Cours Architectures des Réseaux

Communication dans un Réseau

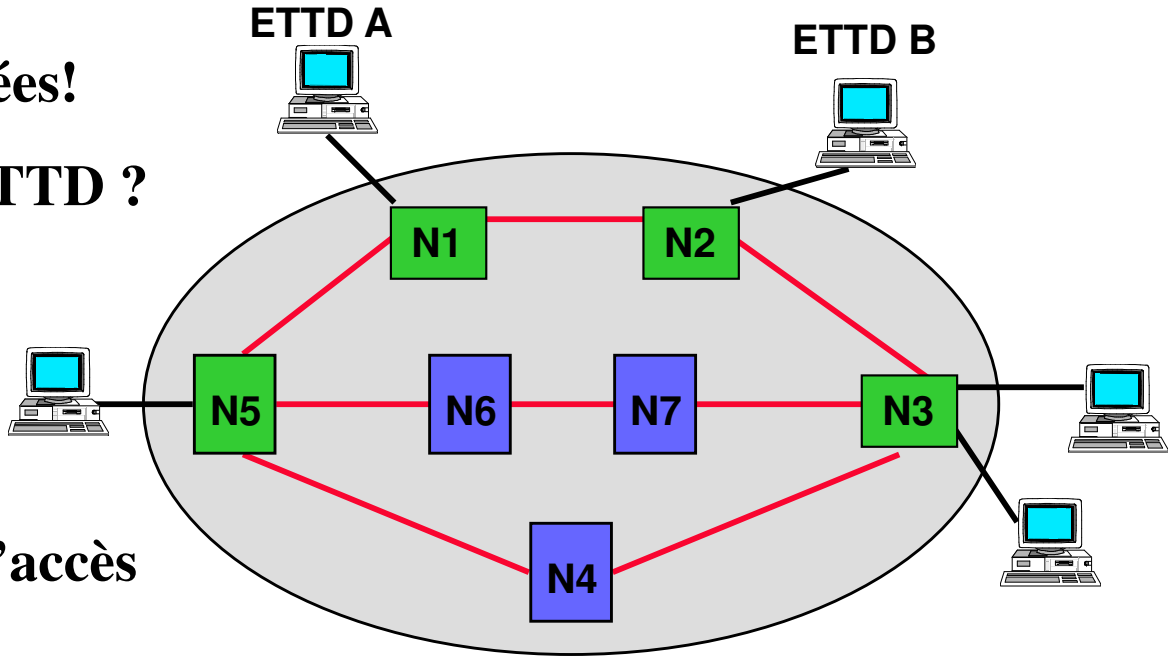
Lila Boukhatem
LISN : Laboratoire interdisciplinaire des sciences du numérique
Université Paris Saclay
Lila.Boukhatem@universite-paris-saclay.fr

PLAN

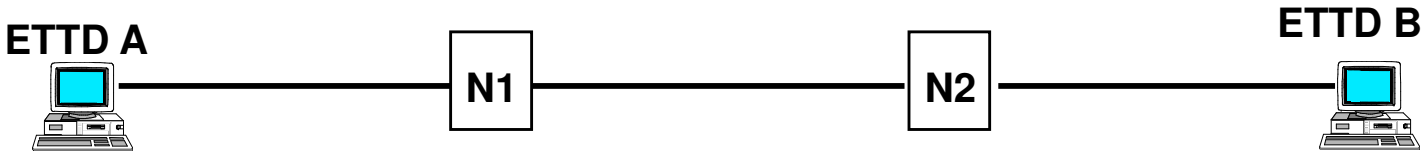
- **Rôle d'un réseau**
- **La commutation**
  - de circuit, de messages, de paquets, de cellules
- **Fonctions d'un réseau**
  - Adressage
  - Routage
  - Contrôle de congestion
- **Conclusion**

Introduction : problématique?

- **On sait faire avec une liaison de données!**
- **mais que se passe-t-il lorsqu'on a *N* ETTD ?**
- **Des nœuds et des liaisons**
  - topologie
  - nœuds d'accès et nœuds internes
- **Chaque station est reliée à un nœud d'accès**



- **Une couche de plus !**



liaison de données A-N1 | liaison de données N1-N2 | liaison de données N2-B

Rôle du réseau

- **Le réseau est un *transporteur* :**
  - Capable de véhiculer des données entre des paires de stations
  - Ne s'intéresse pas à la sémantique des données
- **Les données sont acheminées vers leur destinataire en étant *commutées* de nœud en nœud**
- **3 fonctions essentielles**
  - adressage
  - routage
  - contrôle de congestion

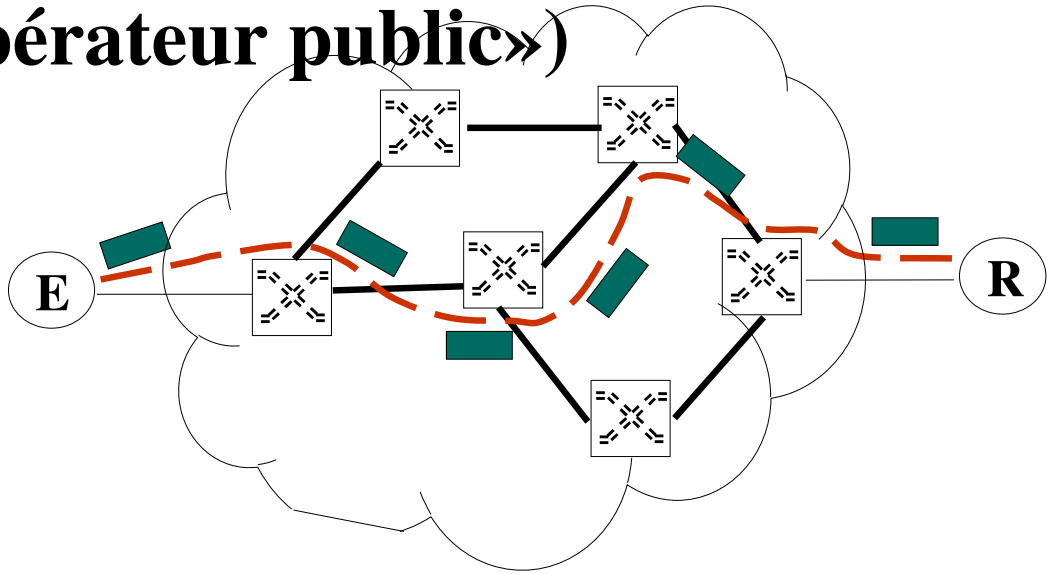
Services de réseau

- **Il doit assurer :**
  - l'*indépendance* par rapport aux supports de transmission sous-jacents
  - le transfert de bout-en-bout (*entry-to-exit*) des données
  - la *transparence* des informations transférées
  - le choix d'une *QoS* (Quality of Service)
  - le service d'*adressage* aux points d'accès du service de réseau
- **Le transfert doit-il être fiable ?**

