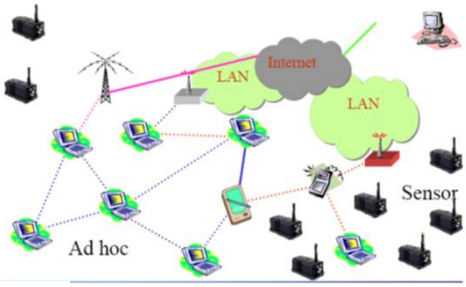
Gestion de la Qualité de service

2004-2018

Cours de DEA & Master 2 Recherche



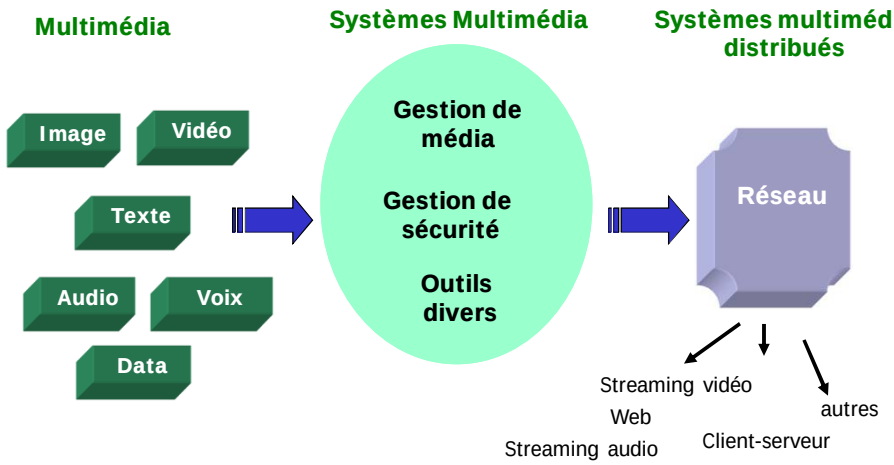
UNIVERSITÉ TOULOUSE III PAUL SABATIER

Zoubir Mammeri

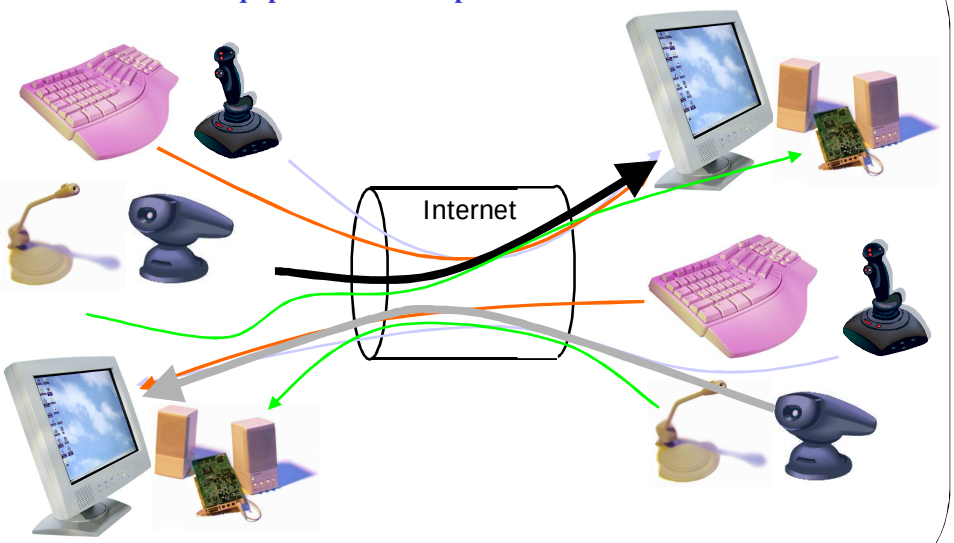
Chapitre 1

Introduction aux applications multimédia

Définitions des concepts liés au *Multimédia* (

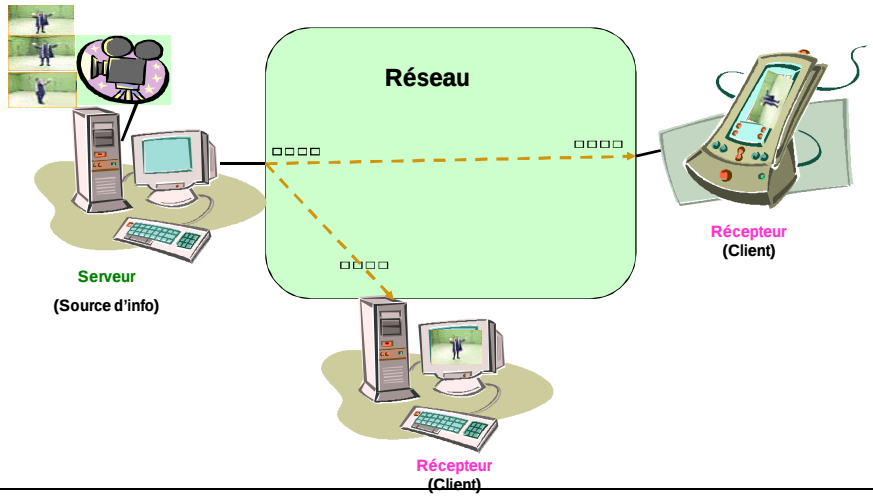


Equipements filaires pour le *Multimédia*

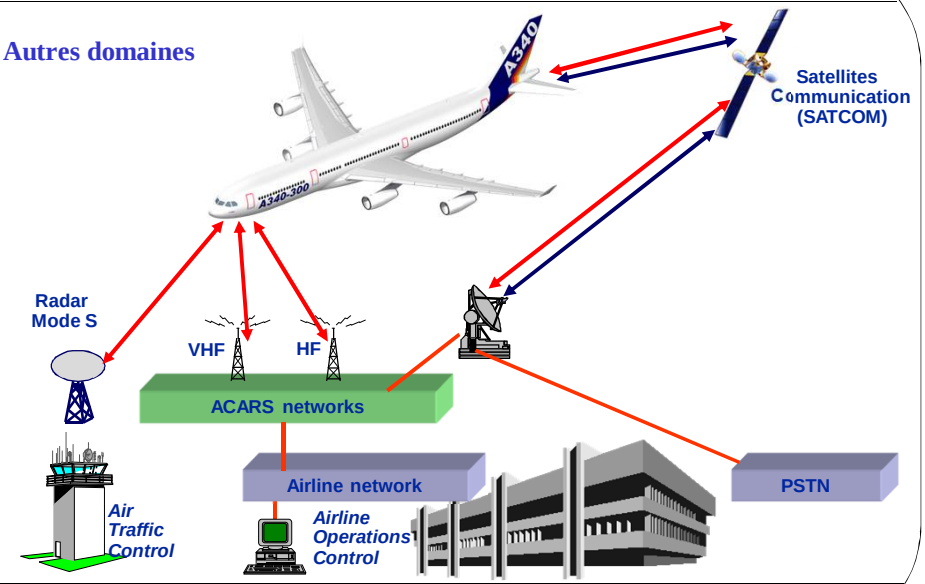


Equipements sans fil et mobiles pour le *Multimédia*

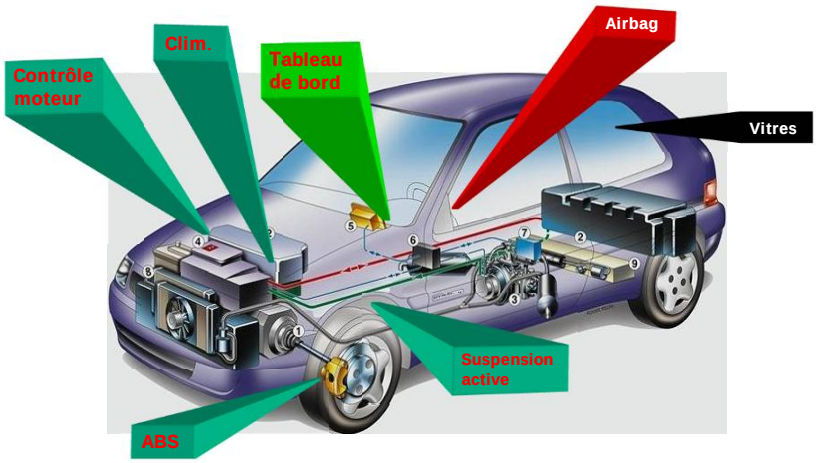
Principe général des applications multimédia distribuées



