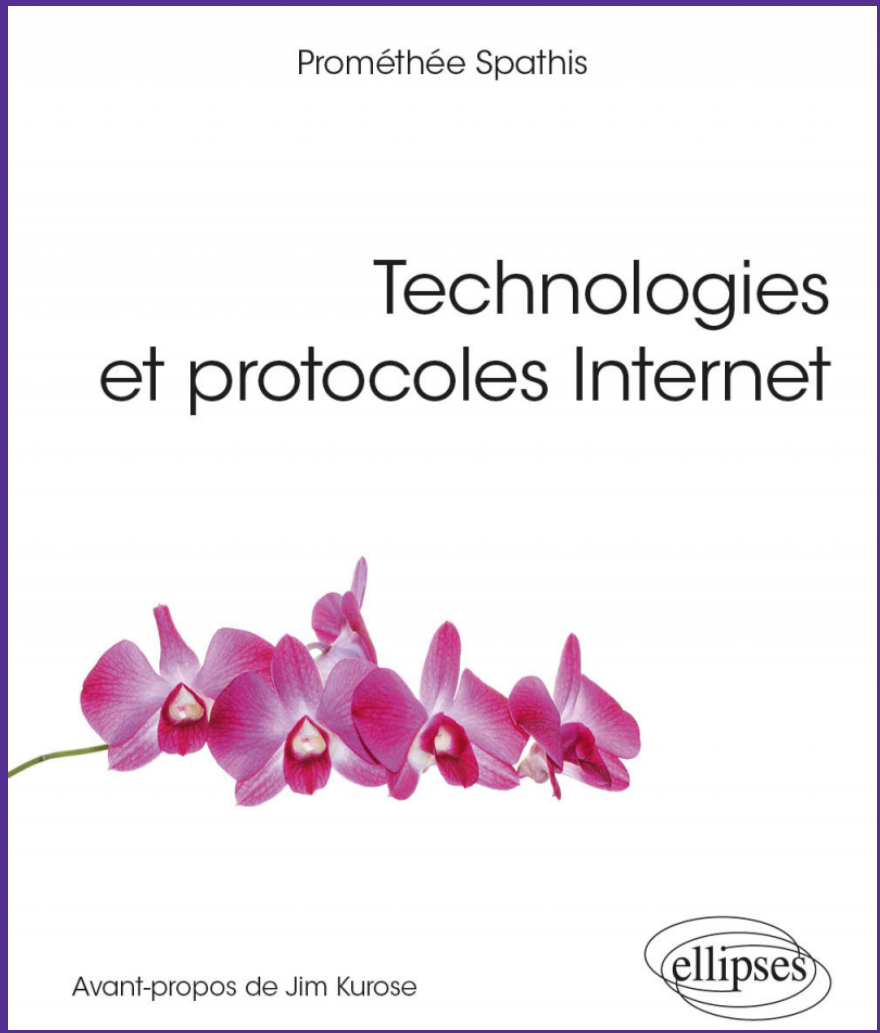
Chap 8 - Routage : RIP & OSPF

Ces transparents sont mis à disposition de tous (étudiants, enseignants, lecteurs).

En contrepartie, merci de bien vouloir :

- mentionner leur source,
- préciser la mention de copyright.

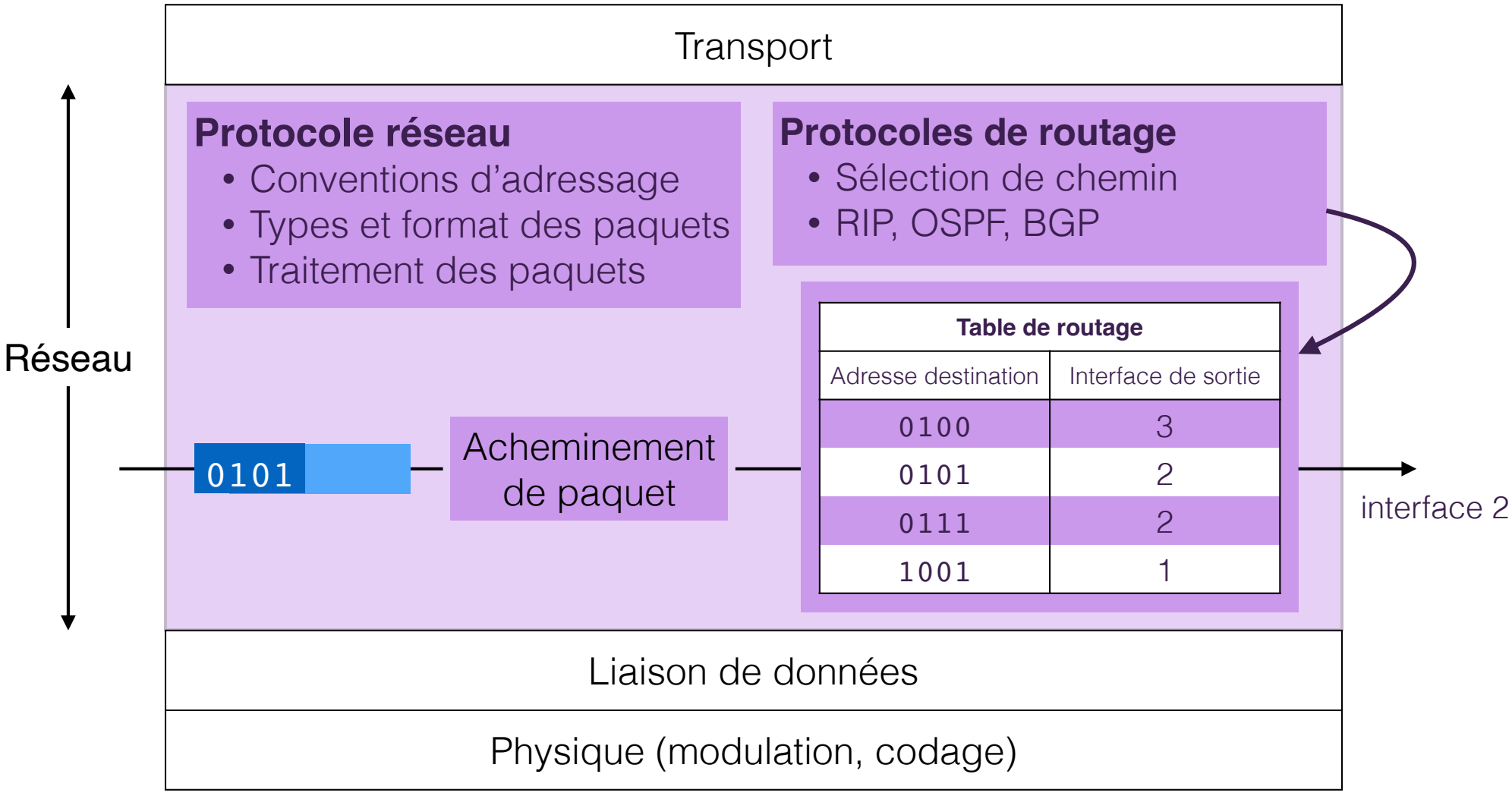
Merci et bon cours !