2 réponses possibles…

Service en mode connecté

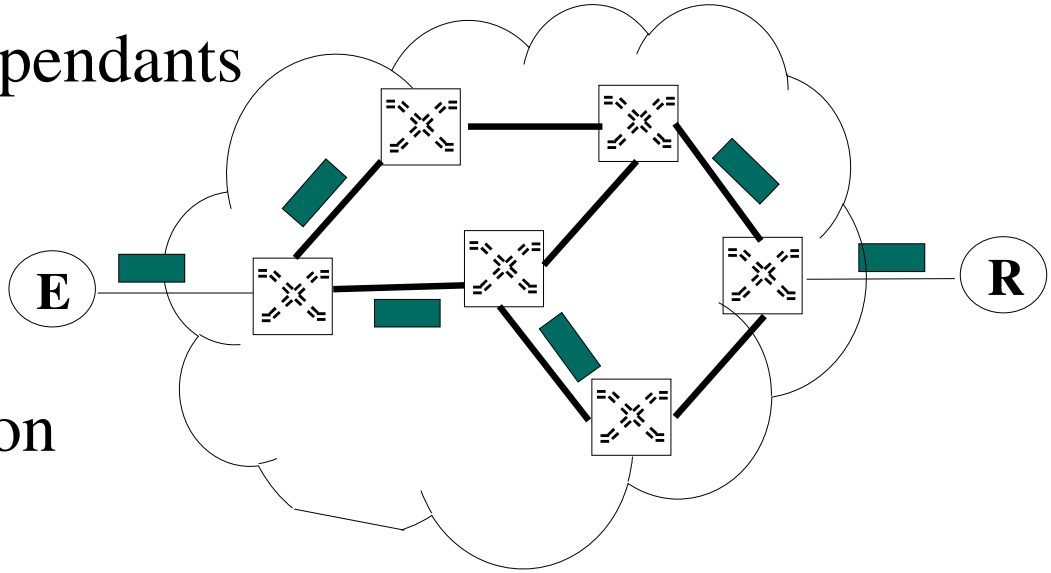
- **service en mode circuit virtuel**
- **transfert fiable (approche «opérateur public»)**
- **communication en 3 phases**
  - établissement de la connexion
  - transfert de données
  - libération de la connexion



- **modélisation du service**
  - N-CONNECT : request, indication, response, confirmation
  - N-DATA : request, indication (confirmation éventuellement)
  - N-DiSCONNECT : request, indication

Service en mode non connecté

- **service en mode datagramme**
- **fiabilité non assurée *best effort* (approche Internet)**
- **communication**
  - par échange de datagrammes indépendants
  - sans notion de connexion
- **modélisation du service**
  - N-UNITDATA : request, indication