Autres domaines



Autres domaines



2. Classes d'applications multimédia

Classification des applications multimédia

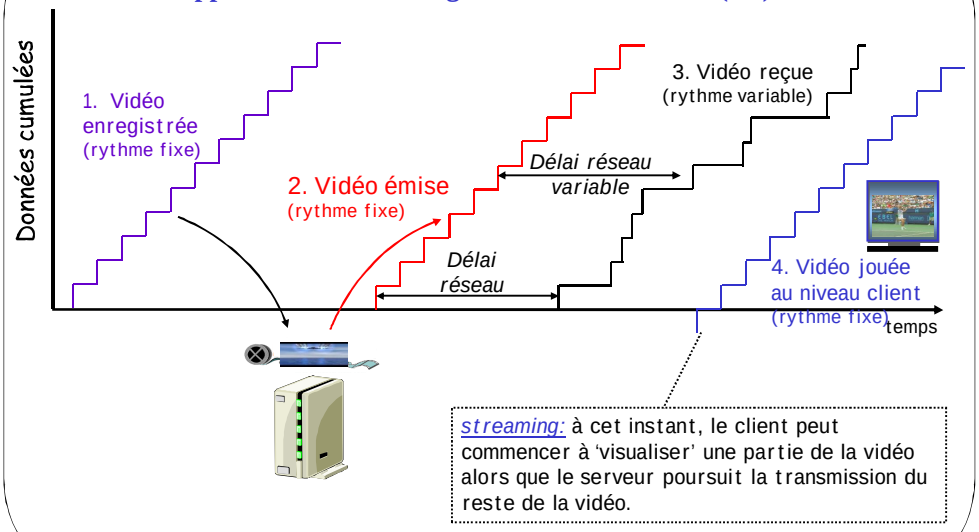
- **Selon l'interactivité**
  - Non interactives : radio et TV, vidéo à la demande, e-learning...
  - Interactives : vidéo surveillance, téléguidage, vidéo conférence, téléphonie, conférence téléphonique, TV interactive, télé-médecine, téléachat, bourse, jeux…
- **Selon la criticité**
  - (Très) critiques : guidage et supervision, télé opération chirurgicale…
  - (Moyennement) critiques : vidéo conférence, bourse, téléachat
  - Non critiques : TV, radio, jeux…
- **Selon les timings (temps réel)**
  - *Streaming* de données audio/vidéo préalablement stockées
  - *Streaming* 1-à-m temps réel de données audio-vidéo
  - Applications interactives d'audio/vidéo

2. Classes d'applications multimédia

Applications « streaming stored Audio/video » (1/2)

- **Clients**
  - demandent des fichiers audio/vidéo (musique, films)
  - sauvegardent les données reçues avant de les 'jouer'
  - les clients peuvent assurer : absorption de la gigue, décompression, correction d'erreur (car ils n'agissent pas en temps réel)
- **Délai**
  - depuis la demande du client jusqu'au début du 'play' : 1 à 10 secondes
  - la réception continue pendant le 'play'.
- **Interactivité de l'utilisateur**
  - possibilités de contrôle : *Pause, resume, rewind, fast forward*…

Applications « streaming stored Audio/video » (2/2)



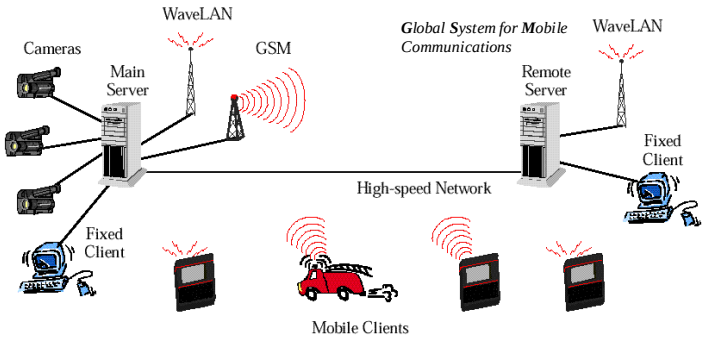
Applications « streaming 1 à m temps réel »
ou « temps réel unidirectionnelles »
ou « Streaming Live Multimedia »

- **Spécificités**
  - Exemples : TV et radio (mais réception via Internet)
  - Plusieurs utilisateurs reçoivent en même temps les mêmes données
- **Délai**
  - Depuis le click sur 'play' jusqu'au début du 'play' : une dizaine de secondes
- **Interactivité de l'utilisateur**
  - Pas d'interaction en général
  - Dans certains cas, on peut faire un pause avec enregistrement automatique, puis reprise en léger différé

Applications « conversationnelles temps réel »

- **Spécificités**
  - Exemple : téléconférence
  - Plusieurs utilisateurs reçoivent en même temps les mêmes données
- **Délai**
  - Vidéo
    délai < 150 msec : acceptable
  - Audio
    délai < 150 msec : bon
    délai < 400 msec : acceptable
- **Interactivité de l'utilisateur**
  - Pas d'interaction dans le sens *Pause*, *Resume*…
  - L'utilisateur est un acteur direct dans la communications (il parle, bouge…)

