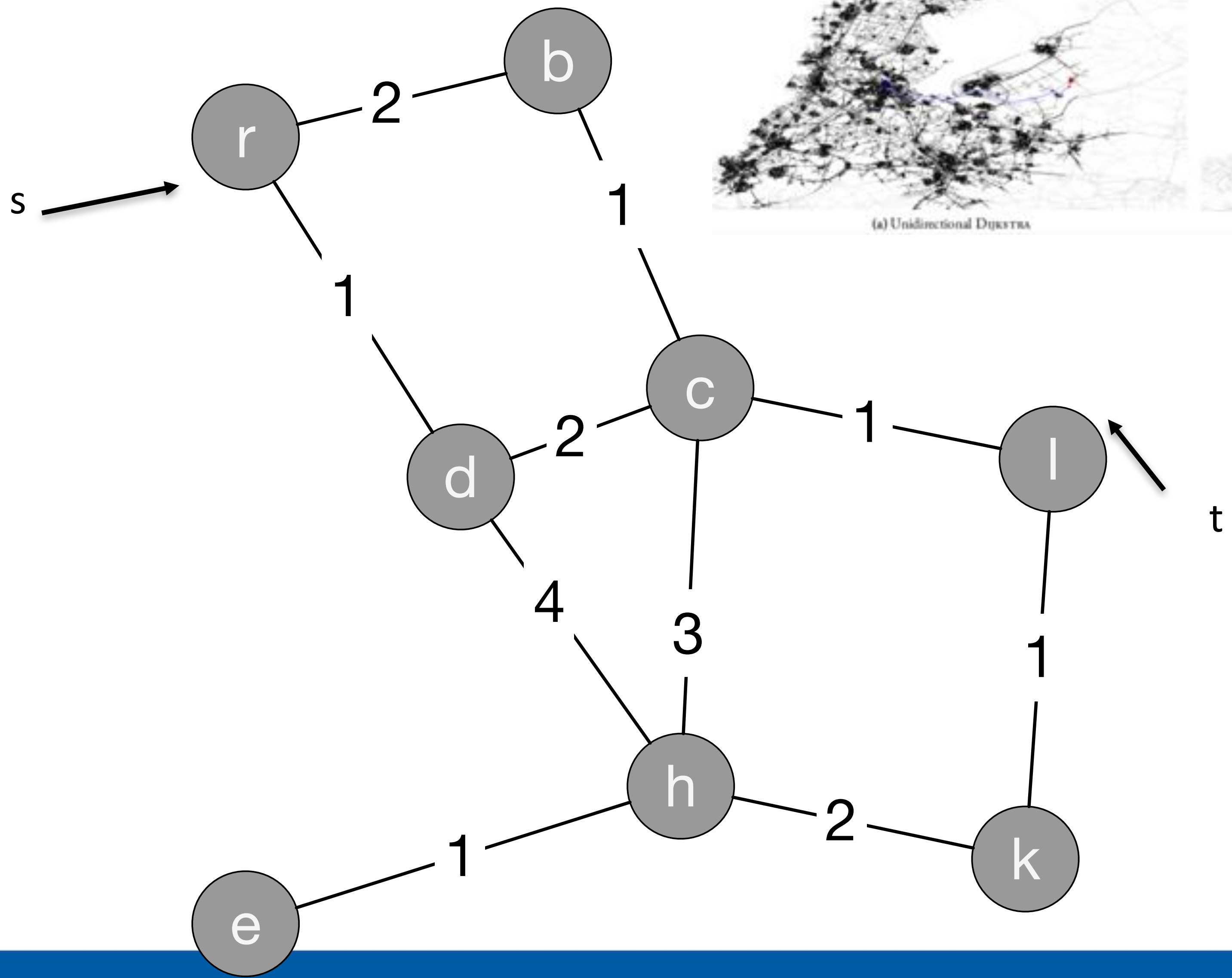
Dijkstra bidirectionnel

- Constat
  - Exploration de tout le graphe : pourquoi ne pas partir de s **et** t pour essayer de les relier ?
  - Souvent O(b^d) → transformé en O(b^d/2)
    - b : branching factor du graphe
- Algo
  - Alterner
    - Dijkstra en partant de s (forward), d_f
    - Dijkstra en partant de t (backward), d_b
  - Condition d'arrêt : exploration du même nœud
    - Suppression des 2 queues (back & forward)
    - Extraire le sommet qui min la somme des deux distance d_f + d_b




```
Test 8 : comparaisons du nombre d'étapes
Loading road networks from file USA-road-d-NY.gr ... Done
longueur chemin forward entre 190636 et 187333 = 35478 nombre d'étapes = 1252
longueur chemin backward entre 190636 et 187333 = 35478 nombre d'étapes = 1204
longueur chemin bidijkstra entre 190636 et 187333 = 35478 nombre d'étapes = 621
```