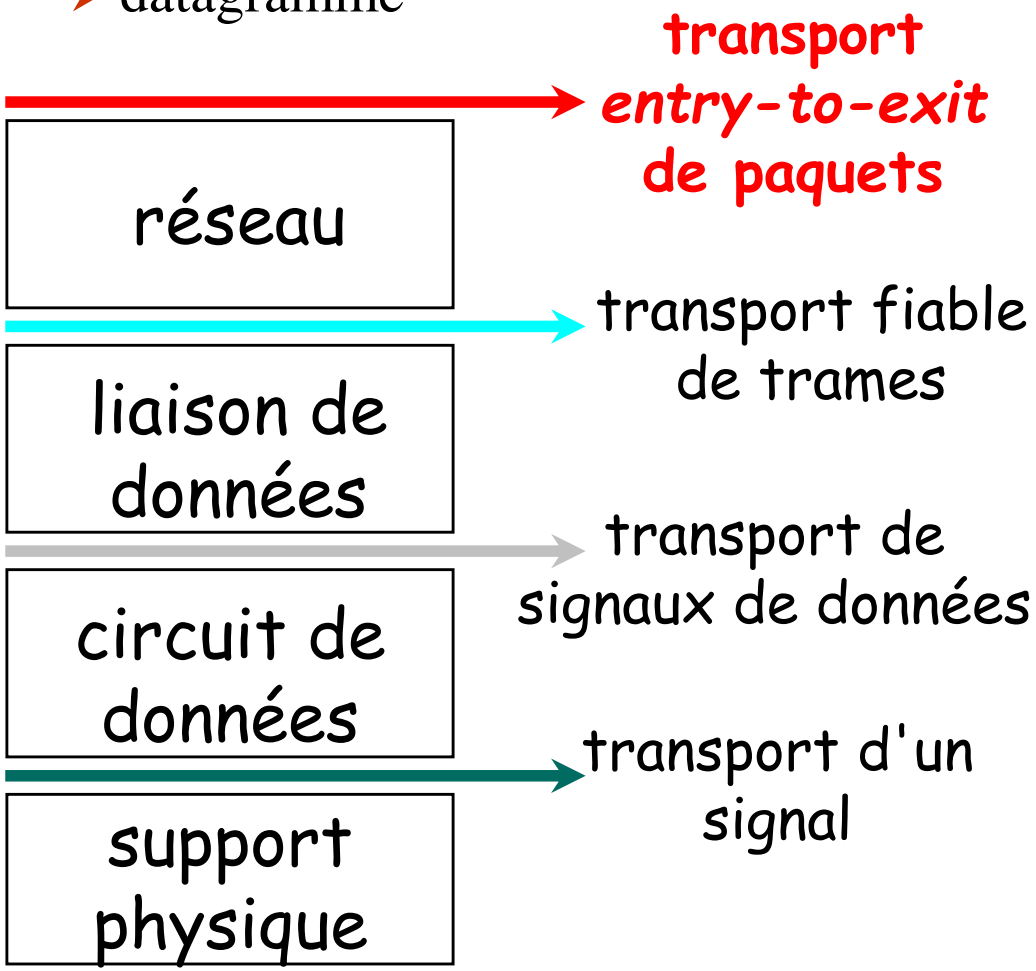


Comparaison des 2 modes de service

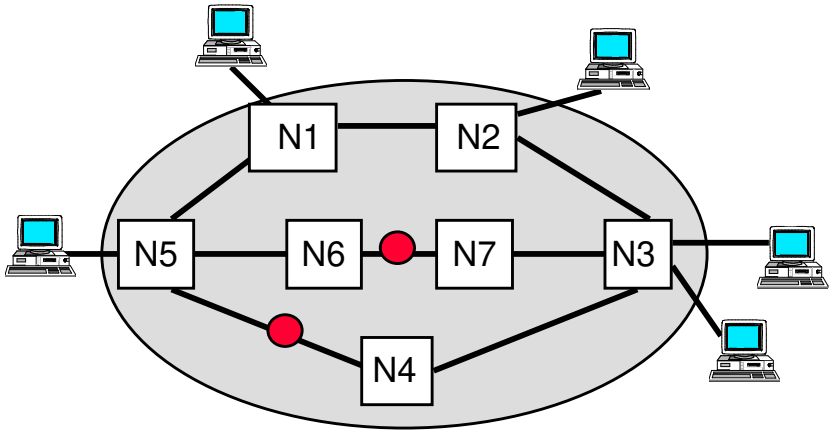
Issue	Datagram subnet	VC subnet
Circuit setup	Not needed	Required
Addressing	Each packet contains the full source and destination address	Each packet contains a short VC number
State information	Subnet does not hold state information	Each VC requires subnet table space
Routing	Each packet is routed independently	Route chosen when VC is set up; all packets follow this route
Effect of router failures	None, except for packets lost during the crash	All VCs that passed through the failed router are terminated
Congestion control	Difficult	Easy if enough buffers can be allocated in advance for each VC

Protocole de réseau

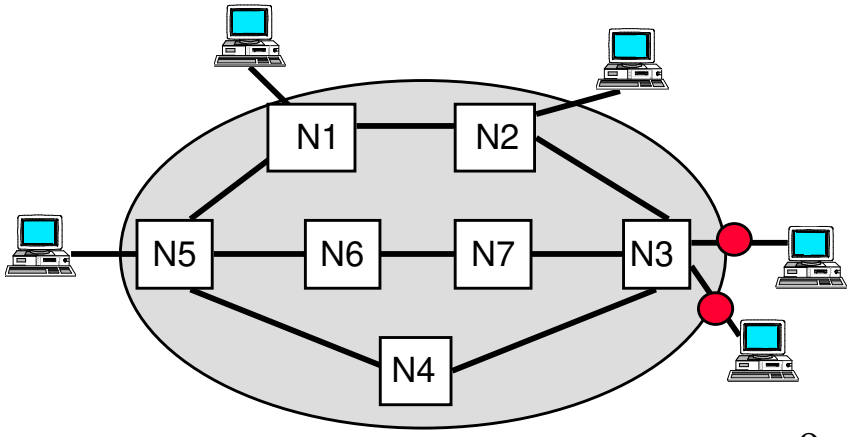
- **2 types de service**
  - circuit virtuel
  - datagramme



- **2 points de vue**
  - entre deux nœuds du réseau



  - entre l'usager et le réseau



PLAN

- **Rôle d'un réseau**
- **Commutation**
  - définition
  - les types de commutation
- **Fonctions d'un réseau**
  - Adressage
  - Routage
  - Contrôle de congestion
- **Conclusion**

Les données binaires

- 1 bit : Binary digit
- 1 octet = 8 bits (en anglais Byte)
- 1 Kbit = 10^3 bits
- 1 Mbit = 10^6 bits
- 1 Gbit = 10^9 bits

Vitesse de transmission

- Débit binaire en bit/s
  - nombre de bits transmis en 1 seconde

Temps de transmission t_t

- Temps nécessaire pour que le message soit envoyé (totalement) sur la ligne. Il dépend du débit du canal (lien).

$$t_t = \frac{l}{D}$$

(Nombre de bits à émettre)

(Capacité (débit) du lien)

- Exemple :

Temps de transmission d'un message de 10 000 bits sur un réseau Ethernet à 10 Mbit/s

$t_t = 10000/10.10^6 = 1$ ms

Temps de propagation t_p

- Transfert non instantané
- Temps nécessaire au signal pour parcourir le support d'un point à au autre de la liaison
- Il dépend de
  - La nature du support
  - La distance d (longueur du lien)
  - La fréquence du signal
  - Vitesse de propagation du signal v :
    - dans le vide (vitesse de la lumière) = 300 000 km/s
    - Câble coaxial : 250 000 km/s

$$t_p = \frac{d}{v}$$

- Exemple
  - Temps de propagation : satellite (250 ms aller-retour), câble coaxial (5 µs/km)

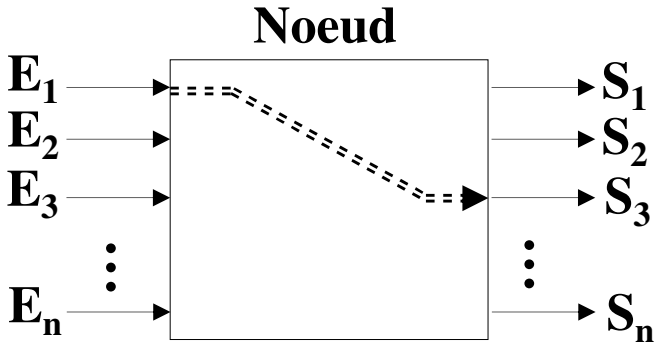
Temps de transfert t_r

- Temps nécessaire pour que le message émis à travers le réseau soit reçu complètement par l'équipement terminal récepteur.
- C'est le temps entre l'émission du 1er bit d'un message et la réception de son dernier bit

$$t_r = t_t + t_p$$

La commutation

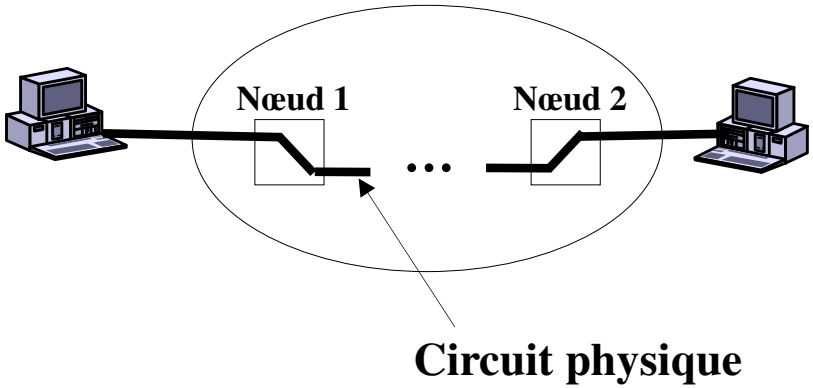
- **Commuter**
  - aiguiller une communication d'un port d'entré vers un port de sortie
- **Plusieurs techniques**
  - Commutation de circuits
  - Commutation de messages
  - Commutation de paquets
  - Commutation de cellules



Commutation de circuits

Principe

- Circuit : lien physique permanent établi pour la durée de la connexion
- Il n'appartient qu'aux deux unités communicantes
- La circuit est établi préalablement au transfert
- Les ressources sont allouées pour toute la durée de la connexion