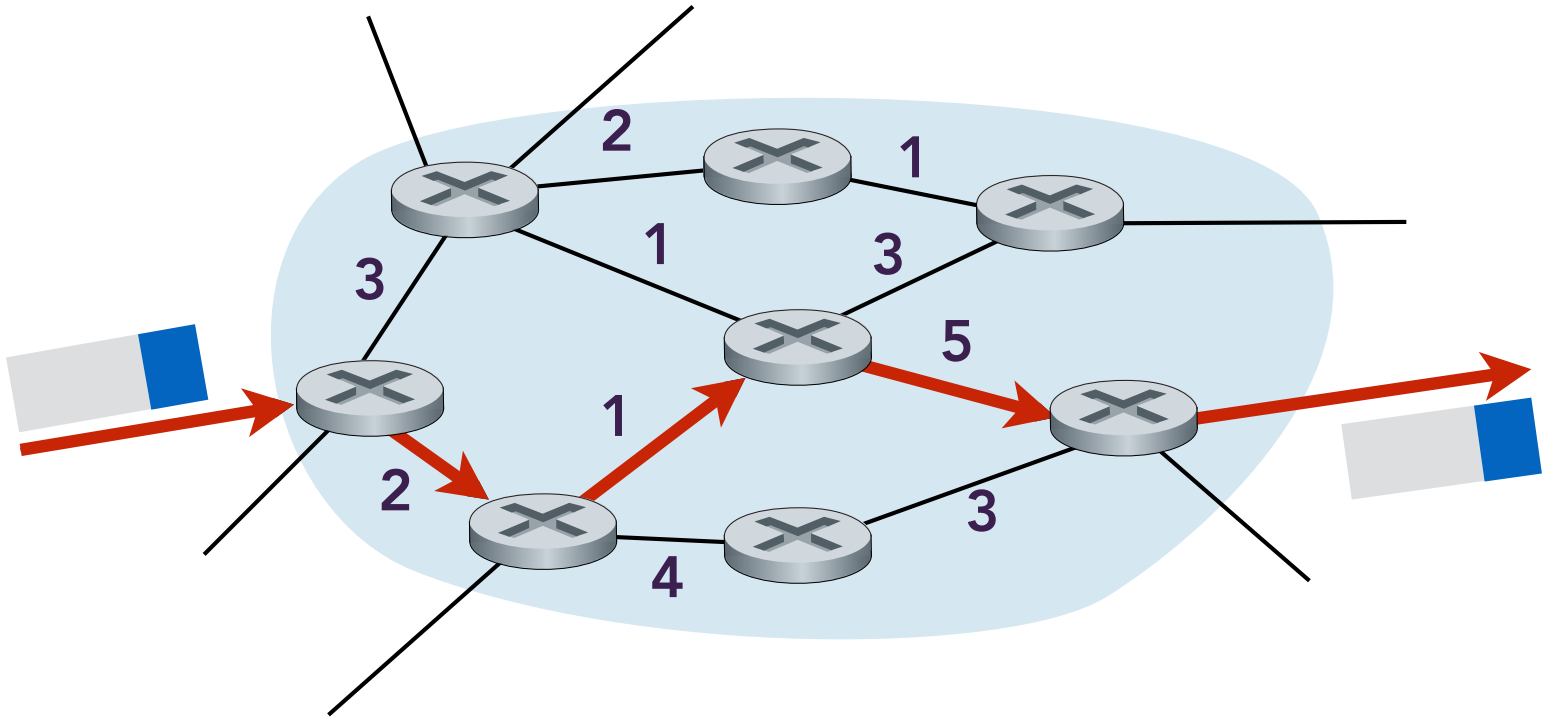
Plan du cours

- Acheminement vs routage
- Découverte de la topologie du réseau
  - **État de lien** : OSPF - *Open Shortest Path First*
  - **Vecteur de distance** : RIP - *Routing Information Protocol*
- Sélection de chemins
  - Algorithme de **Bellman-Ford :** nombre de sauts
  - Algorithme de **Dijkstra :** somme des coûts de lien
- Changements de topologie
  - Détection et propagation des changements de topologie
  - Recalcul des routes et mise à jour des tables de routage
- Architecture de routage hiérarchique à deux niveaux
  - Routage **intra-domaine** : RIP, OSPF
  - Routage **inter-domaine** : BGP

Couche réseau



Acheminement des données



Les paquets de données sont commutés (forwardés) de proche en proche vers leur destination finale

Acheminement des données

mode datagramme

- Chaque routeur maintient une table de routage
  - une entrée par adresse destination
  - chaque entrée indique l'interface de sortie pour atteindre l'adresse destination correspondante
- À la réception d'un paquet
  - un routeur inspecte l'adresse destination du paquet
  - détermine l'entrée maintenue pour cette adresse
  - achemine le paquet sur l'interface indiquée par cette entrée
- Les routeurs suivant répètent les mêmes actions
  - le paquet se rapproche de sa destination finale au fur et à mesure

D'où proviennent les tables de routage ?

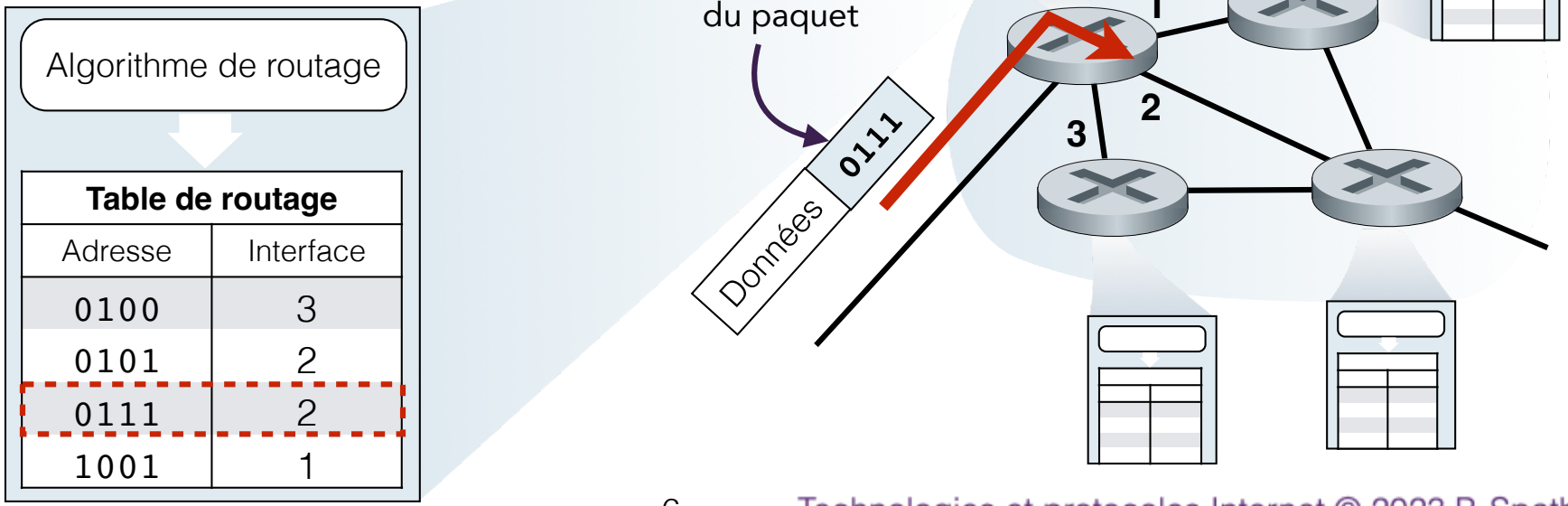
Routage vs Acheminement

Routage

- Création et maintien des table de routage
  - Calcul et mise à jour des routes
- Algorithmes de routage
  - Dijkstra, Bellman-Ford
- Protocoles de routage
  - OSPF, IS-IS, RIP, BGP,…

Acheminement

- Commutation proche en proche des paquets
  - Choix de l'interface de sortie sur laquelle aiguiller les paquets



Impact du choix des routes

- Performance de bout-en-bout
  - La qualité des chemins affecte les performances des utilisateurs
  - Métriques de performance : délai de propagation, débit, pertes, …

- Utilisation des ressources réseau
  - Répartir le trafic sur tous routeurs et les liens du réseau
  - Eviter la congestion en re-dirigeant le trafic sur des liens moins encombrés

- Perturbations suite à des changements de topologie
  - Pannes, maintenances, équilibrage de charge
  - Limiter les pertes et variations de délai pendant la convergence

Calcul des routes

Statique

- L'administrateur configure manuellement les entrées des tables d'acheminement

- Plus de contrôle
- Choix des routes et commutation basés des paramètres autres que la destination seule
- Ne passe pas à l'échelle
- Adaptation lente aux perturbations du réseau

Dynamique

Les routeurs échangent des informations de routage en utilisant un protocole de routage, un algorithme de routage calcule les routes

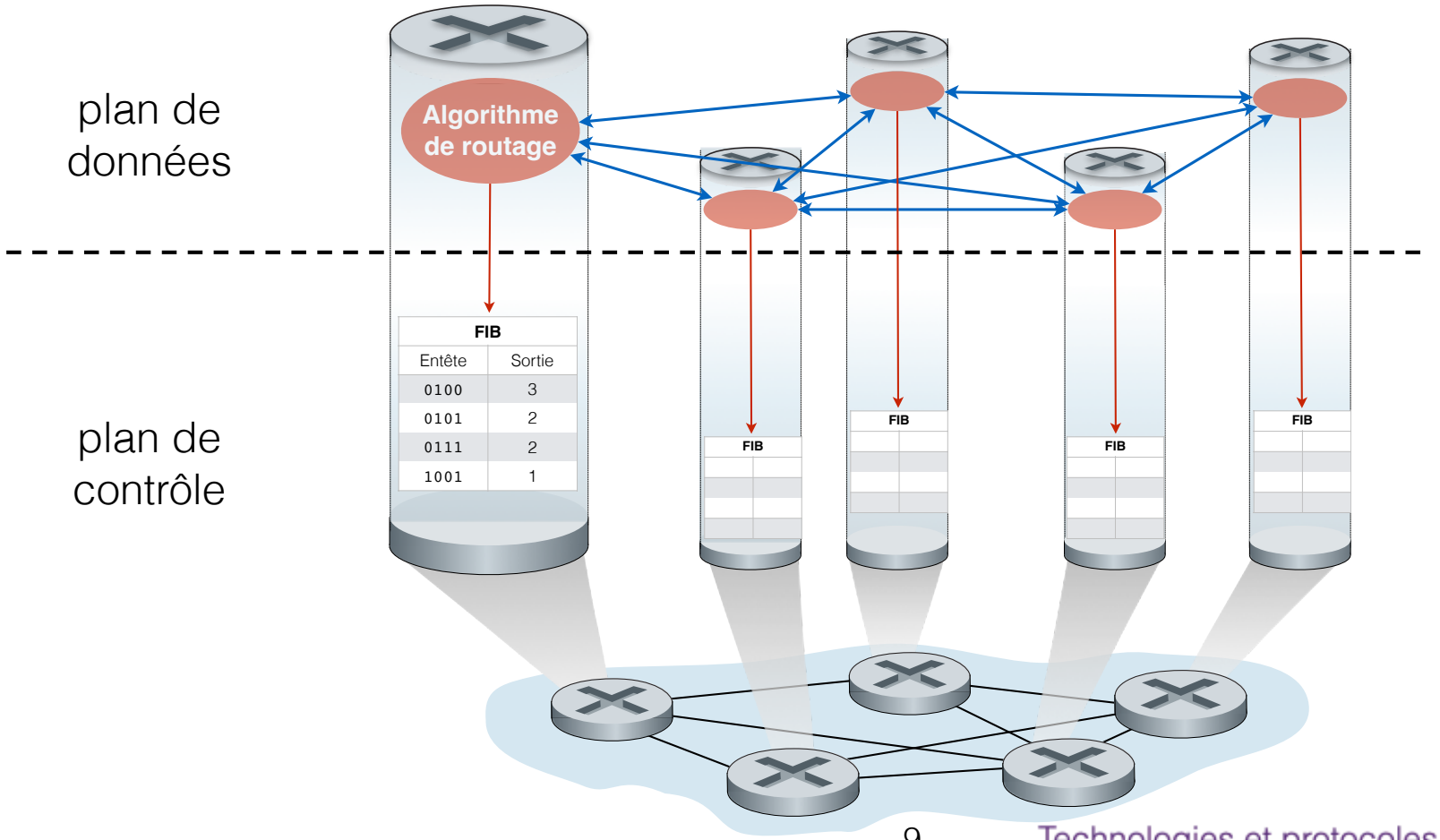
- Adaptation rapide aux changements de topologie
- Passe à l'échelle
- Algorithmes distribués complexes
- Consomme du CPU, RAM et BP
- Débogage compliqué
- Choix des routes et commutation basés uniquement sur la destination