Dijkstra bidirectionnel



(a) Unidirectional Dijkstra (b) Bidirectional Dijkstra

Recherche dirigée ou A*

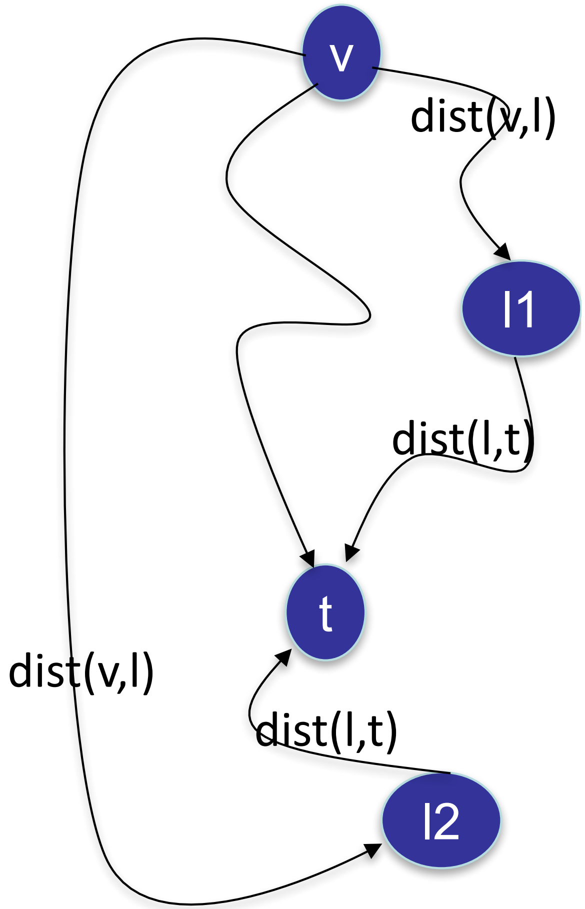
- Stratégie
  - Explorer les chemins les plus *prometteurs*
    - ❖ Modifier la priorité des sommets dans la queue
    - ❖ Exploiter une heuristique d'estimation de distance

- Approche
  - Soit un graphe orienté pondéré G(S,A,L), Soit $s, \mathrm{t} \in S$
  - Soit une fonction (heuristique) potentiel : $\rho_t : S \rightarrow \mathbb{R}_0^+$ qui associe un coût à chaque sommet
    - ❖ Estimation du coût jusqu'à la destination
  - Dijkstra modifié:
    - ❖ Remplace le coût d'une arête : $l(u,v) - \rho_t(u) + \rho_t(v)$

- Condition d'optimalité
  - ρ_t doit sous-estimer la distance à t
    - ❖ Si $\rho_t = 0$ → Dikstra classique
    - ❖ Si ρ_t trop grand → on explore pas le sommet car il est estimé « trop loin » → il faut sous-estimer !
    - ❖ Optimalement $\rho_t(v) = dist(u,v)$!

Recherche dirigée ou A*

- Comment définir $\rho_t(v)$?
  - Réseaux routiers
    - ❖ Distance euclidienne * vitesse maximale
  - Landmarks
    - ❖ Un petit nombre (fixe) de landmarks est calculé (qqs milliers)
      - – Ensemble des landmarks $\mathcal{L}$
    - ❖ Pré-calcul des routes de tous vers ces landmarks
    - ❖ $\rho_t\ (v) = \max_{l \in \mathcal{L}}\{\, dist(v,l)\ - dist(t,l); dist(l,t) - dist(l,v)\}$
      - – Preuve d'optimalité ?
        - » $\leq dist(v,t)$
    - ❖ Quel intérêt ?
      - – La mémoire : stockage de $|\mathcal{L}| * |S|$ distances sachant que $|\mathcal{L}| \ll |S|$



Incertitude & Dynamique

- Données non exactes
  - Bande-passante résiduelle d'un lien
  - Congestion d'une route
  - Temps de propagation des métriques ou même leur estimation

- Incertitude sur la pondération du graphe
  - Quel chemin pour arriver au plus tard à 10h ?
  - Quel chemin optimisant l'IQR de mon temps de parcours ?
  - Quel risque d'arriver après 11h30 ?

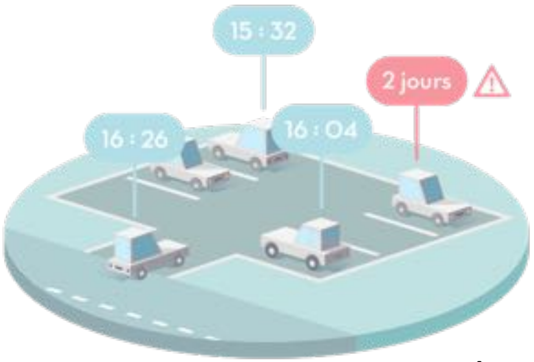
- Dynamique
  - Prévision des changements de poids
  - Quel trajet si je pars maintenant ?
    - ❖ Métrique qui dépendra de ***quand*** je passerai par cette arête

Société & startups - France

Bus à la demande
padam

temps réel:
V-traffic, HERE, waz

Optimisation de tournées / flottes
Axiodis, OpenStreet, Antsway,
you2you, toursolver



Parking
smartgrain

Véhicules électriques
Gridpocket, BeNomad, Bluenovia

Mondial