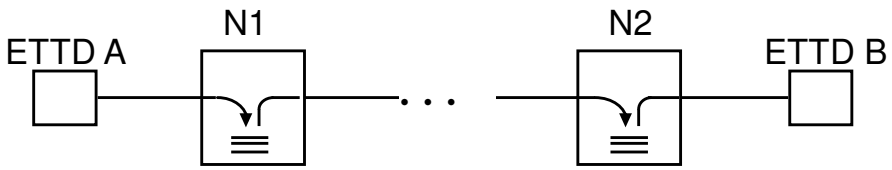
Commutation de circuits

- Avantages :
  - Délai de transfert constant
  - Idéale pour les applications à débits constants
  - Séquencement garanti
  - Pas de congestion
- Inconvénients :
  - Trafics sporadiques : mauvaise utilisation des ressources
  - Risque de rejet à l'établissement
  - Échanges brefs : délais d'établissement

Commutation de messages

Principe

- un *message* est une information formant logiquement un tout et pour la source et pour le destinataire
- chaque message est envoyé indépendamment des autres
- fonctionnement de type *Store-and-Forward*



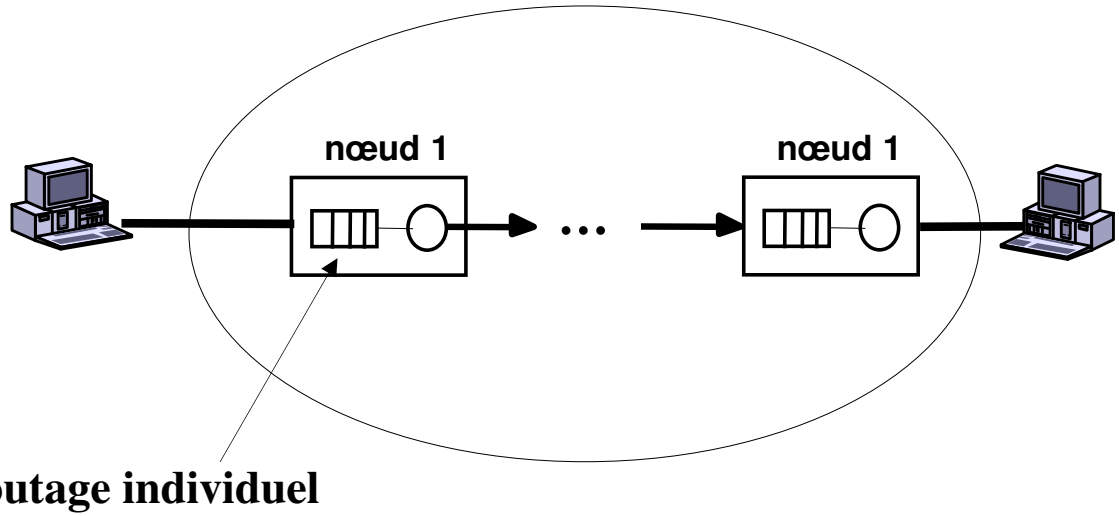
Commutation de messages

- **Avantages**
  - les ressources ne sont utilisées que lorsque nécessaire
- **Inconvénients**
  - ressources de stockage importantes
  - temps de transfert importants et variables
  - risques de congestion
  - taux d'erreurs message importants

Commutation de paquets

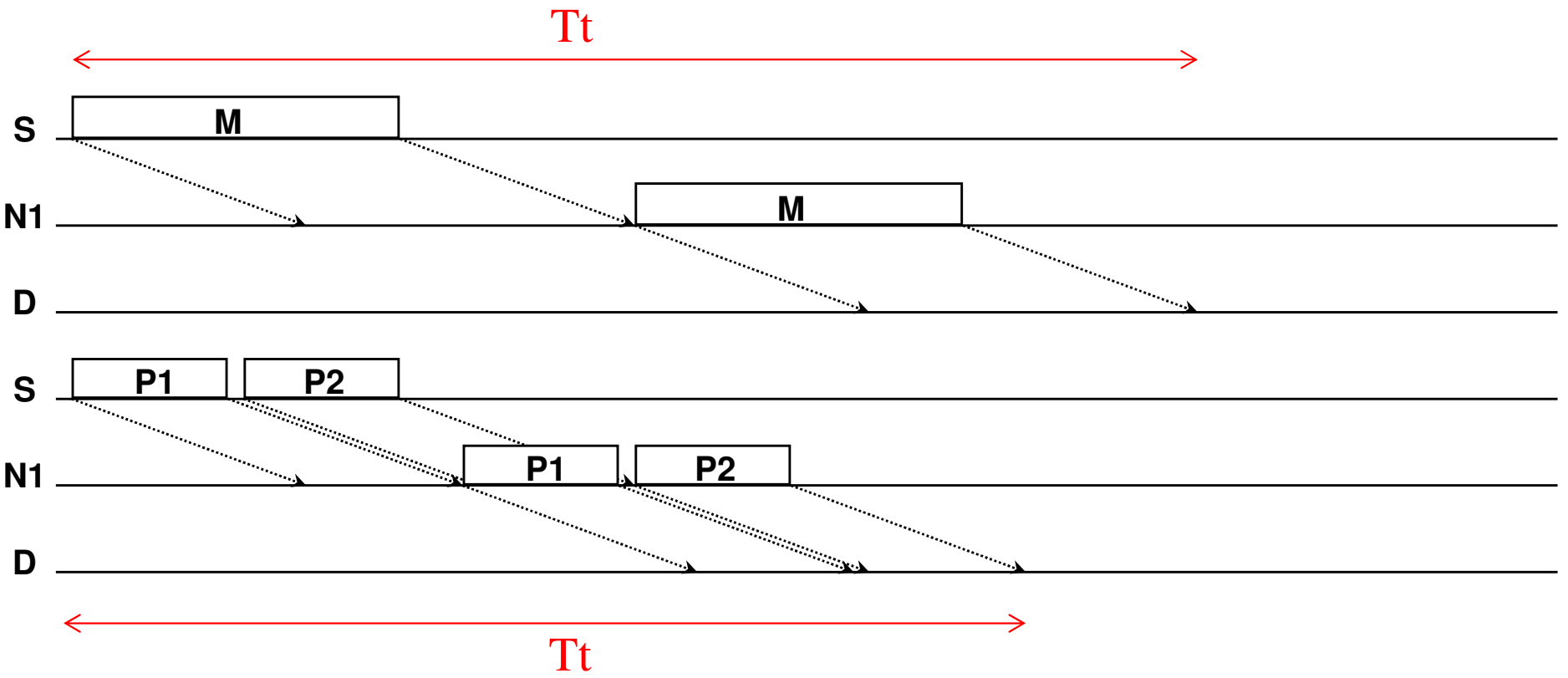
Principe

- Pareil que commutation de messages sauf : découpage des messages en paquets de taille limitée
- En-tête utilisé pour le routage, la correction d'erreurs, le contrôle de flux, etc.
- Buffers dans les commutateurs
- Ressources partagées : optimisation