En pratique : un mix des deux
routage statique aux extrémités, dynamique au cœur

Protocole de routage

algorithme local distribué

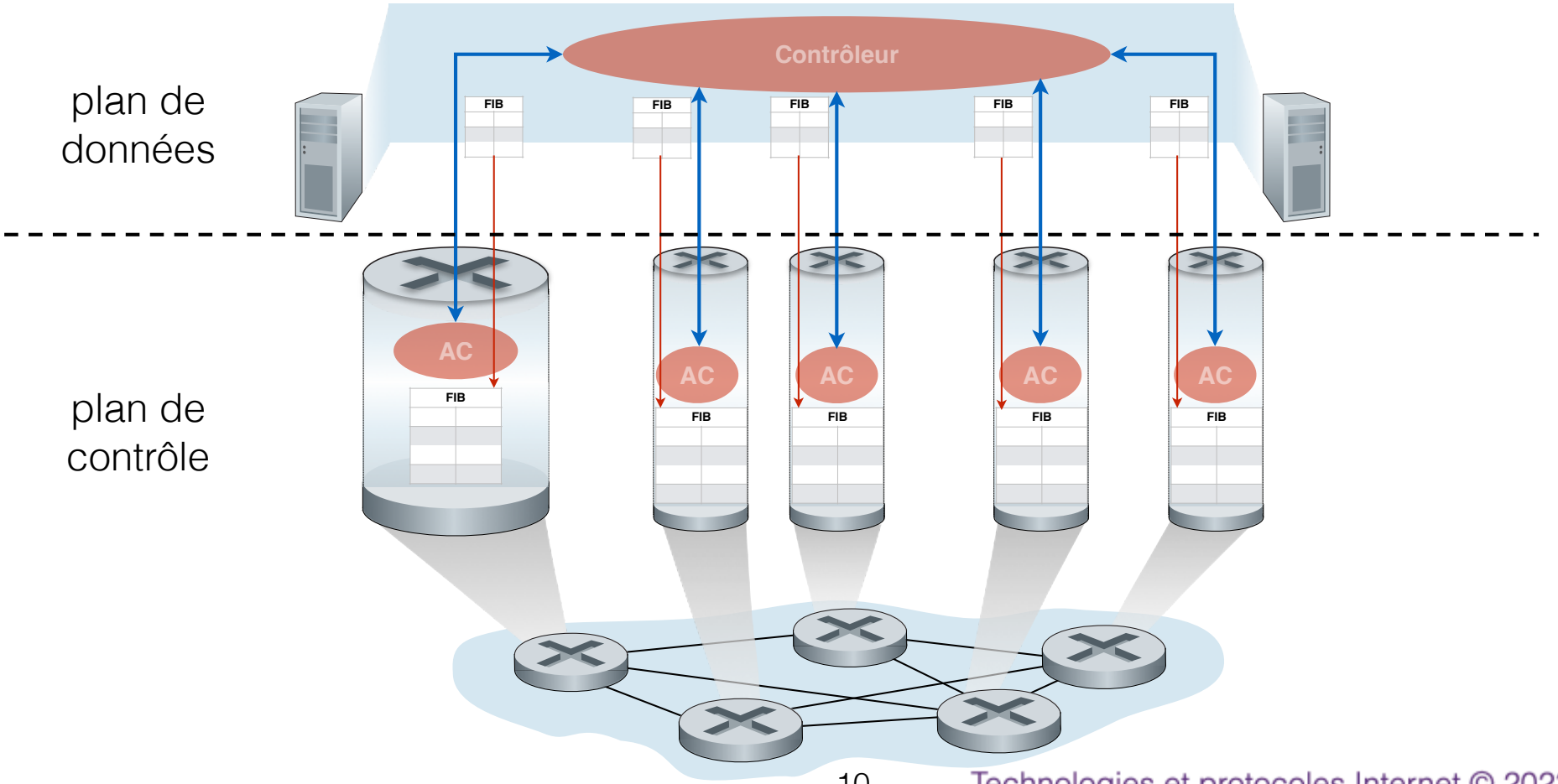
Les routeurs exécutent un algorithme local distribué qui calculent les routes incluses dans leur table d'acheminement



Protocole de routage

algorithme centralisé

Un contrôleur calcule les routes pour le compte des routeurs qu'il installe dans les tables des routeurs



Protocole vs Algorithme de routage

- Algorithmes de routage
  - calcul des "meilleurs" routes (généralement les plus "courtes")
  - pour s'exécuter, les algorithmes ont besoin de connaitre la topologie du réseau
    - Dijkstra : topologie complète
    - Bellman-Ford : topologie partielle
- Protocole de routage
  - les routeurs connaissent :
    - leurs voisins directs (par configuration manuelle)
    - le coût des liens les reliant à leur voisin (par configuration manuelle en général)
    - l'état de ces liens
  - les routeurs communiquent entre eux selon un protocole de routage :
    - type et format des messages échangées
    - ces messages contiennent des informations relatives à la topologie du réseau
  - dans le but de découvrir la topologie complète ou partielle du réseau

Routage Internet

- Sélection des routes
  - Les routeurs sélectionnent une route unique selon l'adresse de la destination
  - Critère de sélection (politique de routage) : la route la plus "courte"
    - nombre des sauts (Bellman-Ford / RIP)
    - somme des coûts des liens (Dijkstra / OSPF)
  - Le routage ne réagit pas à la charge de trafic
- Changements de topologie
  - Pannes, maintenances, changements de configuration, …
  - Convergence du protocole de routage
    - détection du changement
    - propagation de ce changement à tous les routeurs
    - recalcul des routes
    - mise à jour des tables de routage
  - Pertes et déséquencements transitoires des paquets des données

Protocoles de routage dans Internet

Etat de liens

- Les routeurs envoient périodiquement :
  - des paquets « état de liens »
  - à tous les routeurs du réseau (inondation)
- Construction d'une base de données d'état de liens
  - qui contient la topologie complète du réseau
- Algorithme de Dijkstra
  - calcul des routes complètes les plus courtes vers toutes les destinations du réseau
  - extraction du saut suivant et màj des tables de routage
- Politique de routage
  - choix du coût des liens (délai, bande passante, distance)
- Protocoles à état de lien
  - OSPF, IS-IS

Vecteur de distance

- Les routeurs envoient périodiquement
  - des messages « vecteur de distance »
  - à leurs voisins directs
- Algorithme de Bellman-Ford
  - Pour chaque destination connue ou apprise :
    - sélection du voisin ayant annoncé la route la plus courte pour une destination donnée
    - mise à jour de la table de routage si cette route est plus courte que celle déjà présente dans la table
  - Envoi du nouveau vecteur de distance si la table a changé
- Pas de politique de routage
  - distance exprimée en nombre de sauts
- Protocoles à vecteur de distance
  - RIP

Routage à état de liens

Routage à état de lien

- Chaque routeur surveille ses liens adjacents
  - envoi périodique de messages « hello »
  - pour déterminer l'état des liens : actif (up) ou pas (down)
- Chaque routeur inonde des paquets « état de liens »
  - périodiquement ou suite à un changement de topologie :
    - rupture ou réparation d'un lien
    - panne ou recouvrement d'un routeur
    - changement de coût
  - pour donner à tous les routeurs une vue complète du réseau (stockée dans une base de données)
- Chaque routeur exécute l'algorithme de Dijkstra
  - calcul du chemin complet vers toutes les destinations du réseau
  - détermination du saut suivant pour chaque destination
  - mise à jour de la table de routage

Algorithme de Dijkstra

Algorithme itératif

- Après *k* itérations, le routeur connaît les chemins de coût minimum vers *k* noeuds
- ***S*** : noeuds dont le chemin de coût minimum est connu

  Initialement, $S = \{u\}$, où *u* est le noeud source

  À chaque itération, l'algorithme ajoute un noeud à *S*
- D(*v*) : coût du chemin de la source vers le noeud *v*

  Initialement, $D(v) = c(u, v)$ pour tous les noeuds *v* adjacents à *u*

  … et $D(v) = \infty$ pour tous les autres noeuds *v* (non adjacents à *u*)

  D(*v*) est mis à jour au fur-et-à-mesure que des chemins sont calculés

Algorithme de Dijkstra

Initialisation

- $S = \{u\}$
- **Pour** tous les noeuds v :
  - **Si** v adjacent à u : $D(v) = c(u, v)$
  - **Sinon** : $D(v) = \infty$

Tant que tous les noeuds ne sont pas dans S :