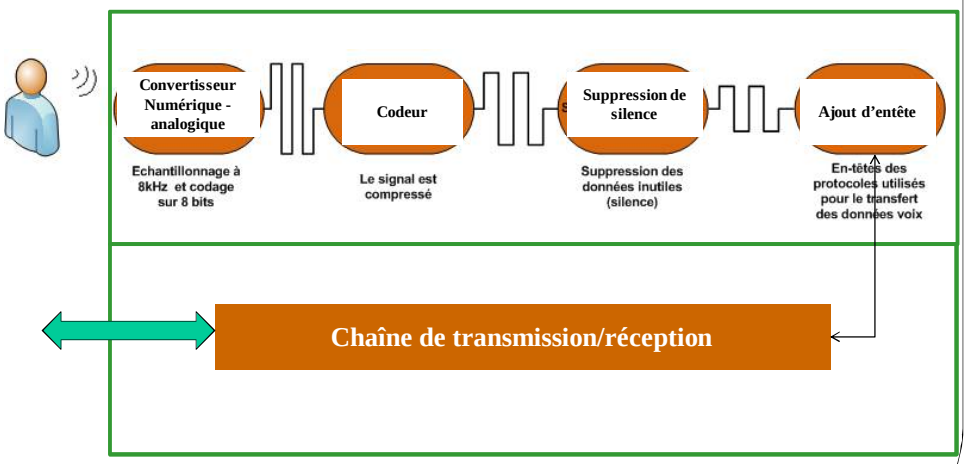
Multimédia en environnement mobile



- Applications :
  - Pompiers, ambulances, TV, VoD, Web, commerce mobile, jeux…

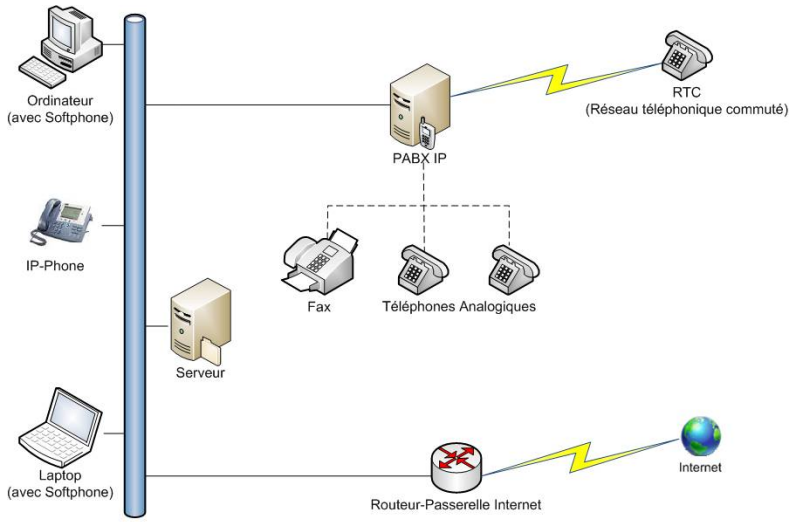
3. Téléphonie sous IP en best effort

De la voix au paquet – Principe général



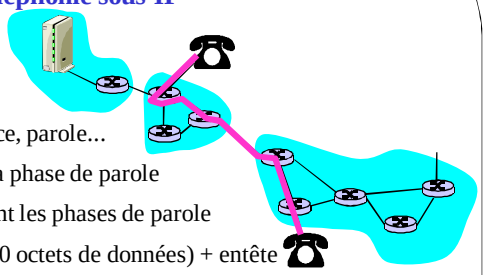
3. Téléphonie sous IP en best effort

Architecture générale du VoIP



3. Téléphonie sous IP en best effort

Principe de la téléphonie sous IP



- Communication téléphonique : parole, silence, parole...
  - Normalement : il faut 64 kb/s pendant la phase de parole
- Les paquets sont générés uniquement pendant les phases de parole
  - Message = un morceau de parole (de 160 octets de données) + entête
- Chaque message est encapsulé dans un segment UDP.
- L'application envoie des segments UDP via la socket-UDP toutes les 20 ms durant les phases de parole. Le débit d'envoi est de 8 kb/s.
- Jusqu'à 10% (voire 20%) de perte de paquets est tolérable.
- Les paquets qui affichent un retard supérieur à 400 ms sont rejetés à la réception.
- La gigue est gérée par l'utilisation d'estampilles de paquets, des numéros de séquence et en retardant certains paquets avant d'être écoutés par le récepteur.

3. Téléphonie sous IP en best effort

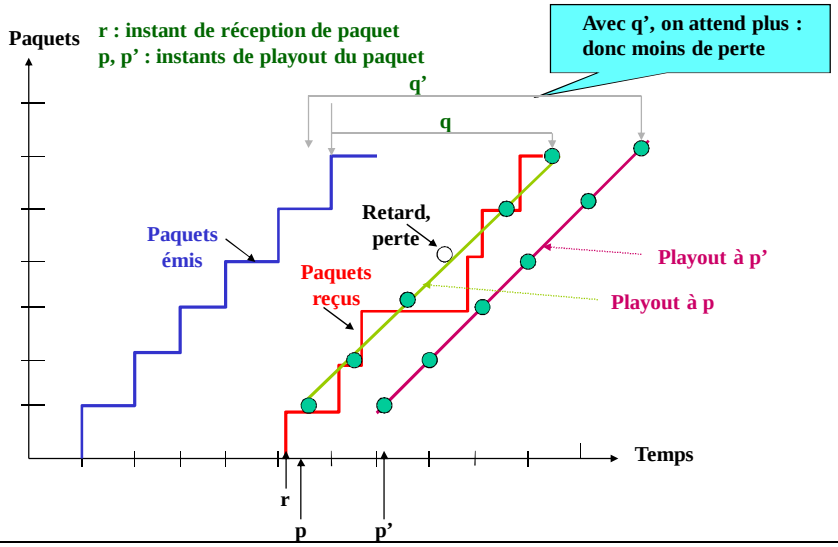
Délai dans la phase de Playout (1/3)

■ **Playout à délai fixe**

- La source produit un morceau de parole et l'émet à l'instant *t*.
- Le récepteur essaie de jouer le morceau reçu *q* msec plus tard :
  - Si le morceau arrive à *t*+*q* ou avant : le morceau est effectivement joué à *t*+*q*.
  - Si le morceau arrive après *t*+*q*, le morceau est perdu.
- Compromis pour la valeur de *q* :
  - Large valeur de *q* : moins de paquets perdus, mais mauvaise interactivité.
  - Petite valeur de *q* : meilleure interaction, mais plus de perte de paquets.

■ On essaie de retarder le plus possible le début du play d'une phase de parole; mais une fois commencée, la phase de parole est jouée avec un rythme fixe.

Délai dans la phase de Playout (2/3)



Délai dans la phase de Playout (3/3)

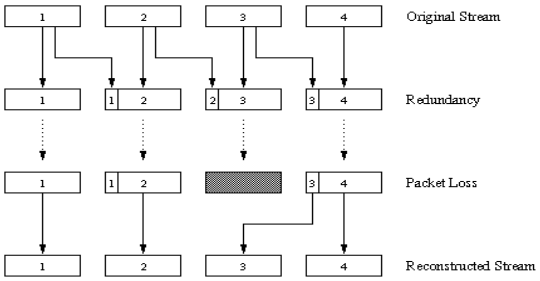
■ **Playout à délai adaptatif**

- L'objectif est d'utiliser une valeur *p - r* qui reflète les performances du réseau en terme de délai (variable) de communication durant un appel téléphonique.
- Le délai de playout est calculé pour chaque morceau en se basant sur la moyenne de délai de communication et la variation observée de ce délai.
- Le début d'une phase de parole est identifié à partir d'estampilles de morceaux successifs et/ou des numéros de séquence des paquets.

Recouvrement en cas de perte de paquets (1/2)

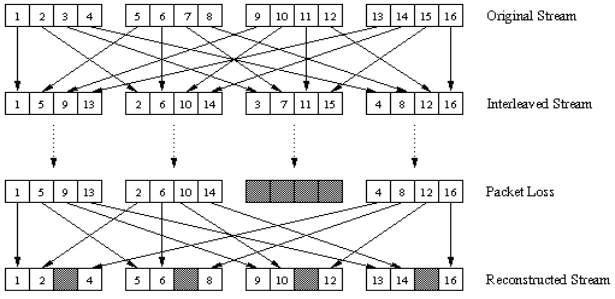
- Comme les retransmissions sont inappropriées dans un contexte temps réel, une stratégie de recouvrement doit être mise en place. Dans le cas de téléphonie sous IP, deux techniques sont utilisées pour réduire l'impact des pertes : FEC (*Forward error correction*) et Entrelacement.

- **Recouvrement par FEC :** Ajouter de l'information de redondance en mixant les valeurs de plusieurs morceaux dans un paquet.



Recouvrement en cas de perte de paquets (2/2)

- **Recouvrement par entrelacement**
  - Pas de redondance, mais il peut engendrer des retards dans le playout.
  - Diviser les périodes de 20 msec de parole en périodes plus petites de 5 ms et entrelacer les petits morceaux.
  - En cas de perte, utiliser des morceaux incomplets (plutôt que de perdre des gros morceaux entiers).



4. Exigences des applications multimédia

Caractéristiques des applications multimédia

- Manipulation de grandes quantités de données 'continues'
- Des débits minimum sont nécessaires
- Livraison des informations en respectant des timings
- Les applications interactives nécessitent des temps d'aller retour faibles
- Coexistence (et partage de ressources) avec des applications non multimédia
- Ressources requises :
  - Processeurs (à haute performance)
  - Serveurs puissants
  - Mémoire principale dédiée (pour la bufferisation par le client)
  - Mémoire disque à grande capacité
  - Bande passante de réseau avec un minimum de latence

Les bonnes quantités
Au bon moment.

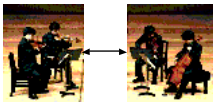
4. Exigences des applications multimédia

Exigences des applications multimédia (1/7)

- Exigences : délai, gigue, débit
- Les valeurs exigées changent avec l'évolution de l'offre technologique
  On ne demande pas les mêmes choses pour une connexion Internet à 56 kb/s qu'à une connexion à 10 Mb/s.
- L'utilisateur (humain) sait à la fois être exigeant et s'adapter à ce qu'on lui offre.
- Tendance actuelle de la demande : des délais de plus en plus courts, des débits de plus en plus élevés, des taux de perte de plus en plus faibles.

Exigences des applications multimédia (2/7)

- Téléphonie et audio conférence
  - Faible débit (~ 64 Kb/s), mais les délais doivent être courts (< 250 ms)
- Vidéo à la demande
  - Débit élevé (~ 10 Mb/s), latence non critique
- Vidéo conférence
  - Débit élevé pour chaque participant (~1.5 Mb/s), délai faible (< 100 ms), états synchronisés.
- Répétition musicale distribuée
  - Débit élevé (~1.4 Mb/s), très faible latence (< 100 ms), haute synchronisation de média (dérive entre son et image < 50 ms).