Commutation de paquets

- **Commutation de messages vs. paquets**



Commutation de paquets

- Avantages :
  - Diminution du temps de transfert
  - Ressources utilisées quand nécessaire
  - Efficace pour les applications à échanges sporadiques
- Inconvénients :
  - Délai d'acheminement (temps de transfert) variable
  - Déséquencement
  - Risque de congestion

Commutation de cellules

- **Principe**
  - idem commutation de paquets en mode connecté
  - cellule = paquet de taille fixe et courte (53 octets)

Commutation de cellules

petite taille

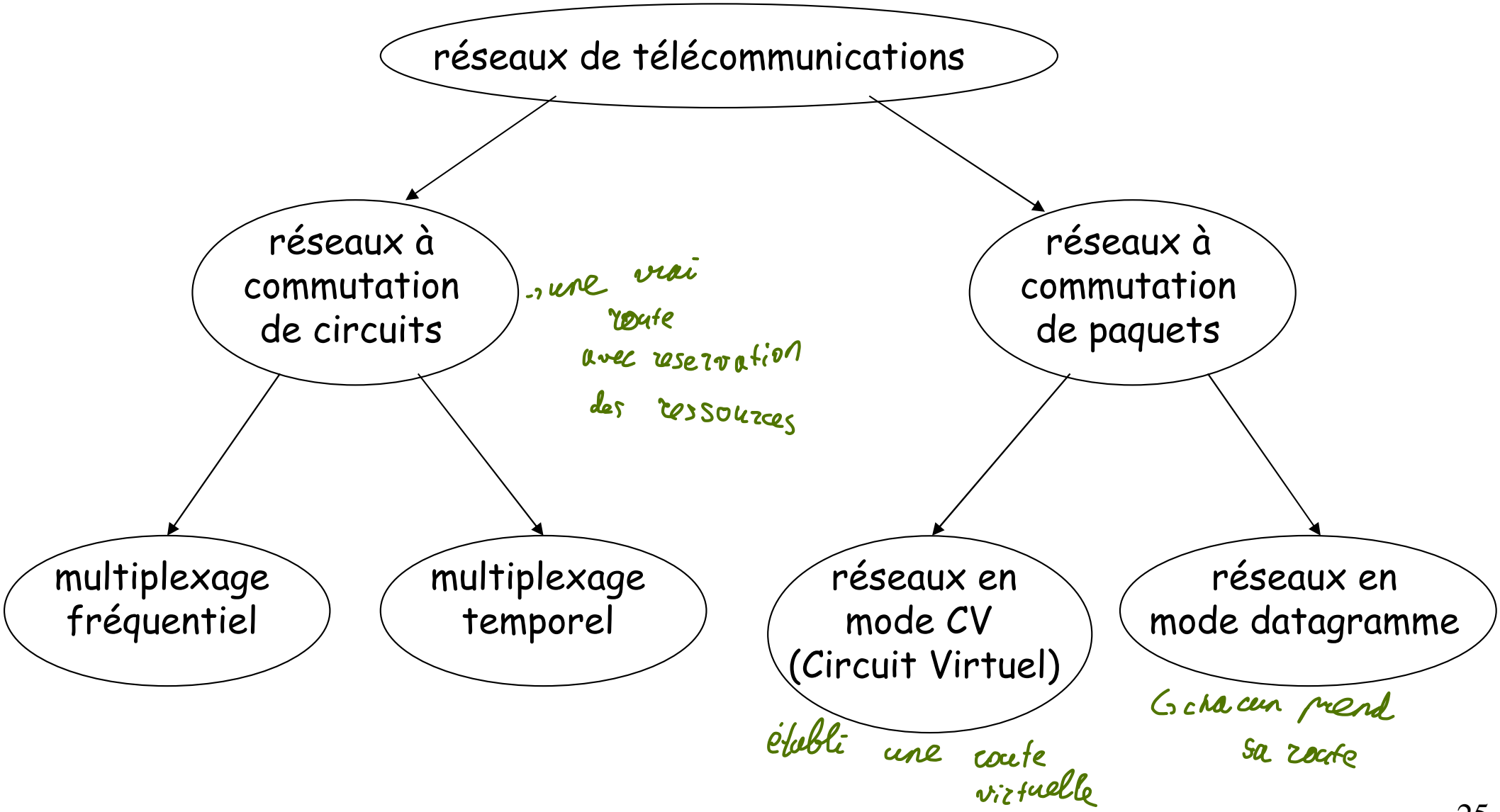
- ☺ **réduction du temps de constitution des paquets**
- ☺ **réduction du délai d'acheminement**
- ☺ **réduction du nombre de pertes (dues à des dépassements de files d'attente)**
- ☺ **réduction de la taille des tampons des nœuds**
- ☺ **meilleur entrelacement des messages (voir schéma) : puisque les grands flux de données sont découpés en petites cellules, le trafic isochrone peut s'intercaler sans subir de retard significatif**
- ☺ **gigue faible**
- ☹ **diminution de l'efficacité de transmission (*overhead* important)**
- ☹ **augmentation du nombre de traitements dans les nœuds de commutation car plus d'en-têtes à traiter**

taille fixe

- ☺ **augmentation de la capacité des nœuds (traitement // dans les nœuds)**
- ☺ **Délimitation implicite de la cellule**
- ☺ **meilleure performance (utilisation de technologies à très haute intégration *hardware*)**
- ☺ **gestion mémoire des commutateurs plus simple**
- ☹ **mauvaise utilisation de la bande passante (le cadrage des cellules nécessite des octets de bourrage)**

L'entrelacement

Pour récapituler…



Plan

- **Rôle d'un réseau**
- **Commutation**
- **Fonctions d'un réseau**
  - Adressage
  - Routage
  - Contrôle de congestion
- **Conclusion**

L’adressage

- **Problématique**
  - lorsqu'un système de transmission est utilisé par plus de 2 équipements, pour délivrer une unité de données à son destinataire, il faut connaître l'adresse de ce dernier
  - de manière unique et non ambiguë
- **Techniques d’adressage**
  - plate / absolue
    - ☺ adresse identique partout
    - ex : Ethernet
  - hiérarchique
    - ☺ facilite le routage
    - ex : téléphone, @ postale, IP
- **Types d'adresses**
  - unicast
  - broadcast
  - multicast