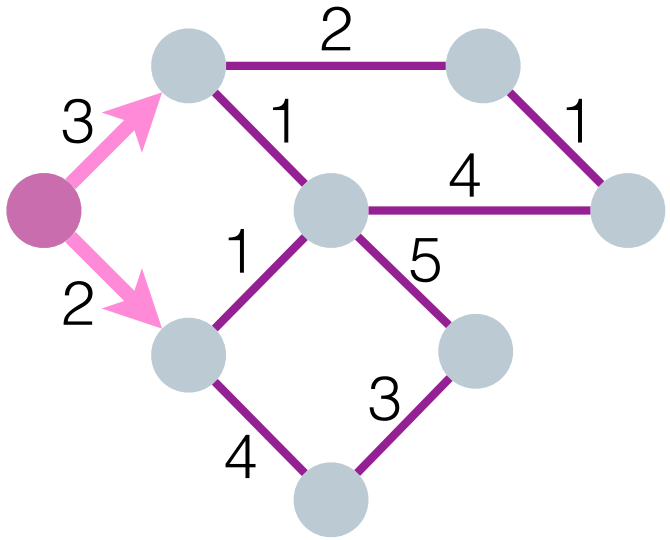
- Trouver w qui n'est pas dans S avec le plus petit $D(w)$
- Ajouter w à S :
  - $S = S \cup \{w\}$
- **Pour chaque** noeud v adjacent à w :
  - $D(v) = \min\{D(v), D(w) + c(w, v)\}$

Min-priority queue
(Fibonacci heap)

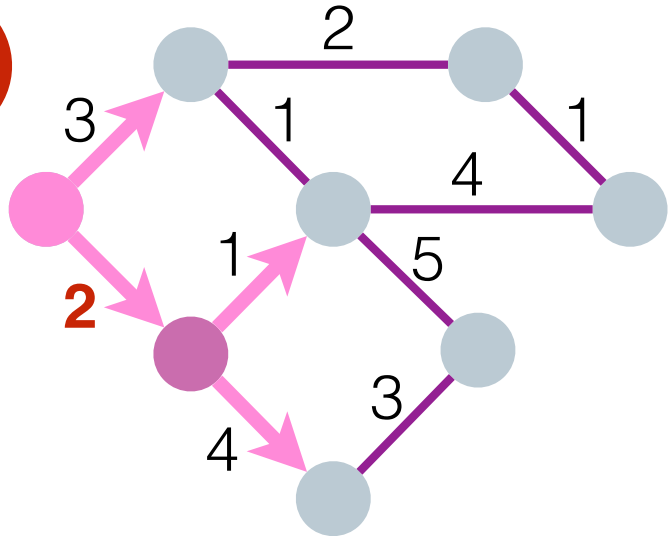
Avec *Min-priority queue* : $O(|V|^2)$, $|V|$ nombre de noeuds
Sans *Min-priority queue :* $O(|E| + |V|\log|V|)$, $|E|$ nombre d'arêtes