- Jeux (selon une étude faite par ITU-T)
  - Un délai maximum de 70 ms est plus apprécié par les joueurs qu'un délai de 200 ms.
  - La gigue devra être de 20 ms maximum, car le joueur adapte sa stratégie à un délai fixe (en tirant sur les cibles par exemple). La gigue élevée conduit à un jeu ennuyeux.

Exigences des applications multimédia (3/7)

Common wired-network performance characteristics

Application	Reliability	Delay	Jitter	Bandwidth
E-mail	High	Low	Low	Low
File transfer	High	Low	Low	Medium
Web access	High	Medium	Low	Medium
Remote login	High	Medium	Medium	Low
Audio on demand	Low	Low	High	Medium
Video on demand	Low	Low	High	High
Telephony	Low	High	High	Low
Videoconferencing	Low	High	High	High

Common wireless WAN (3G) network performance characteristics

Application	Reliability	Delay	Jitter	Bandwidth
E-mail	Low	High	–	Low
File transfer	Low–medium	High	–	High
Web access	Low–medium	Medium	–	Medium
Remote login	Low	Low	–	Low
Control	Null	Low	–	–
Real time	Low–medium	Low	Low	Medium–high

Exigences des applications multimédia (4/7)

Typical QoS application requirements in 3G

Type of application and example			(Kbps)	Losses (%)	Delay (ms)
Data		FTP	Limitless	0	TCP timer
Real time	Audio	Voice	≤64	10^{-4}	≤300
		Voice over IP	10–64	$5–10^{-2}$	≤300
	Video	MPEG-4	≤2000	10^{-2}	≤40
		H.320	≤64	10^{-4}	≤40
Non-real time	Audio	CD	150	10^{-4}	Buffer length
	Video	MPEG-4	Limitless	0	Buffer length
Network services			Limitless	0	–

Exigences des applications multimédia (5/7) - in 5G networks

Data rates	1-10Gbps (resp.100s of Mbps)
Capacity	36TB/month/user (resp. 500 GB)
Spectrum	Higher frequencies & flexibility
Energy	~10% of today's consumption
Latency reduction	~ 1ms (e.g. tactile internet)
D2D capabilities	NSPS, ITS, resilience, …
Reliability	99.999% within time budget
Coverage	>20 dB of LTE (e.g. sensors)
Battery	~10 years
Devices per area	300.000 per access node

Ultra-dense networks

Ultra Reliable Comm.

Massive Machines

Exigences des applications multimédia (6/7)

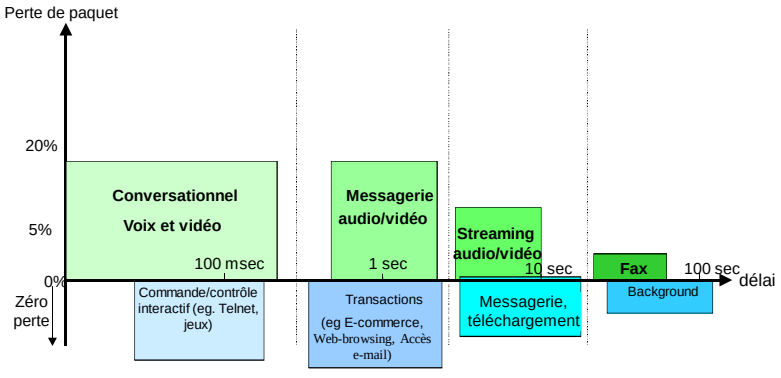
End-user performance expectations—streaming services

Medium	Application	Degree of Symmetry	Data rate (Kbps)	Key performance parameters and target values		
				Startup delay (s)	Transport delay variation	Packet loss at session layer
Audio	Speech/music medium/high quality	Primarily one-way	5–128	< 10	<2 s	<1% Packet loss ratio
Video	Movie clips surveillance real-time video	Primarily one-way	20–384	< 10	<2 s	<2% Packet loss ratio
Data	Bulk data transfer/retrieval, layout, synchronisation information	Primarily one-way	<384	< 10	NA	Zero
Data	Interactive games	Primarily one-way		< 10	NA	Zero

End-user performance expectations—interactive services

Medium	Application	Degree of Symmetry	Data rate	Key performance parameters and target values		
				One-way delay	Delay variation	Information loss
Audio	Voice messaging	Primarily one-way	4–13 Kbps	<1 s for playback <2 s for record	<1 ms	<3% Frame error rate
Data	Web-browsing—HTML	Primarily one-way		<4 s/page	NA	Zero
Data	Transaction services—high priority, e.g. ATM	Two-way		<4 s	NA	Zero
Data	E-mail (server access)	Primarily one-way		<4 s	NA	Zero

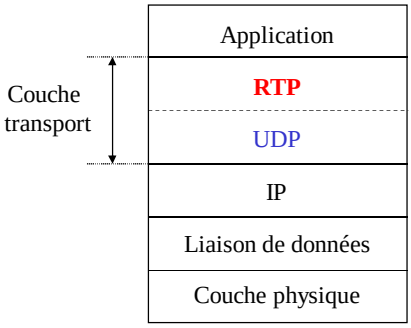
Exigences des applications multimédia (7/7)



5. Protocoles RTP et RTCP

Real-Time Protocol (RTP) (1/3)

- RTP : une solution pour les AMM avec Internet en best effort
- Fonctionne essentiellement au dessus de UDP



5. Protocoles RTP et RTCP

Real-Time Protocol (RTP) (2/3)

- **Entête de paquet RTP**
  - Type de flux (7 bits)
  - Numéro de séquence (16 bits) : utilisé pour détecter les pertes.
  - Estampille (32 bits) : fournit l'instant d'échantillonnage du premier octet du paquet. Elle est utilisée pour absorber la gigue.
  - Identificateur de source de synchronisation (32 bits) : identifie la source du flux. Chaque flux dans RTP a un identificateur affecté par la source de manière aléatoire (mais distinct de ceux déjà existants) au début du flux.

Type de flux	Numéro de séquence	Estampille	Identificateur de source de synchronisation	Données

Real-Time Protocol (RTP) (3/3)

- **Champ Type de flux**

Quelques types de flux audio supportés par RTP

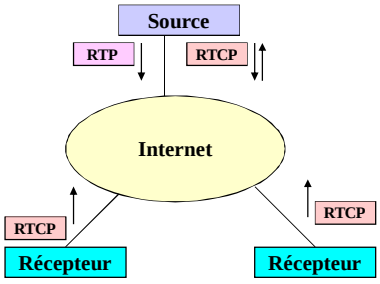
Type de flux	Format audio	Echantil-lonnage	Débit
0	PCM	8 KHz	64 Kb/s
1	1016	8 KHz	4.8 Kb/s
3	GSM	8 KHz	13 Kb/s
7	LPC	8 KHz	2.4 Kb/s
9	G.722	8 KHz	48-64 K/ps
14	MPEG Audio	90 KHz	----
15	G.728	8 KHz	16 Kb/s

Quelques types de flux vidéo supportés par RTP

Type de flux	Format vidéo
26	Motion JPEG
31	H.261
32	MPEG1 Vidéo
33	MPEG2 Vidéo

Real-Time Control Protocol (RTCP) (1/2)

- RTCP permet d'acheminer des paquets contenant des rapports concernant un flux multimédia entre une source et un récepteur.
- Les rapports contiennent des statistiques sur : le nombre de paquets transmis, le nombre de paquets perdus, la gigue de transfert…
- Les paquets rapports sont envoyés par les récepteurs, éventuellement à la demande des sources.
- Les paquets rapports sont utilisés par la source pour modifier/adapter son rythme aux conditions du réseau.