Problématique

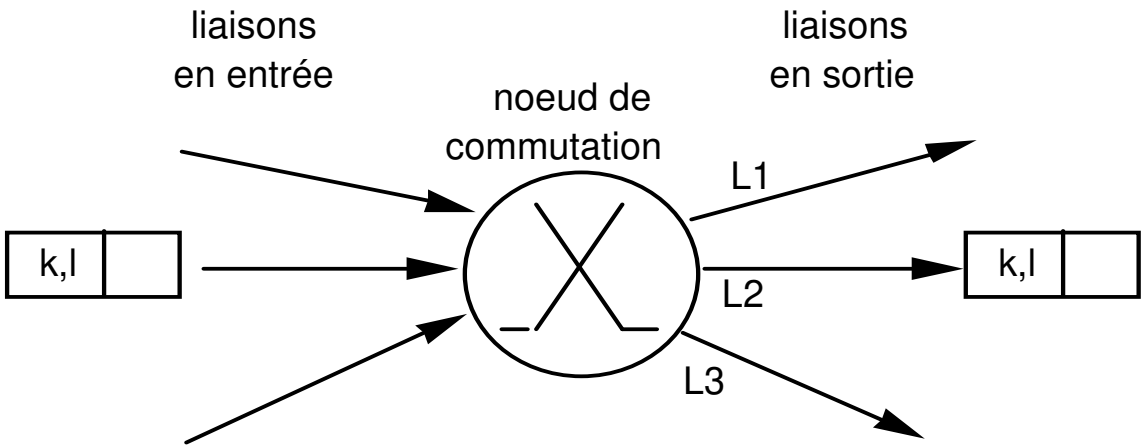
- trouver *la meilleure* route pour aller d'une source vers une destination
- *la meilleure* ?
  - Plusieurs critères de coût
    - distance, nombre de nœuds traversés
    - temps de réponse
    - débit
    - fiabilité
    - coût financier

Le routage

- **Routage : terme général, utilisé pour 2 fonctions**
  - acheminement (*forwarding*)
    - trouver un chemin au paquet vers sa destination par consultation de tables
    - fonction simple, se déroulant localement à un noeud
  - adaptation des chemins (*routing*)
    - construire et mettre à jour les tables servant à l'acheminement
    - fonction complexe, mettant en œuvre des algorithmes distribués qui s'appuient sur des protocoles de routage

La fonction d'acheminement

- **Principe**
  - chaque nœud dispose d'une *table de routage* qui donne pour chaque destination le port (la ligne) de sortie sur lequel le noeud doit commuter le paquet
  - par sauts successifs, de nœud en nœud, le paquet parvient au destinataire
- **Exemple**



destination	liaison de sortie
a	L1
l	L2
N	L3

  - un paquet d'origine *k* et à destination de *l* traverse un nœud
  - le nœud consulte sa table de routage qui donne L2 comme liaison de sortie

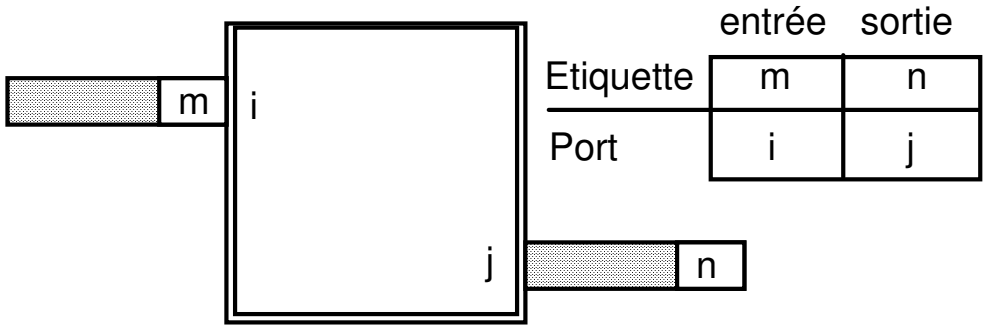
La fonction d'acheminement

- **Notion d'étiquette**
  - information de contrôle contenue dans l'en-tête du paquet, utilisée pour acheminer le paquet
- **2 modes d'acheminement**
  - acheminement par voie logique (circuit virtuel)
  - acheminement par datagramme

Acheminement par voie logique

Principe

- étiquette = N° VL
  - signification locale au multiplex (i.e. à un tronçon de connexion)
  - permet de distinguer plusieurs communications empruntant ce multiplex
- translation d'étiquette par le nœud
  - les informations nécessaires à cette translation sont stockées dans une table du nœud appelée *table de translation* (*lookup*)

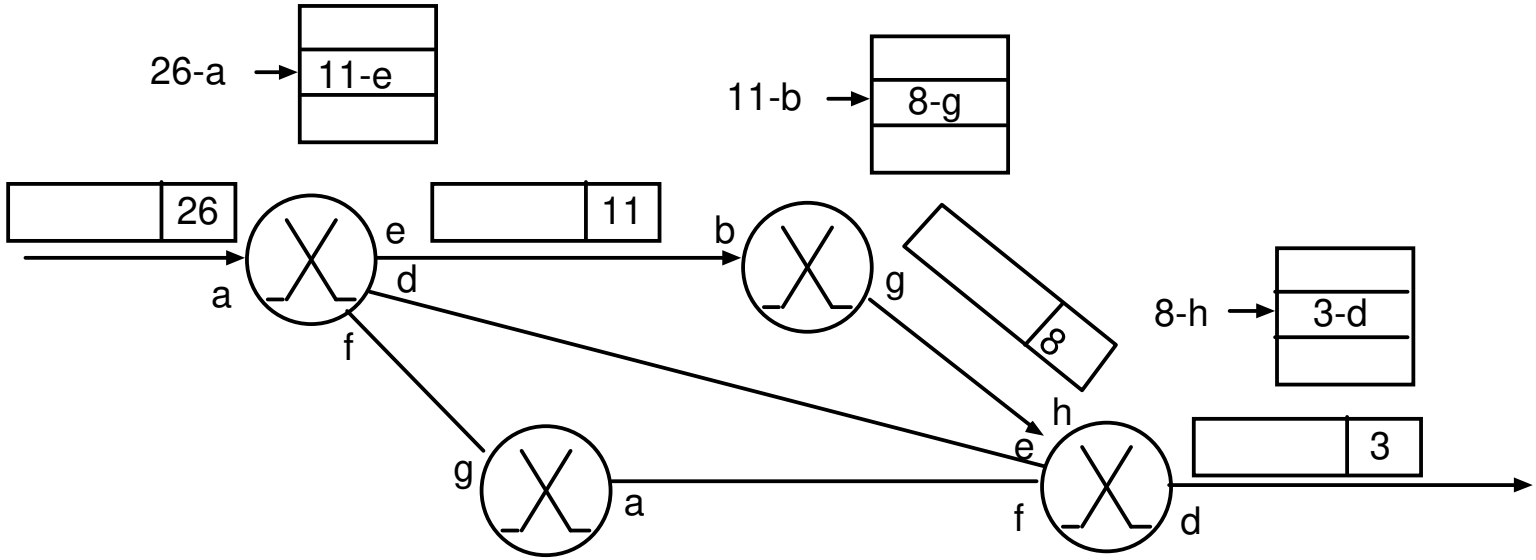


Acheminement par voie logique

- Préalablement à tout transfert de données, un itinéraire doit être marqué dans les nœuds via une correspondance temporaire entre une voie logique entrante sur un multiplex entrant et une voie logique sortante sur un multiplex sortant
  - mode connecté
  - tables de translation mises à jour lors de l'établissement et de la libération des connexions
  - les paquets d'une même connexion suivent le même chemin physique
    ➔ *circuit virtuel*