Algorithme de Dijkstra

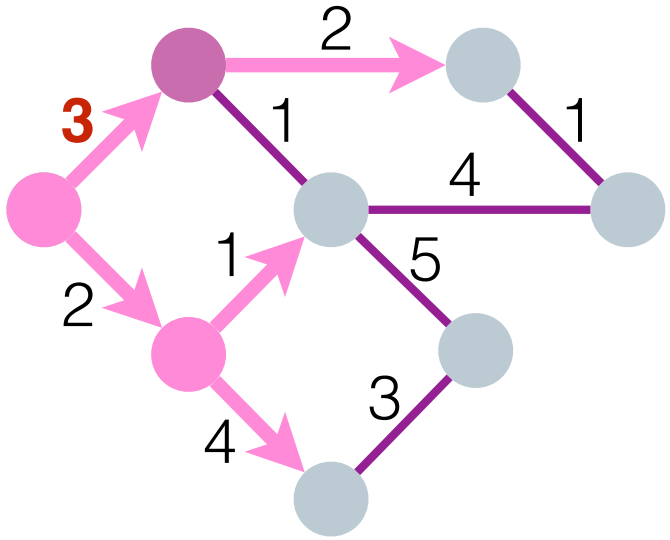
1



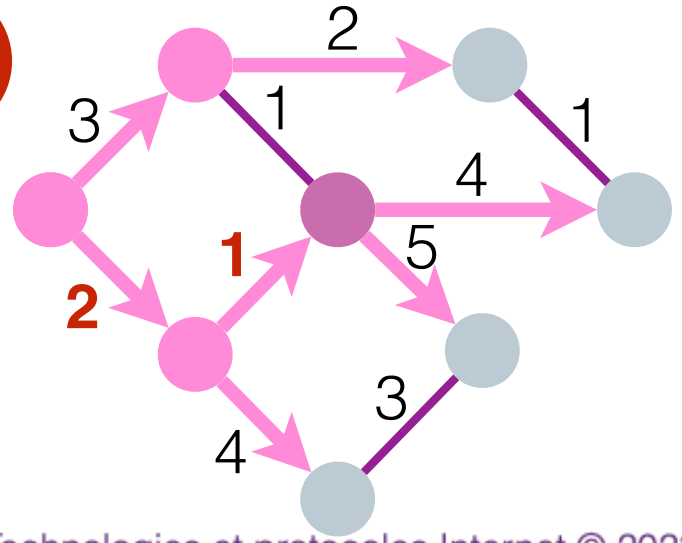
2



3

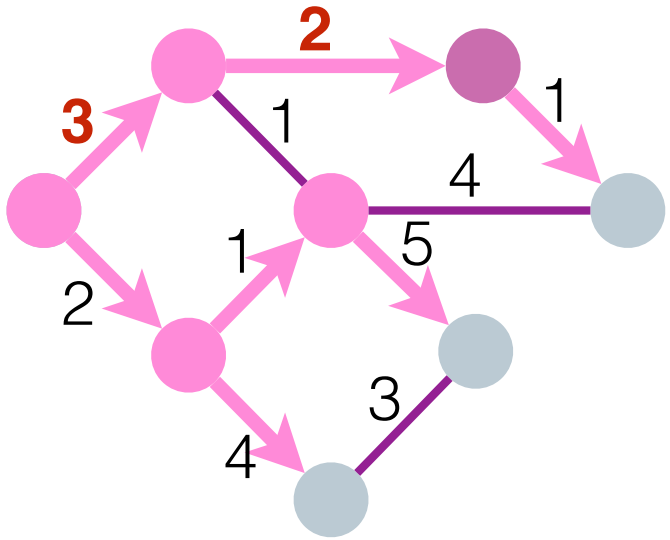


4

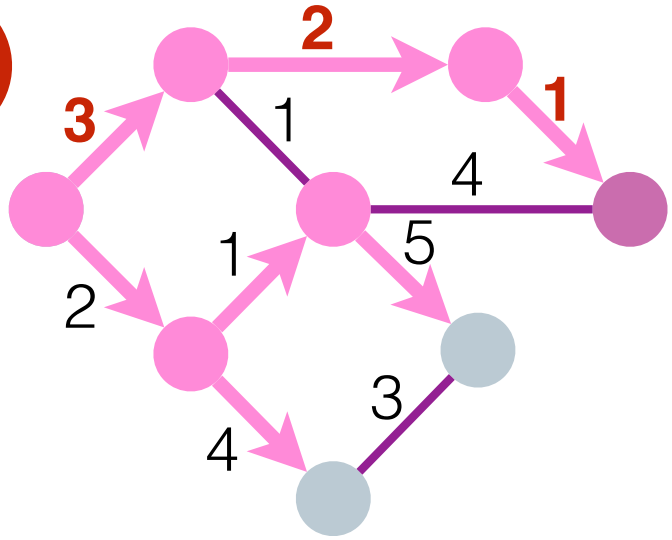


Algorithme de Dijkstra

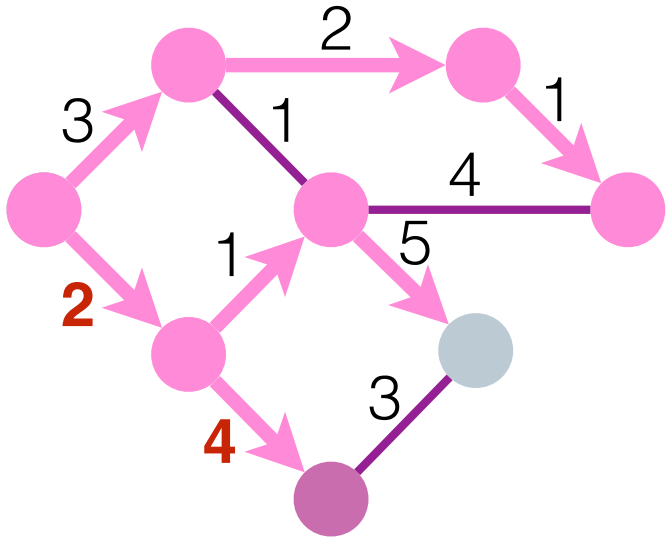
5



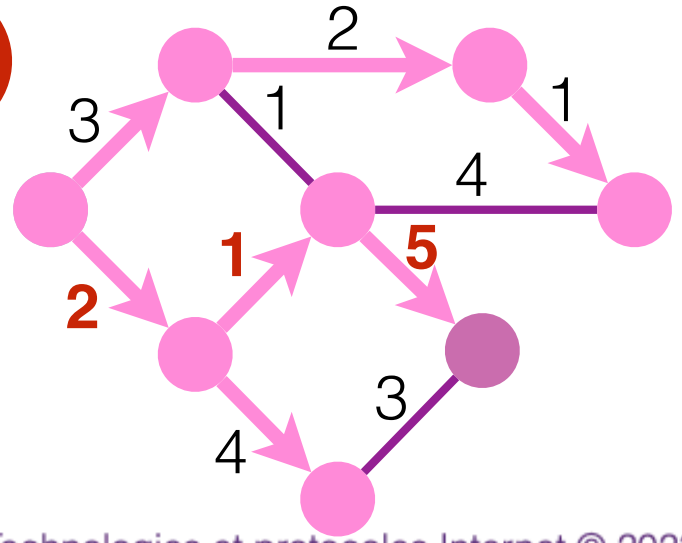
6



7

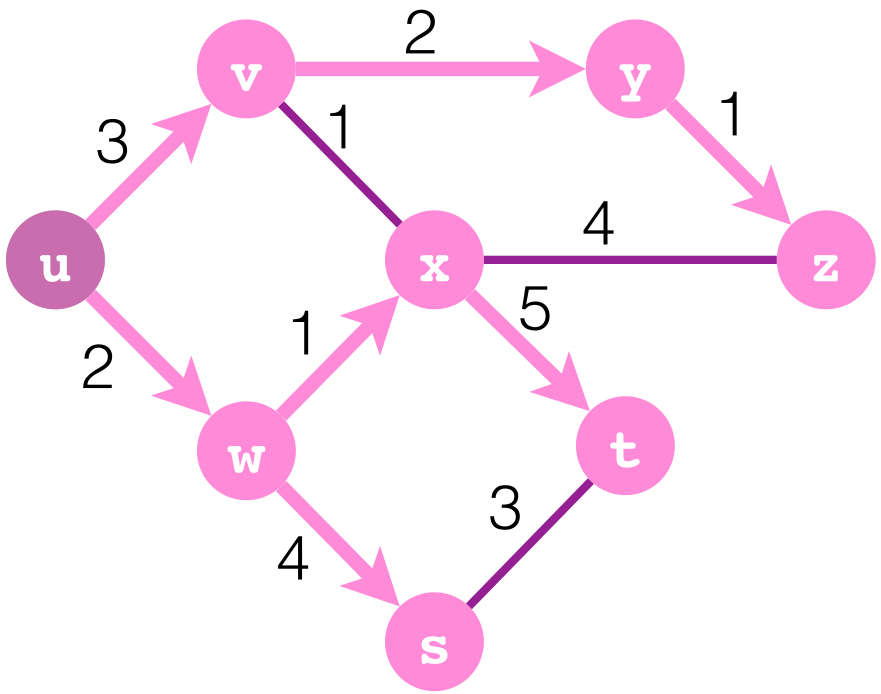


8



Arbre des plus courts chemins

Arbre des chemins les plus courts

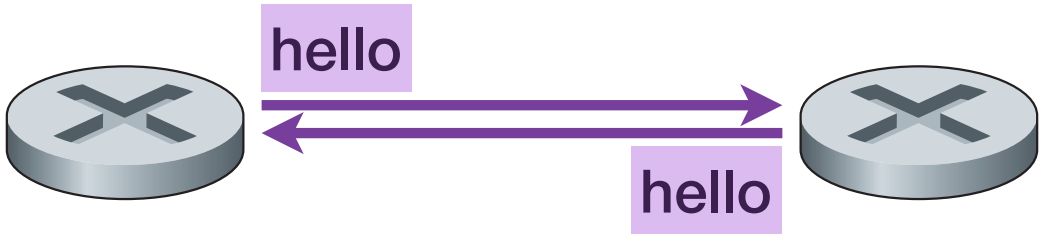


Extraction des sauts suivants

Table de routage de u	
Destination	Saut suivant
v	v
w	w
x	w
y	v
z	v
s	w
t	w

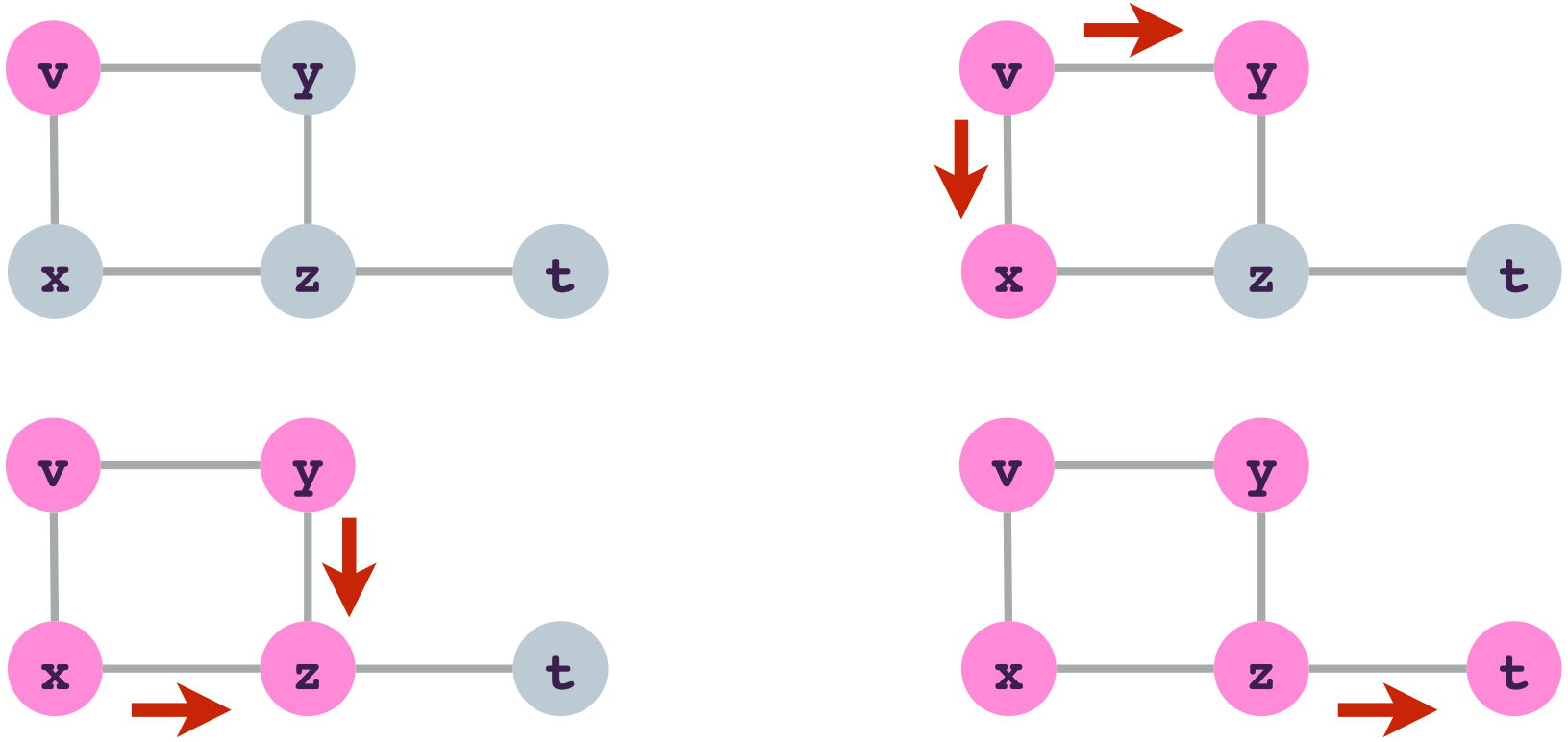
Détection des changements de topologie

- Beaconing
  - Envois périodiques de messages “hello” dans les deux directions
  - Détection d’une panne après plusieurs “hello” non reçus
- Compromis de performance (trade-offs)
  - Vitesse de détection de pannes
  - Impact sur la bande passante consommée et le CPU des routeurs
  - Détections erronées possibles



Inondation

- Inondation
  - Les routeurs envoient les paquets état de liens sur toutes ses interfaces
  - Et les routeurs suivants les renvoient sur toutes leurs interfaces,
  - … sauf sur celle par laquelle les paquets ont été reçus



Inondation fiable

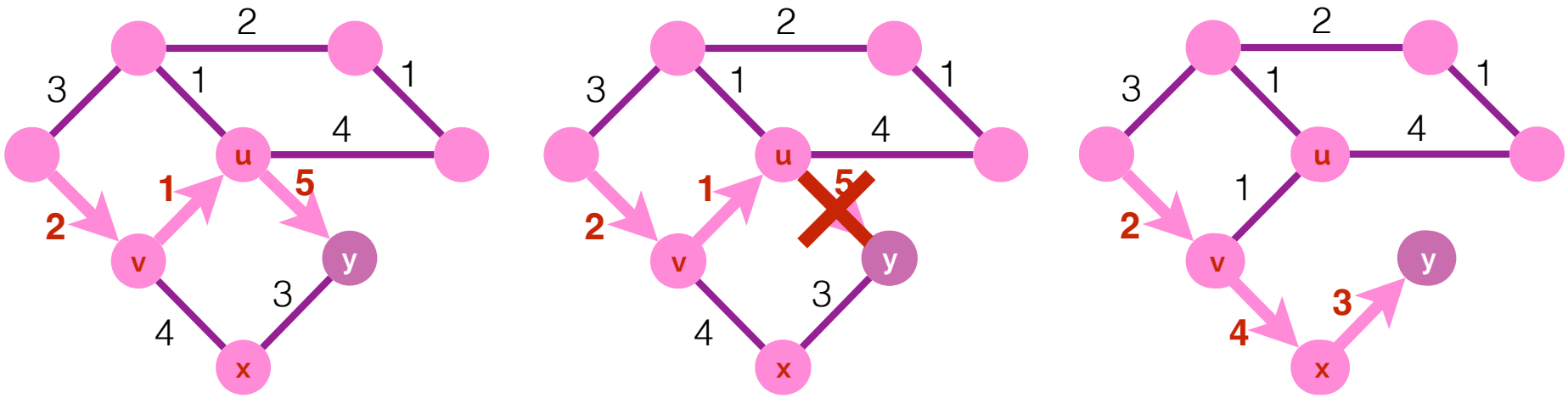
- Inondation fiable
  - Garantir que tous les routeurs ont reçu les mêmes informations de routage
  - … et que ces derniers utilisent la dernière version
- Problèmes
  - Pertes de paquets état de liens
  - Réception en désordre
- Solutions
  - Contrôle d'erreur : acquittements et retransmissions
  - Numérotation en séquence des paquets états de lien
  - Réduction de la durée de vie des paquet états de lien (Time-to-Live)

Quand inonder ?