Real-Time Control Protocol (RTCP) (2/2)

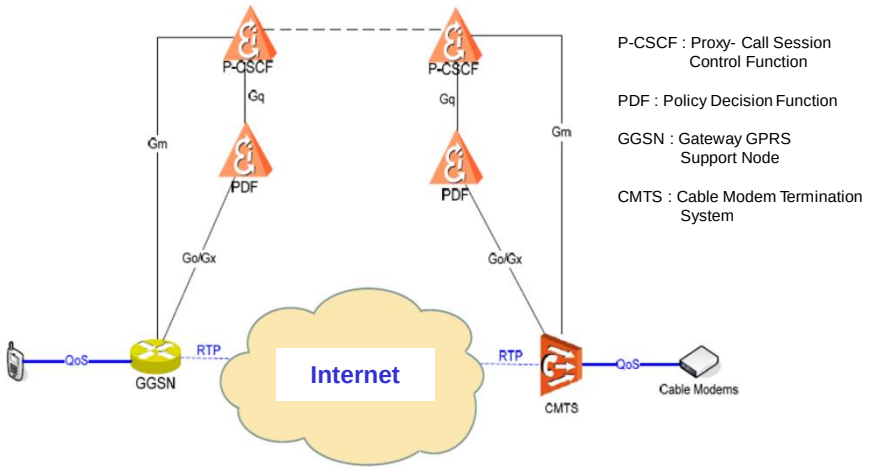
- Si chaque récepteur envoie ses paquets-rapports à tous les autres sources/récepteurs du flux : surcharge importante du réseau.
- RTCP ajuste les intervalles de temps entre rapports en fonction du nombre de récepteurs participant à un flux.
- Typiquement, la bande passante utilisée pour RTCP est limitée à 5% de la bande passante de la session. Cette fraction est partagée entre les demandes de rapports émises par les sources (25%) et les rapports émis par les récepteurs (75%)
  - T_s : période de transmission de paquet RTCP par la source :

$$T_s = \frac{Nombre\ de\ sources}{5\%*25\%*Bande\_passante\_session} *Taille\_paquet\_moyen\_RTCP$$

  - T_r : période d'émission de paquet RTCP par un récepteur :

$$T_r = \frac{Number\ de\ récepteurs}{5\%*75\%*Bande\_passante\_session} * Taille\_paquet\_moyen\_RTCP$$

RTP et UMTS



P-CSCF : Proxy- Call Session Control Function

PDF : Policy Decision Function

GGSN : Gateway GPRS Support Node

CMTS : Cable Modem Termination System

Etat actuel des standards 3GPP

6. Conclusion

Passé: Diffusion de vidéo unidirectionnelle

Présent, Futur: Vidéo interactive à la demande

6. Conclusion

Convergence

Aujourd'hui

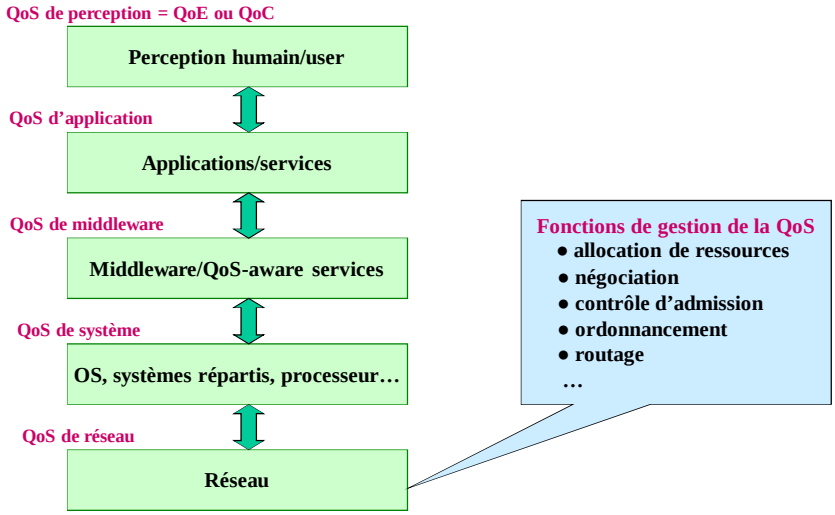
Demain

6. Conclusion

- **Applications multimédia**
  - Une réalité
  - Elles touchent à tous les secteurs d'activités
- **Challenges du multimédia**
  - Conception de contenu, ergonomie
  - Support de média continus
  - Synchronisation temps réel
  - Gestion de qualité de service (QoS)
  - Communication multi partite sur des réseaux filaires ou non filaires
  - Gestion de la mobilité des clients et des serveurs
  - Interopérabilité d'équipements hétérogènes
  - Coût faible des équipements et des logiciels
  - Equipements performants (écran, processeur, stockage...) embarqués et transparents

6. Conclusion

Mutimédia et QoS



Multimédia sur l'Internet actuel

- **TCP et UDP actuels**
  - PAS de garantie de délai ou perte
  - Seulement un service *best effort* est fourni

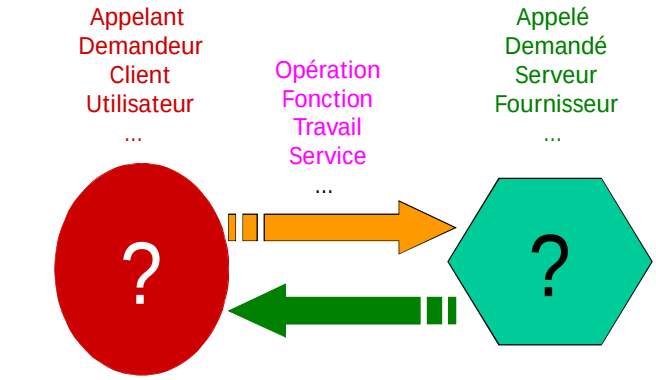
- Aujourd'hui les applications MM utilisent des techniques de niveau *Application* pour remédier aux insuffisances d'Internet, ce qui conduit parfois à des services peu appréciés par l'utilisateur final.

- **Il faut des mécanismes appropriés de gestion de la QoS à l'intérieur d'Internet.**

Chapitre 2

Concepts et mécanismes de base de la qualité de service

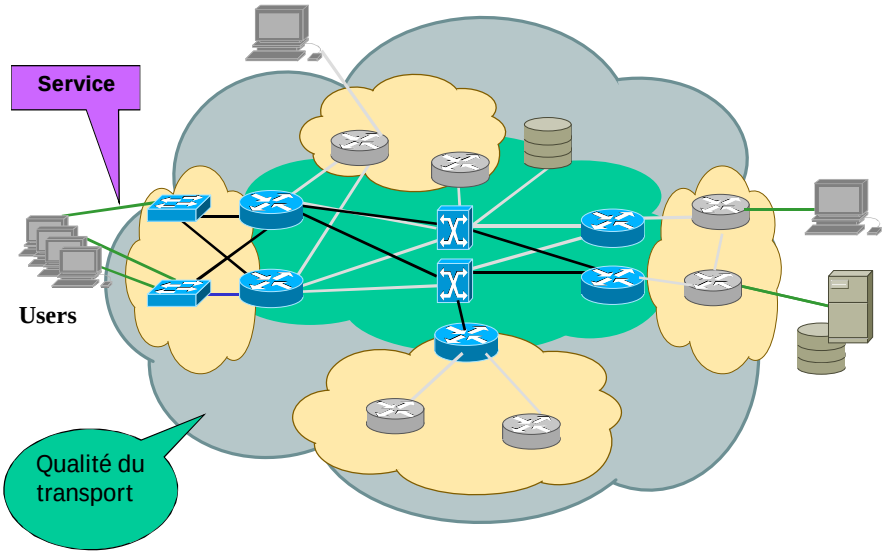
1. Introduction : étendue de la QoS



Qualité

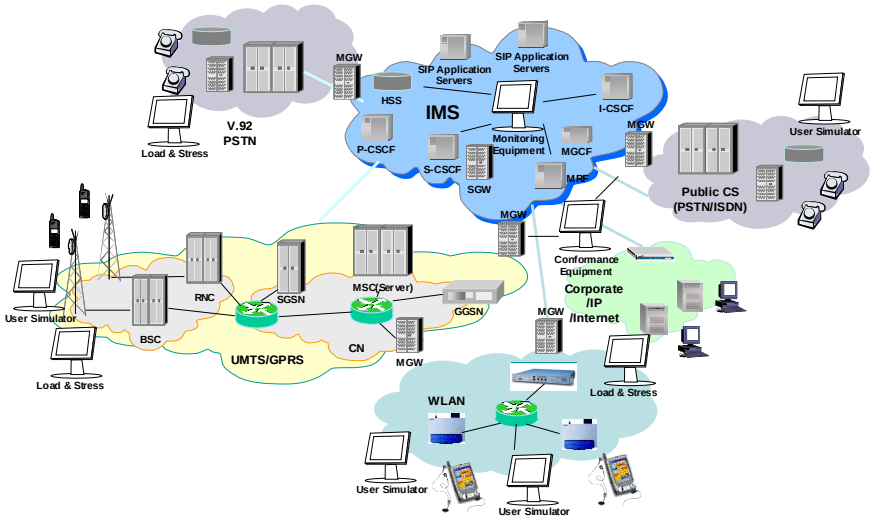
– Exigée, souhaitée, implicite/explicite, convenue à l'avance,
  Faire confiance (Aveugle, Vérifiable, Vérifiée)
– Mesurable (Qualitative/Quantitative) ou non
– Mesurable en ligne / hors ligne