Acheminement par voie logique

- **exemple**
  - la succession des numéros logiques peut être vue comme un circuit : c'est le *circuit virtuel*



Acheminement par datagramme

- **Principe**
  - étiquette = adresse du destinataire
  - les paquets sont acheminés individuellement
- **Avantages et inconvénients**
  - ☺ l'expéditeur peut émettre sans accord préalable explicite du réseau
  - ☹ fiabilité non garantie
    - ☹ déséquencement
    - ☹ pertes de paquets
  - ☹ overhead (étiquette + longue)

Prise de décision du routage

- **Le moment de prise de décision du routage (consultation de la table de routage) dépend du mode d'acheminement**
  - acheminement par voie logique
    - décision de routage prise une seule fois lors du passage du paquet d'établissement
    - tous les autres paquets de la connexion suivent la même route, après consultation de la table de translation
  - par datagramme
    - une décision de routage est prise pour chaque paquet
    - la route peut être identique ou non pour deux paquets ayant le même destinataire

La fonction d'adaptation des chemins

Principe

- réalisée grâce aux **protocoles de routage** qui maintiennent des **tables de routage** dans les nœuds du réseau
- une table de routage comporte au moins deux colonnes
  - la première pour la destination (ou pour le réseau de destination)
  - la seconde pour l'adresse du nœud correspondant au « saut » suivant sur le « meilleur » chemin vers la destination souhaitée.
- lorsqu'un datagramme arrive sur un routeur (lorsqu'un paquet d'appel arrive sur un commutateur), le routeur (le commutateur) consulte sa table de routage pour décider du prochain nœud pour ce paquet

Protocoles de routage : objectifs

- **tout protocole de routage doit communiquer des informations sur la topologie *globale* du réseau à chaque routeur afin que celui-ci puisse prendre une décision *locale* de routage**
- **information globale**
  - difficile à collecter
  - sujette à des modifications fréquentes
  - Volumineuse

- **Simplicité : minimisation des messages de contrôle échangés**
- **Minimisation de l'espace des tables de routage**
- **Robustesse : éviter les trous, les boucles et les oscillations**
- **Optimisation : utilisation du meilleur chemin**
- **Rapidité de convergence**
- **Flexibilité : adaptation dynamiquement aux changements de topologie**

Protocoles de routage : classification

- **centralisé vs. distribué**
  - en routage centralisé, un nœud central se charge de collecter les informations sur chaque lien (on/off, utilisation, capacité), et de calculer la table de routage pour chaque nœud du réseau
  - en routage décentralisé, les routeurs coopèrent selon un protocole de routage distribué de façon à construire des tables de routage consistantes
- **à la source vs. saut par saut**
  - en routage à la source, un paquet peut transporter toute sa route
  - en routage saut par saut, un paquet ne véhicule que l'adresse de la destination
- **déterministe vs. stochastique**
  - en routage déterministe, tous les paquets vers une même destination seront retransmis au même nœud suivant
  - en routage stochastique, chaque routeur maintient plusieurs nœuds aval pour une même destination
- **à chemin unique vs. à chemin multiple**
  - en routage à chemin multiple, chaque routeur maintient une route principale et des routes alternatives qu'il peut utiliser en cas d'indisponibilités de la route principale
- **statique vs. dynamique**
  - en routage statique, le choix de la route ne dépend pas de l'actuelle mesure de l'état du réseau
  - en routage dynamique, le choix de la route dépend de l'actuelle mesure de l'état du réseau

Algorithmes de routage

- **Le routage est, par essence, un problème de la théorie des graphes**
  - trouver le chemin de coût minimum entre deux nœuds quelconques, sachant que le coût d'un chemin est la somme des coûts des liens qui le composent
- **Les algorithmes utilisés :**
  - supposent que chaque routeur connaît l'adresse de chacun de ses voisins, ainsi que le coût pour l'atteindre ;
  - permettent à chaque routeur de déterminer l'information de routage globale, i.e. le prochain nœud pour atteindre chaque destination possible sur la route la plus courte, en échangeant de l'information de routage seulement avec ses voisins
- **Deux classes d'algorithmes de routage**
  - les algorithmes à vecteurs de distance (Distance Vector) : Bellman-Ford
  - les algorithmes à états de liens (Link State) : Dijkstra

Algorithmes à vecteurs de distance

Principe

- **Un routeur met à jour sa table de routage** pas à pas, par **échange périodique** d'information de routage (vecteurs de distance) avec **ses voisins directs**
- Un vecteur de distance donne pour chaque destination le coût pour y aller (exp: distance mesurée en nombre de sauts)
  - Un vecteur de distance = {(X=d, int)}
  - X : destination, d : meilleur coût (métrique), int : interface locale.
  - Pour aller à **X** , meilleur coût est d, en passant par l'interface int
- L'algorithme **cumule les distances** afin de tenir à jour une base de données contenant les informations sur la topologie du réseau
- La **modification d'une entrée** dans la table d'un routeur engendre **l'émission de la nouvelle table** (vecteurs) vers les routeurs voisins
- Les échanges continuent jusqu'à ce que l'algorithme **converge** (régime permanent)

Vecteur de distance

Remarques

- La modification des tables se fait périodiquement
  - Difficulté : choix de la valeur adéquate de la période pour ne pas « trop » surcharger le réseau et pour pouvoir détecter l'existence d'une panne
- Si un routeur donné tombe en panne et cesse d'émettre périodiquement ses vecteurs, les autres routeurs mettent leur métrique pour ce routeur à l'infini