- Changements de topologie
  - Panne de lien ou de routeur
  - Recouvrement de lien ou de routeur
- Changements de configuration
  - Changement du coût des liens
- Changements de configuration
  - Rafraîchir les informations d'état de lien
  - Généralement toutes les 30 minutes
  - Corriger les corruptions d'information éventuelles

Convergence

- Changements de topologie
  - pannes, maintenances, changements de configuration
  - convergence du protocole de routage
    - détection du changement
    - propagation de ce changement à tous les routeurs
    - recalcul des routes
    - mise à jour des tables de routage
  - pertes, déséquencement des paquets des données



Délai de convergence

- Source du délai de convergence
  - Latence de détection de la panne
  - Inondation de l'information d'état de lien
  - Calcul du plus court chemin
  - Création de la table d'acheminement
- Performance pendant la période de convergence
  - Paquets perdus dûs aux blackholes et aux expirations de TTL
  - Paquets en boucle consomment des ressources
  - Paquets reçus hors séquence
  - Non compatible avec la VoIP, les jeux en ligne ou la vidéo, …

Réduire le délai de convergence

- Détection des pannes
  - Temporisateurs de messages “hello” plus petits
  - Les technologies de niveau liaison peuvent détecter les pannes
  - Inondation plus rapide
  - Inonder immédiatement
  - Envoyer des paquets d’état de lien avec une forte priorité
- Recalcul des routes
  - Augmenter CPU sur les routeurs
  - Algorithme de Dijkstra incrémental
- Mises à jour de la table d’acheminement
  - Structures de données acceptant les mises-à-jour incrémentales

Routage à vecteur de distance

Routage à vecteur de distance

- Chaque routeur envoie des messages vecteur de distance
  - un message vecteur de distance contient :
    - la liste des destinations connues du routeur
    - la distance du chemin connu du routeur pour chacune de ces destinations
  - les messages vecteur de distance sont envoyés aux voisins directs :
    - périodiquement ou suite au changement des tables de routage
  - l'absence de messages indique la rupture du lien ou une panne du routeur voisin
- Chaque routeur exécute l'algorithme de Bellman-Ford
  - pour chaque destination connue ou apprise de ses voisins :
    - sélection du voisin ayant annoncé la distance la plus courte
  - mise à jour de sa table de routage si :
    - la route passant par ce voisin est plus courte que celle connue du routeur
    - la destination n'était pas connue

Algorithme de vecteur de distance

Equation de Bellman-Ford

Soit

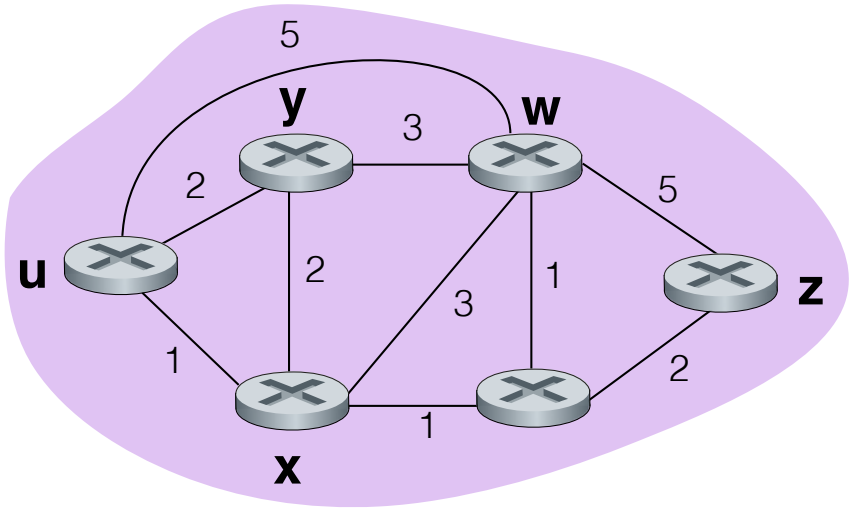
$d_x(y)$ le coût du chemin de plus petit coût de x vers y

Alors

$$d_x(y) = \min_v\{c(x,v) + d_v(y)\}$$

- v : minimum de tous les voisins v de x
- $c(x,v)$: coût vers le voisin v
- $d_v(y)$: coût depuis voisin v vers la destination y

Exemple avec Bellman-Ford



Vecteurs de distance des voisins de u concernant z

$d_v(z) = 5,\ d_x(z) = 3,\ d_w(z) = 3$

D'après l'équation de Bellman-Ford:

$$d_u(z) = \min\{c(u,v) + d_v(z),\ c(u,x) + d_x(z),\ c(u,w) + d_w(z)\}$$
$$= \min\{2 + 5,\ 1 + 3,\ 5 + 3\} = 4$$

u choisit y comme saut suivant pour la destination z

Algorithme de vecteur de distance

- $d_x(y)$ est le coût du chemin minimum de x vers y connu du routeur x
  - Les routeurs maintiennent un vecteur de distance : $\boldsymbol{D_x} = [d_x(y) : y \in N]$
- Initialement :
  - Les routeurs connaissent le coût des liens directs vers chacun de leurs voisins v : $c(x,v)$
  - Les routeurs initialisent leur vecteur de distance avec les informations concernant leurs voisins directs :

$$\boldsymbol{D}_v = [d_x(y) : y \in N]$$

- Périodiquement :
  - Les routeurs envoient leur vecteur de distance à leurs voisins directs
  - Sur réception du vecteur de distance d'un voisin, le routeur met à jour son propre vecteur de distance en utilisant l'équation de Bellman-Ford :

$$d_x(y) = \min\{c(x,v) + d_v(y)\} \text{ pour chaque routeur } y \in N$$

Algorithme de vecteur de distance

Itératif et asynchrone

- Chaque itération locale est provoquée par :

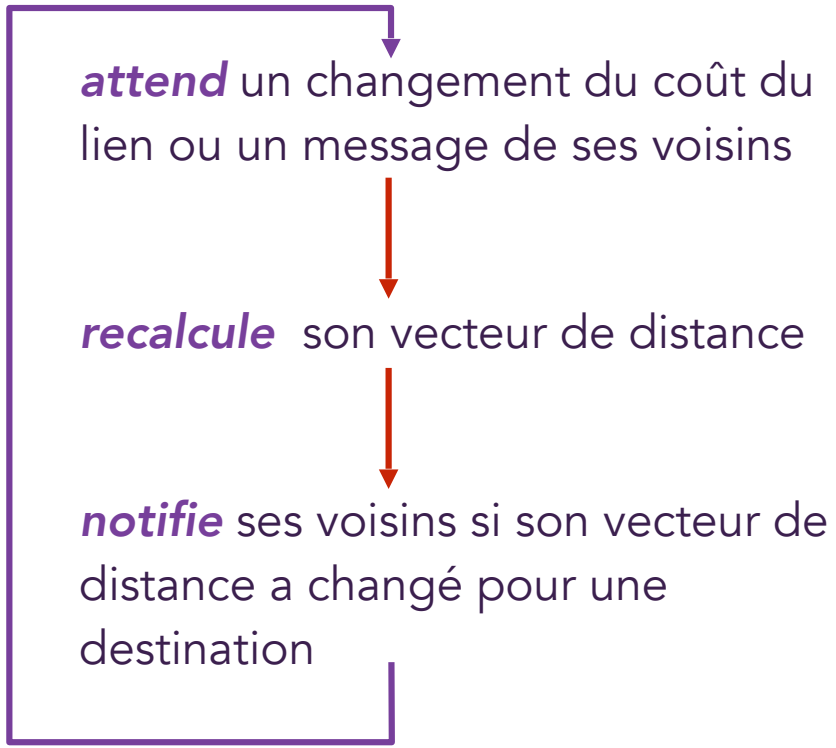
  (1) Changement du coût du lien

  (2) Mise à jour du vecteur de distance du voisin

Distribuée

- Chaque routeur notifie ses voisins que son vecteur de distance a changé
- Les voisins peuvent ensuite notifier leurs voisins si nécessaire

Chaque routeur :



Routeur x

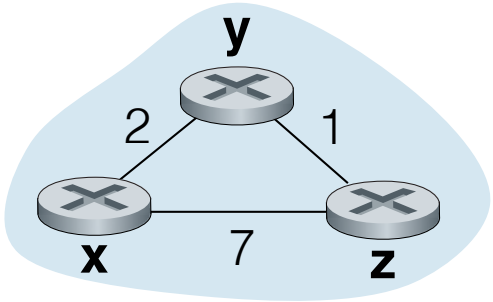
de \ coût vers	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

Routeur y

de \ coût vers	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

Routeur z

de \ coût vers	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0



temps

$d_x(y) = \min\{c(x,y) + d_y(y), c(x,z) + d_z(y)\}$
$= \min\{2+0, 7+1\} = 2$

$d_x(z) = \min\{c(x,y) + d_y(z), c(x,z) + d_z(z)\}$
$= \min\{2+1, 7+0\} = 3$

Routeur x

de \ coût vers	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

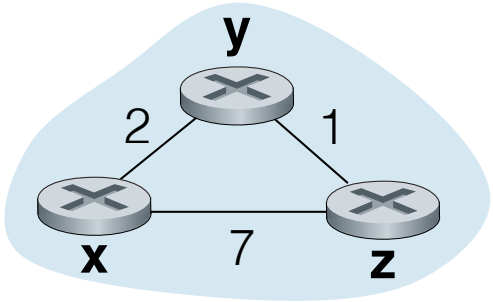
de \ coût vers	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