1. Introduction : étendue de la QoS



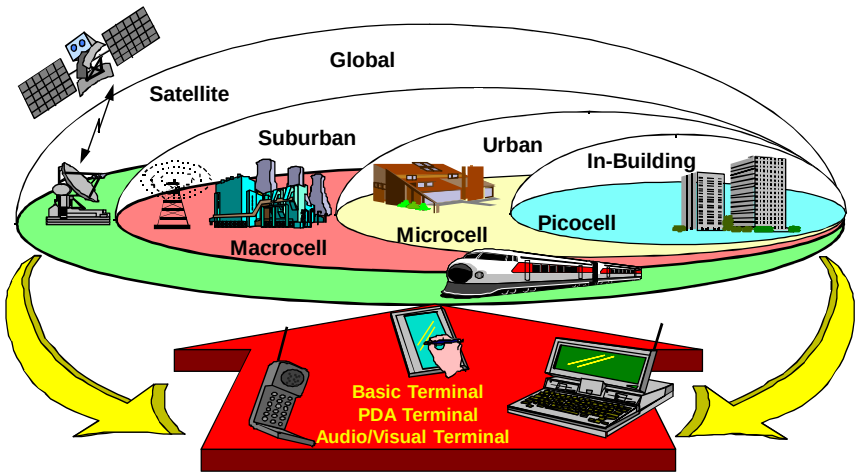
1. Introduction : étendue de la QoS

➔ **Diversité des Réseaux**

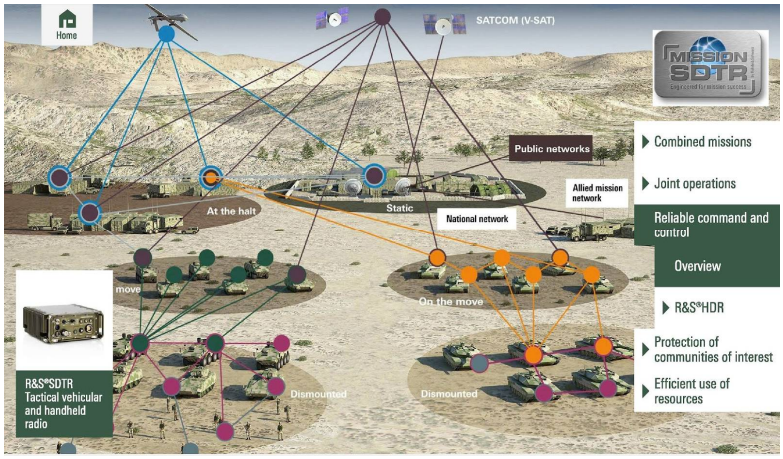


1. Introduction : étendue de la QoS

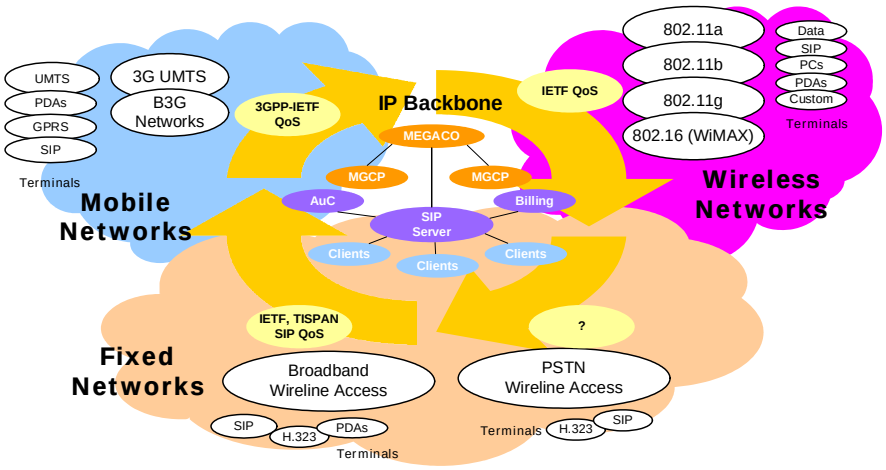
➔ **Diversité des Réseaux**



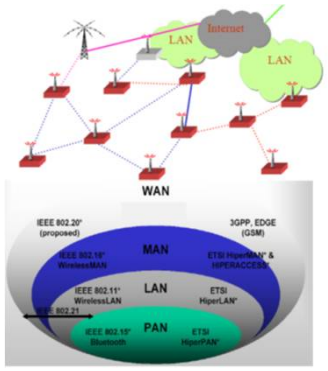
➔ Diversité des Réseaux



➔ Diversité des Réseaux



➔ Diversité des Réseaux



- **Réseaux publiques, privés, dédiés…**
- **Filaires - Sans fil (ondes radio, IR, Satellite)**
- **PAN, LAN, MAN, WAN**
- **Industriels, embarqués, bureautiques…**
- **Site : salle de TP, gare, train, avion, voiture…**
- **1 domaine, n domaines**
- **Administration : centralisée, répartie, autonomie**
- **Environnement : montagne, tunnel, chaleur, humidité, rayonnement…**
- **Réseaux : invisibles, intelligents, self-organizing, accessibles partout…**
- **…**

➔ Diversité du public concerné

- **Genres d'intervenants (utilisateurs)**
  - Personne, robot, capteur, objet logiciel…
  - Informaticien, automaticien, opérateur de télécom, fournisseur d'accès, militaire, industriel … grand public
  - Très exigent (« le réseau est censé répondre aux exigences ») …., on prend ce qu'offre le réseau (« se contenter de ce le réseau offre »)
  - Accepte de négocier, veut tout préconfiguré…
  - Accepte un coût : très élevé, …, modeste, gratuit
  - …
- **Vues des intervenants**
  - Développement de réseaux et de services
  - Contenu et sa diffusion
  - Contenu et son utilisation
  - Transport de bout en bout
  - Transport sur un domaine, un routeur ou une antenne
  - …

➔ Diversité des Applications

➔ Diversité des Applications

- **Secteurs / domines d'activité**
  - Commande/supervision de centrales nucléaires
  - Organisation de commandement militaire
  - Santé
  - Transport
  - Vidéo surveillance, identification de personnel
  - Contrôle-commande
  - Commerce électronique
  - Loisirs, Musique, Jeux
  - ...
- **Nature des échanges**
  - Critiques ou non
  - Applications : Transactionnelles, Réactives, Interactives...
  - *1* vers *1*, *1* vers *m*, *m* vers *1*, *n* vers *m*
  - ...