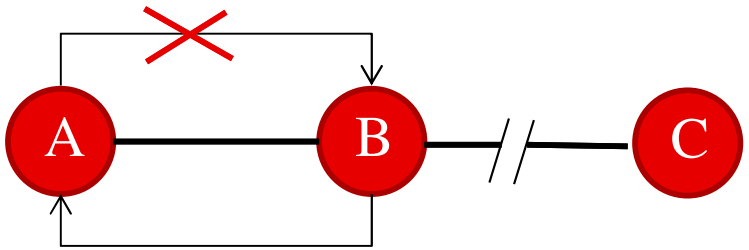
Vecteur de distance

Boucles de routage

- Peut se produire si convergence lente avec une nouvelle info entrainant des entrées de routage incohérentes

Solutions

- Métrique de mesure infinie : « Diminuer » la valeur de l'infini (16)
- L'horizon partagé (Split Horizon)
  - Aucune information de mise à jour ne sera renvoyée sur le chemin par lequel on a appris la modification de topologie



Algorithmes à états de liens

- Link State Routing
  - Chaque routeur doit avoir une vision complète de la topologie du réseau
- Etapes
  - Découvrir ses voisins et apprendre leurs adresses réseau respectives
  - Construire un paquet spécial disant tout ce que le routeur vient d ’apprendre
  - Diffuser ce paquet spécial à tous les autres routeurs
  - Construire une carte du réseau
  - Calculer le plus court chemin vers tous les autres routeurs

Algorithme à états de liens

Découverte des voisins

- A l'initialisation, le routeur détermine quels sont ses voisins
  - Utilisation du paquet : HELLO
  - Les routeurs répondent au paquet en donnant les informations de routage

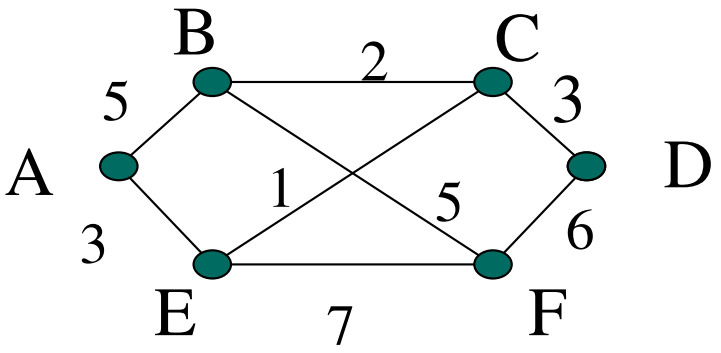
Construction du paquet LSP

- Chaque routeur construit un paquet spécial (LSP) contenant notamment
  - L 'identité du routeur source
  - La liste des routeurs voisins, à chaque voisin est associé le coût correspondant
- Construction des paquets périodique ou évènementielle

Algorithme à états de liens

Construction du paquet

- Exemple



A	
B	5
E	3

B	
A	5
C	2
F	5

C	
B	2
D	3
E	5

D	
C	3
F	6

E	
A	3
C	1
F	7

F	
B	5
D	6
E	7

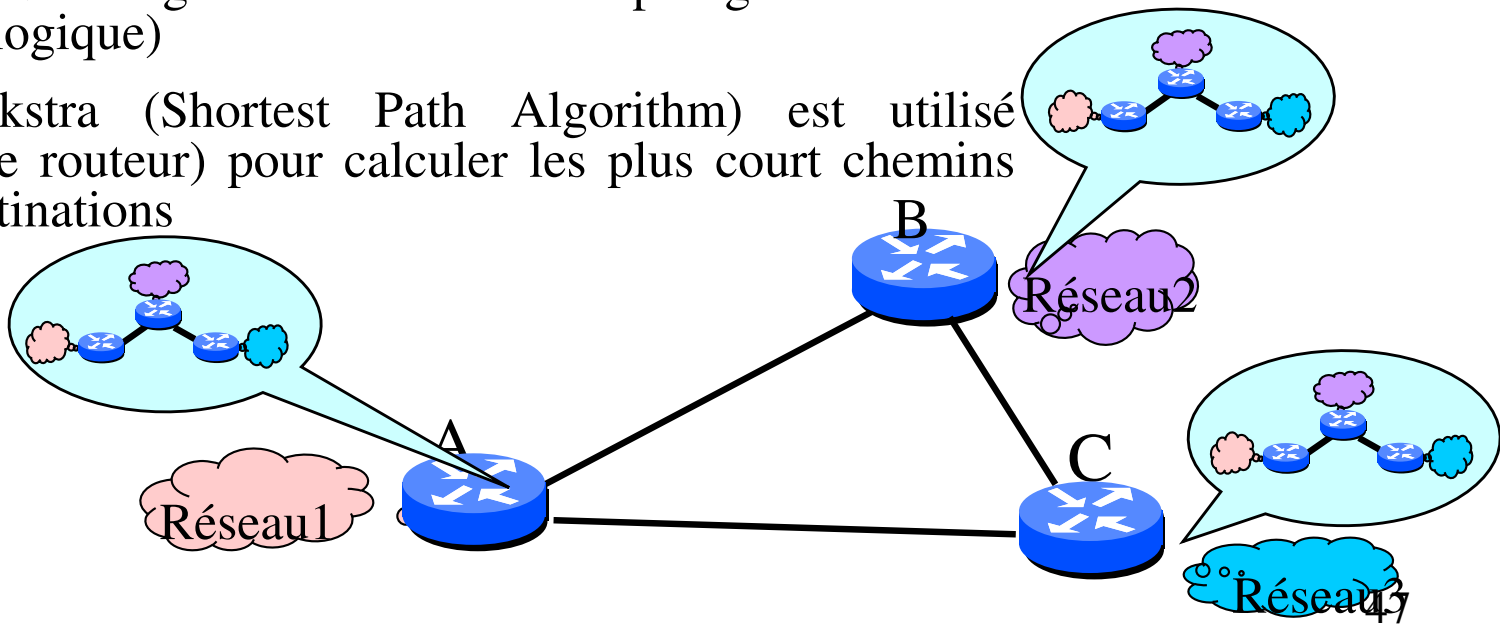
Algorithme à états de liens

Diffusion des paquets

- La distribution des paquets utilise l'inondation (flooding)
  - Chaque paquet entrant est émis sur chaque ligne excepté la ligne d'arrivée (d'entrée)

Calcul des nouvelles routes

- A partir des paquets reçus, chaque routeur construit le graphe complet du réseau
- Chaque routeur a une vision globale de toute la topologie du réseau (Base de données topologique)
- L'algorithme de Dijkstra (Shortest Path Algorithm) est utilisé localement (sur chaque routeur) pour calculer les plus court chemins vers les différentes destinations



Link State versus Distance Vector

Critères	Distance-Vector	Link-State
Complexité (CPU)	Simple	Grande
Mémoire	Peu	beaucoup
Charge (signalisation)	Petite	Grande
Convergence	Lente	Rapide