Routeur y

de \ coût vers	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

Routeur z

de \ coût vers	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0



temps

Routeur x

de \ coût vers	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

de \ coût vers	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

Routeur y

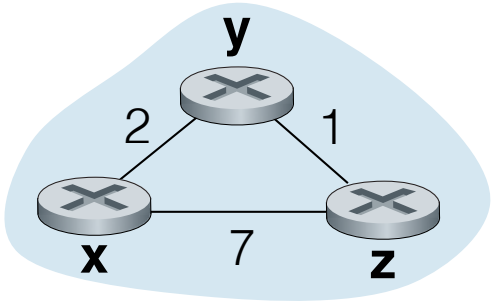
de \ coût vers	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

de \ coût vers	x	y	z
x	0	2	7
y	2	0	1
z	7	1	0

Routeur z

de \ coût vers	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

de \ coût vers	x	y	z
x	0	2	7
y	2	0	1
z	3	1	0



temps

Routeur x

de \ coût vers	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

de \ coût vers	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

de \ coût vers	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

Routeur y

de \ coût vers	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

de \ coût vers	x	y	z
x	0	2	7
y	2	0	1
z	7	1	0

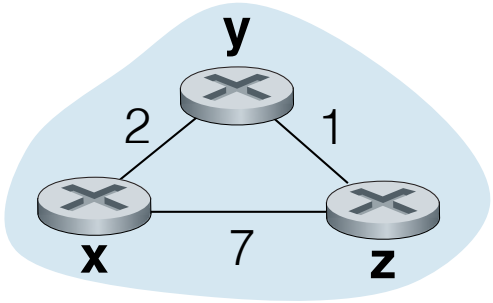
de \ coût vers	x	y	z
x	0	2	7
y	2	0	1
z	7	1	0

Routeur z

de \ coût vers	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

de \ coût vers	x	y	z
x	0	2	7
y	2	0	1
z	3	1	0

de \ coût vers	x	y	z
x	0	2	7
y	2	0	1
z	3	1	0



temps

Routing Information Protocol (RIP)

Un exemple de protocole à vecteur de distance

- **Protocole à vecteur de distance**
  - Les routeurs envoient leur vecteur de distance
    - toutes les 30 secondes
    - suite à une mise à jour provoquant un changement de routage
  - L'absence de vecteurs de distance pendant 180 secondes indique une rupture de lien ou la panne du voisin
- **Coût des liens avec RIP**
  - Tous les liens ont un coût de 1
  - Les distances valides varient de 1 à 15, 16 représente l'infini
    - limiter l'infini permet de réduire le problème du comptage à l'infini
- **RIP est limité aux réseaux de taille restreinte**
  - Réseaux bancaires ou PME

Comparaison des protocoles de routage

Etat de liens

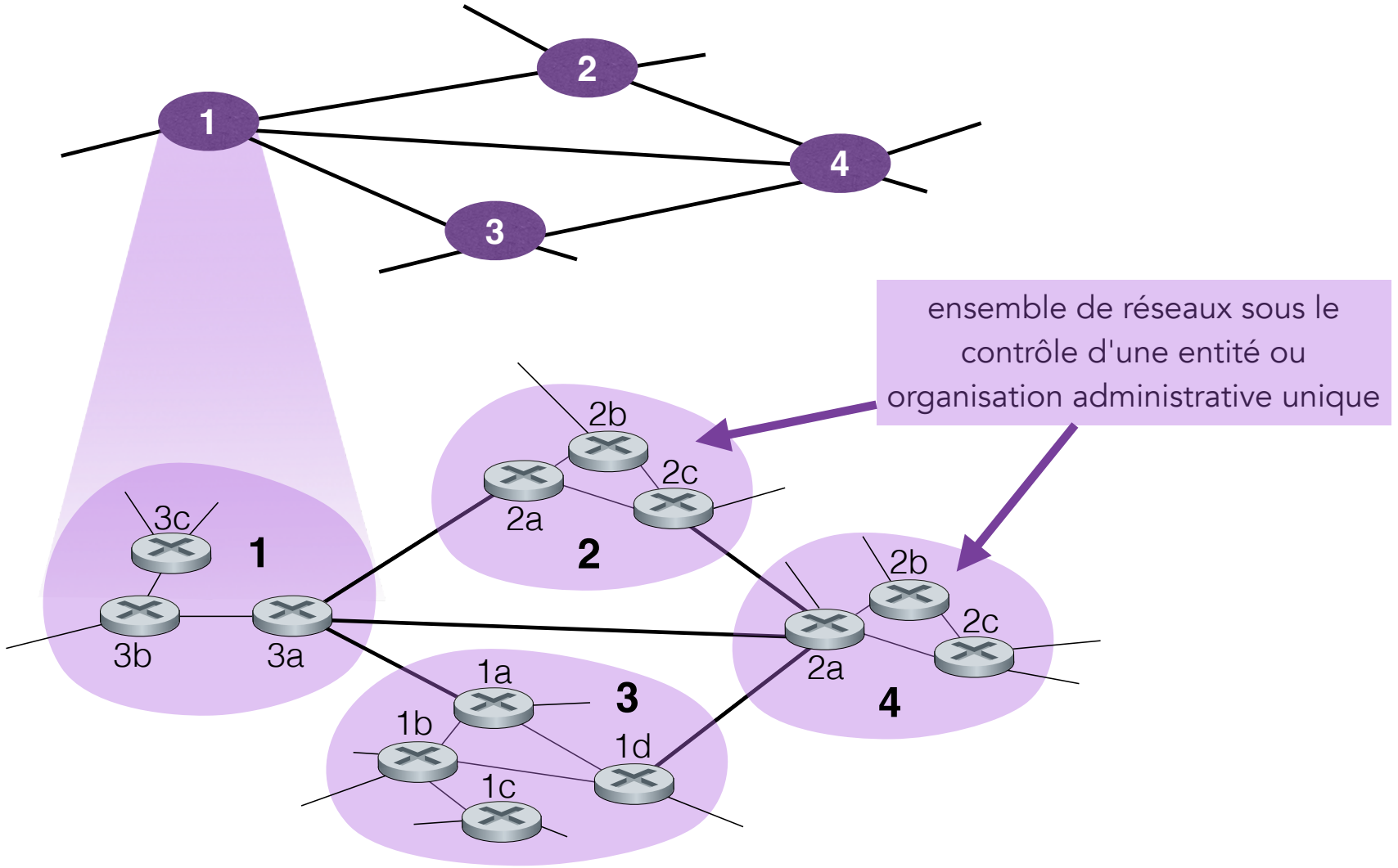
- Les information de topologie sont inondées dans tout le réseau
- Les routeurs calculent les chemins complets les plus courts vers toutes les destinations du réseau
- Les sauts suivants sont déterminés en calculant les chemins complets

- Différentes politiques de routage selon le choix des coûts des liens
- Exemples : OSPF, IS-IS