➔ Diversité des Equipements

➔ Diversité des Equipements

- **Grand public, privé, spécialisé**
- **Fixe, mobile (mobilité lente ou rapide)**
- **Cher / pas cher**
- **Disparaît après utilisation ('sensor') ou non**
- **Ecoute : toujours à l'écoute, dormant, émetteur, récepteur…**
- **Avec contraintes de batterie (rechargeable ou non)**
- **Localisable : à la demande, toujours, de manière intelligente**
- **Equipement intelligent ou non**
- **…**

➔ **Diversité des acteurs de normalisation**

➔ **Difficultés de parler de QoS**

- **Multiforme (temps, sécurité, coût…)**
- **Différentes vues (grand public,…, Informaticien)**
- **Différents niveaux (application, réseau, physique…)**
- **Différents mécanismes et moyens**

➔ **Cours limité au Réseau (transport de données)**

2. Concepts et définitions

Définitions de la QoS

- **Définition de l'ISO et ITU-T [ISO/CEI 13236 - X.641 ; Décembre 1997]**
  « C'est un ensemble d'exigences de qualité sur le comportement collectif d'un ou de plusieurs objets »

- **Définition de l'IETF**
  « La qualité de service désigne la manière dont le service de livraison de paquets est fourni et qui est décrite par des paramètres tels que la bande passante, le délai de paquet et les taux de perte de paquets »

- **Définition de QoS Forum**
  « Mesure collective du niveau de service fourni au client. La QoS peut être caractérisée par différents critères de performance de base qui incluent la disponibilité, le taux d'erreurs, le temps de réponse, le temps d'établissement de connexion, le débit de données, la perte de connexion ou de données à cause de congestions du réseau et la rapidité de détection et de correction de fautes »

2. Concepts et définitions

Définitions de la QoS

➔ **Aspects liés à la QoS**

– Exprime des exigences sur le comportement d'un fournisseur de service

– S'exprime par différents types de paramètres (délai, disponibilité de service,…)

– Implique différents niveaux de service (déterministe ou autres)

– Nécessite la mise en place de divers mécanismes (réservation, contrôle,...)

– Concerne aussi bien le réseau que les applications

– Concerne à la fois différents types d'équipements et différentes couches

Classes (niveaux) de service

- Garantie absolue (déterministe)
- Probabiliste/stochastique/statistique
  - Prédictive, à charge contrôlée
  - Meilleure que le meilleur effort
  - 'Molle'
  - coercitive
- Meilleur effort

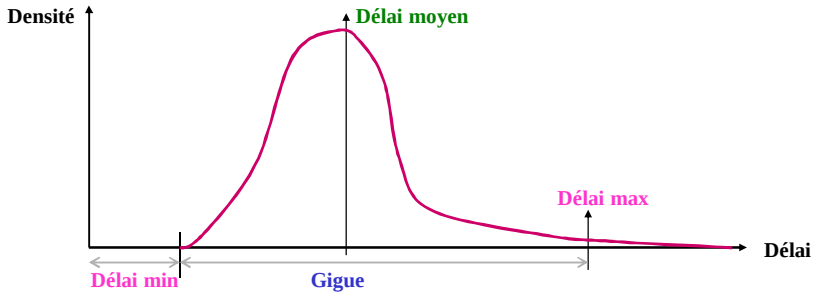
Quel niveau choisir ?

C'est la nature de l'application qui permet de décider

Coût

Paramètres de QoS (1/4)

- **Aspects temporels**
  - Temps de transfert, latence, délai
  - Gigue
  - Temps de réponse (aller-retour)
  - Temps d'établissement/fermeture de connexion
  - ...



Courbe de distribution du délai de transfert

Paramètres de QoS (2/4)

■ **Sensibilité au délai**

– L'être humain n'aime pas attendre en général (les applications qu'il utilise doivent donc en tenir compte). Téléphone sous IP actuel, visioconférence : son et images saccadés…

– Les systèmes commandés sont sensibles au délai (réaction tardive pour un ABS, un airbag, un robot en fin de course…).

■ **Sensibilité à gigue**

– L'être humain est sensible à la gigue du son et image

– Certains systèmes de commande sont sensibles à la gigue.

Paramètres de QoS (3/4)

■ **Volume**

– Bits/s, Paquets/s

– Pourcentage de bande passante

– …

■ **Fiabilité/disponibilité/robustesse**

– Taux de disponibilité

– MTBF, MTTR

– …

■ **Paramètres d'erreurs**

– Taux d'erreur, taux de perte

– Taux de désordre de paquets

– Erreur d'établissement/fermeture de connexion

– ...

Paramètres de QoS (4/4)

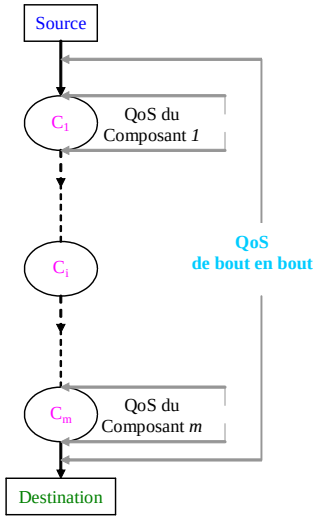
- **Coût**
  - Coûts (€, autres)
  - Pénalité, bonus, ...
- **Autres** (facilité d'utilisation, maintenabilité, simplicité, visibilité, efficacité, extensibilité, passage à l'échelle, interopérabilité…)
- **Sécurité**
  - Capacité du contrôle d'accès
  - Capacité du chiffrement
  - Types d'attaque pris en compte et capacité de résistance
  - Surcoût des mécanismes de sécurité
  - ...

Formes d'expression de la QoS

- **Déterministe**
  - Une valeur `(délai < 10 ms)`
  - Un intervalle de valeurs `(délai dans [80 .. 100])`
- **Probabiliste**
  - Avec une probabilité `P (délai < 100 ms à 90%)`
  - Plus souple
  - Difficulté de choisir les bonnes probabilités
- **Statistique**
  - Expression sur la moyenne, variance, écart type
  - Expression sur la fréquence
  - Loi de distribution
  - Expression (m,k)-firm
  - Expression en logique floue (ex. "débit élevé ", " délai acceptable ", " gigue raisonnable ")

Types de métriques de QoS (1/2)

- **Additive**
  - $QoS(C_1 ; C_2) = QoS(C_1) + QoS(C_2)$
  - ex. Délai
- **Multiplicative**
  - $QoS(C_1 ; C_2) = QoS(C_1) * QoS(C_2)$
  - ex. Disponibilité
- **Concave**
  - $QoS(C_1 ; C_2) = \min\{QoS(C_1), QoS(C_2)\}$
  - ex. Débit
- **Autres**
  - Cas des spécifications non homogènes



Types de métriques de QoS (1/2)

- **Croissante (increasing)**
  - si $Val(QoS_1) > Val(QoS_2)$ alors QoS_1 est meilleure que QoS2
  - ex. débit
- **Décroissante (decreasing)**
  - si $Val(QoS_1) < Val(QoS_2)$ alors QoS_1 est meilleure que QoS2
  - ex. délai

Notion de Contrat

SLA (Service Level Agreement)

- Gestion orientée connexion vs Gestion orientée SLA
- SLA = Contrat User-Provider
- SLA Statique ou Dynamique
- SLA = {

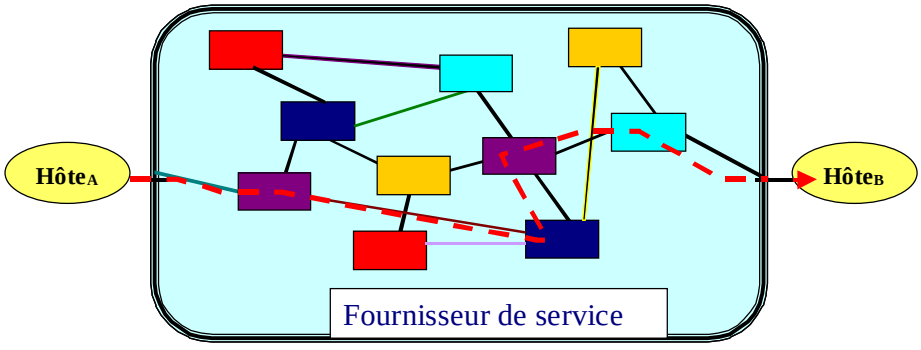
QoS requise,
Spécification de trafic,
Règles de traitement de paquets,
(Service Level Specification)