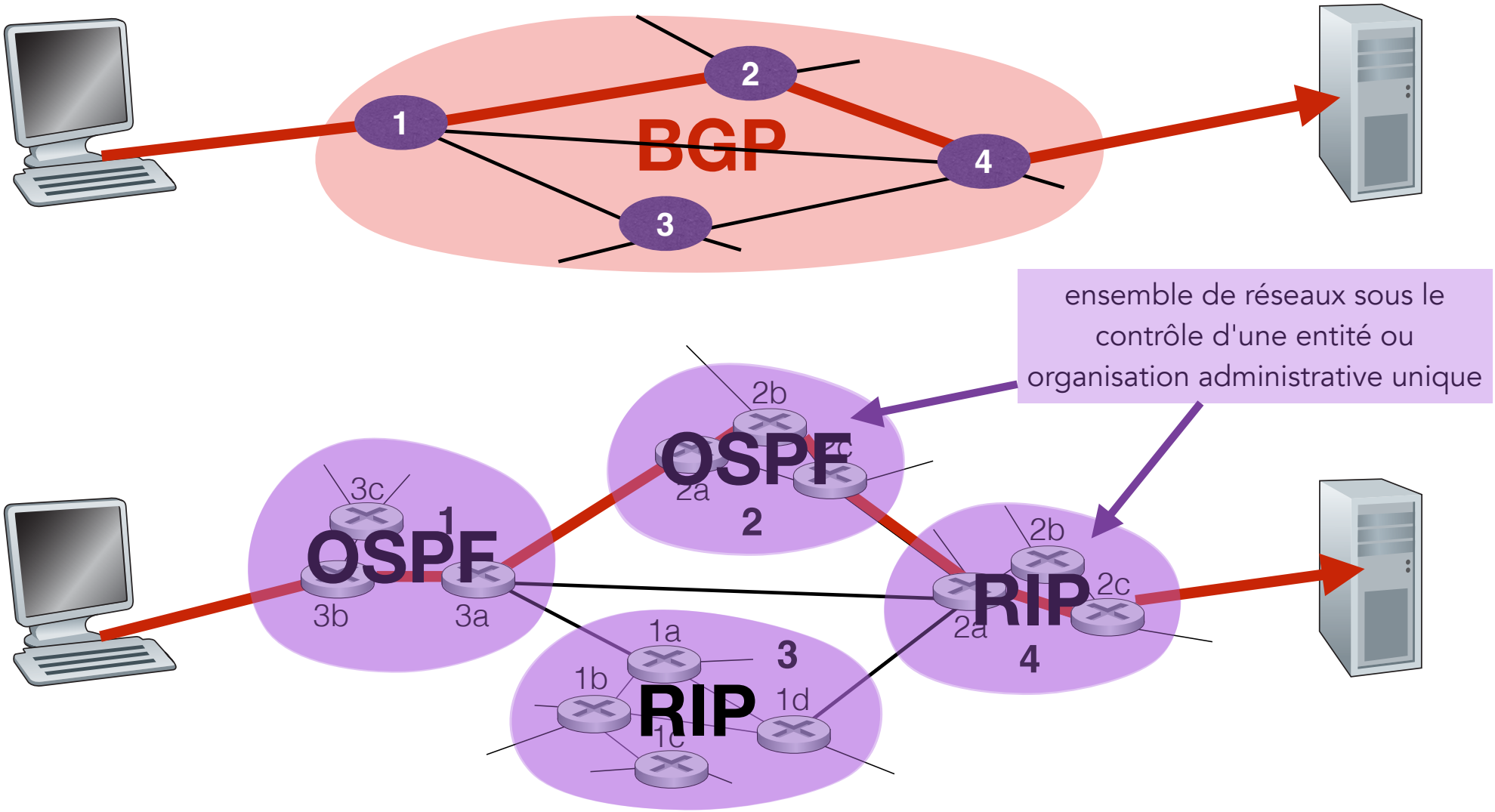
Vecteur de distance

- Un routeur ne connait que la distance des chemins annoncés par ses voisins
- Les routeurs sélectionnent le voisin qui annonce le chemin le plus court pour une destination donnée
- Aucun routeur ne connait les chemins complets
- Le chemin résulte de la séquence des sauts suivants sélectionnés par chaque routeur
- Une seule politique de routage : nombre de sauts des chemins
- Exemples : RIP, BGP

Internet : un réseau de réseaux

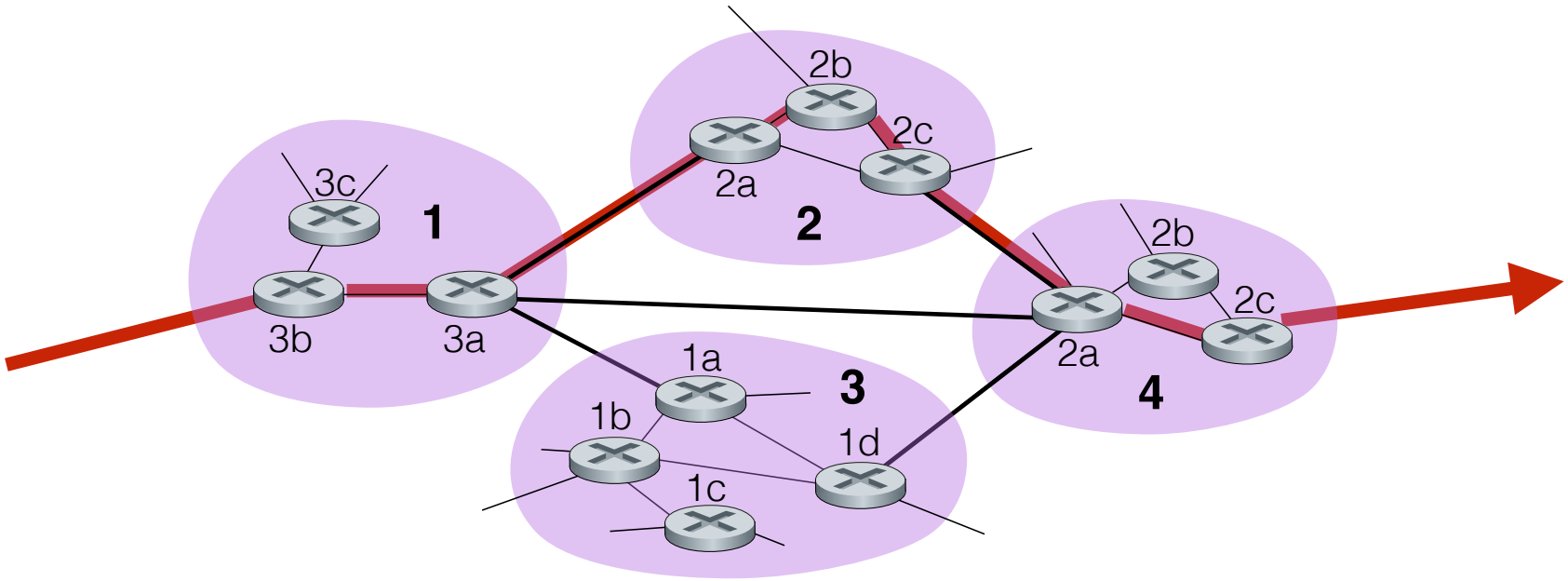


Routage à deux niveaux

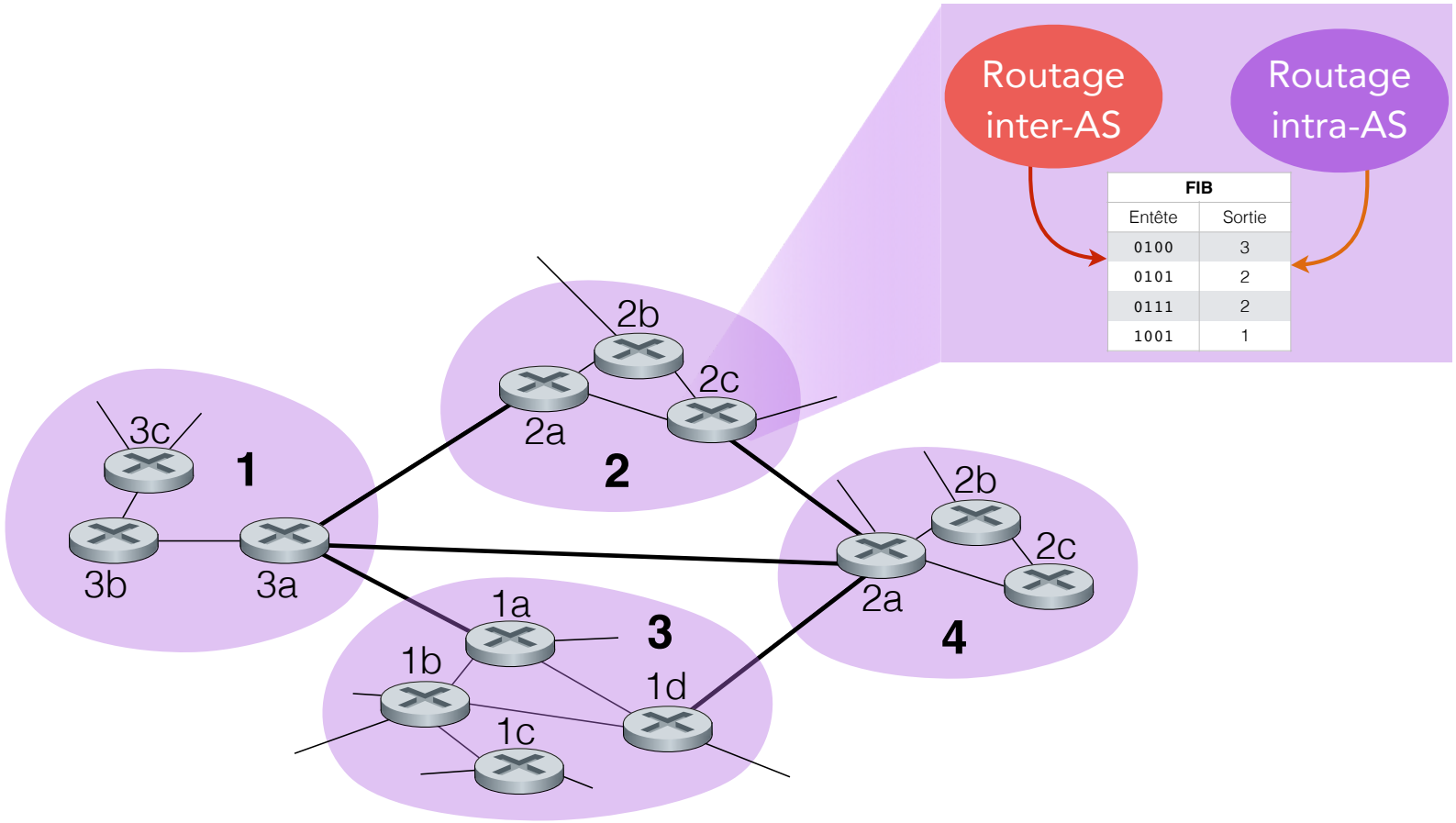


Routage à deux niveaux

- **Protocole de routage inter-domaine : BGP**
  - BGP détermine la séquence des domaines à traverser pour atteindre une destination située hors du domaine de routage de la source
- **Protocoles de routage intra-domaine : RIP, OSPF, IS-IS, …**
  - déterminent les meilleures routes séparant sources et destinations appartenant au même domaine de routage



Routage à deux niveaux



Conclusion

- Le routage permet le calcul des tables de routage
  - Les routeurs découvrent la topologie du réseau
    - les protocoles de routage indiquent aux routeurs comment s'échanger des informations de routage
  - Les routeurs calculent les meilleurs chemins
    - les routeurs exécutent un algorithme de routage
  - Les routeurs construisent et maintiennent une table de routage
- Deux algorithmes de calcul de plus court chemin
  - Routage à état de lien (OSPF, IS-IS) : Dijkstra
  - Routage à vecteur de distance (RIP) : Bellman-Ford
  - Routage à vecteur de chemin (BGP)
- Processus de convergence
  - Changements de topologie
  - Périodes transitoires avant que les tables de routage soient à nouveau cohérentes