Coûts, pénalisation, bonus,
Aspects juridiques
…
}

QoS de bout en bout

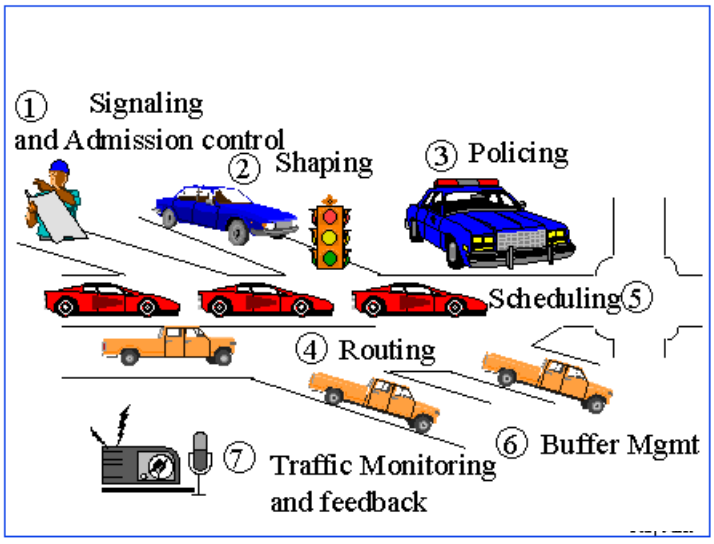
➔ **QoS de bout en bout - QoS locale**

– Utilisateur final → QoS de bout en bout
– Fournisseur de service → Décomposition de la QoS de bout en bout : QoS locale

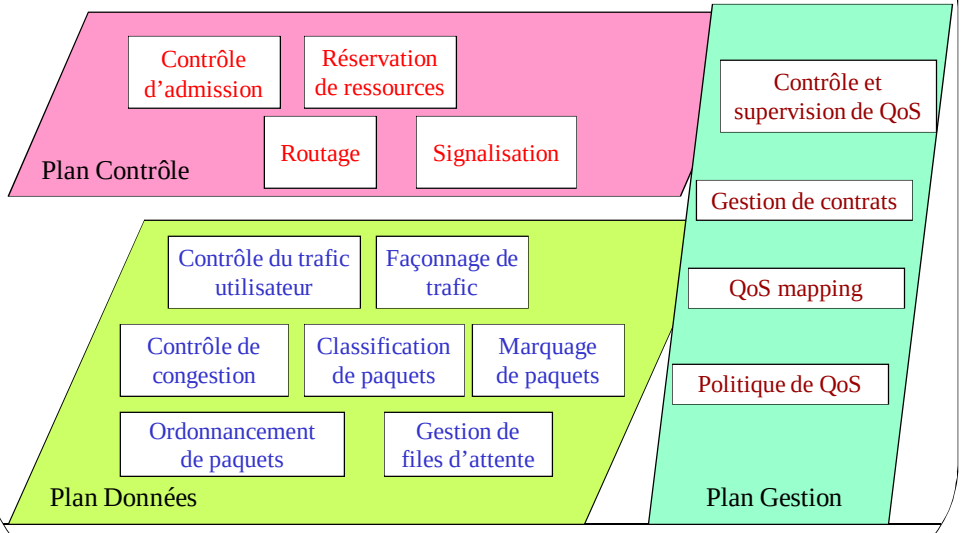


3. Panorama des fonctions de gestion de QoS

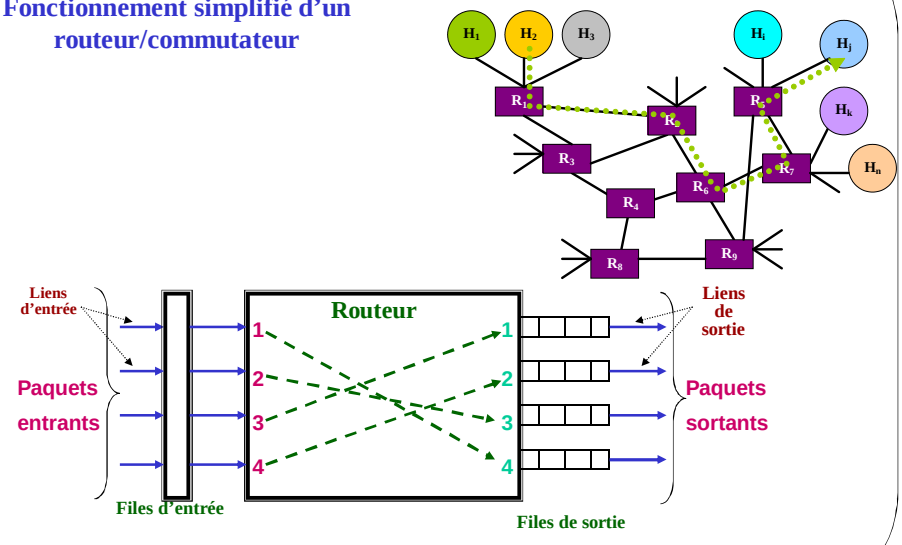
3. Panorama des fonctions de gestion de QoS



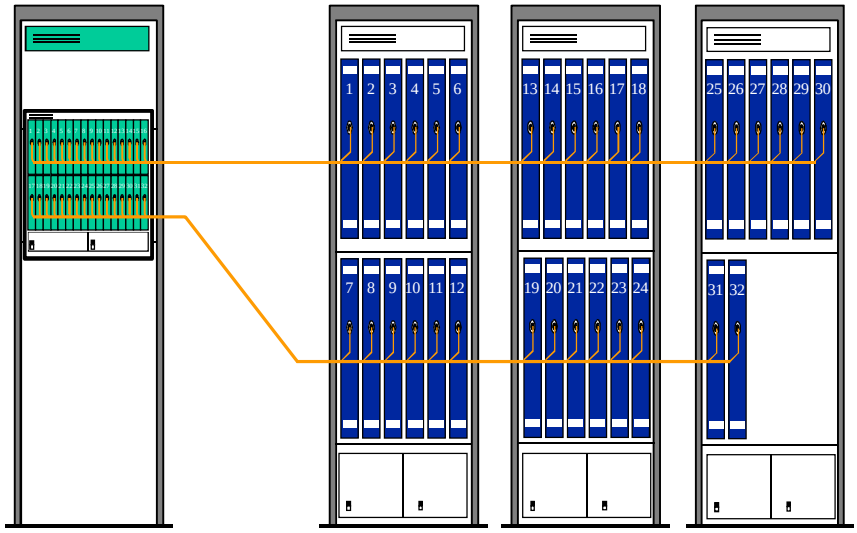
Fonctions mises en œuvre pour la garantie de QoS



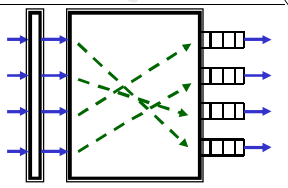
Fonctionnement simplifié d'un routeur/commutateur



Routeurs/commutateurs actuels



Eléments du délai de bout-en-bout



- Délais d'attente dans les files d'entrée
- Délais de construction de paquets
- Délais de commutation
- Délais d'attente dans les files de sortie
- Délais de transmission
- Délais de propagation

Délais négligeables et/ou constants

4. Conclusion

Entités concernées par la QoS

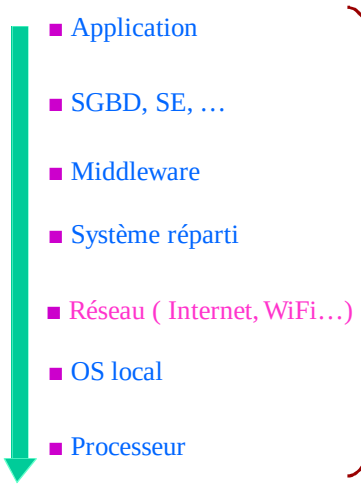
Besoins de niveau applicatif

- Qualité de l'image ou du son
- Disponibilité de l'outil de production
- Disponibilité du système de visioconférence
- Sécurité des échanges
- Coûts, Consommation d'énergie
- …

- Entité d'exécution (Tâche, Processus, Thread, Transaction, Agent, …)
- Entité d'information (Donnée (fraîcheur), Mesures (corrélation), …)
- Entité d'échange (Trame, Paquet, Message, …)

4. Conclusion

Niveaux de prise en compte de la QoS



Problèmes à résoudre

- **Modèles d'expression de QoS**
- **Fonctions de gestion de QoS**
- **Validation/vérification de QoS**

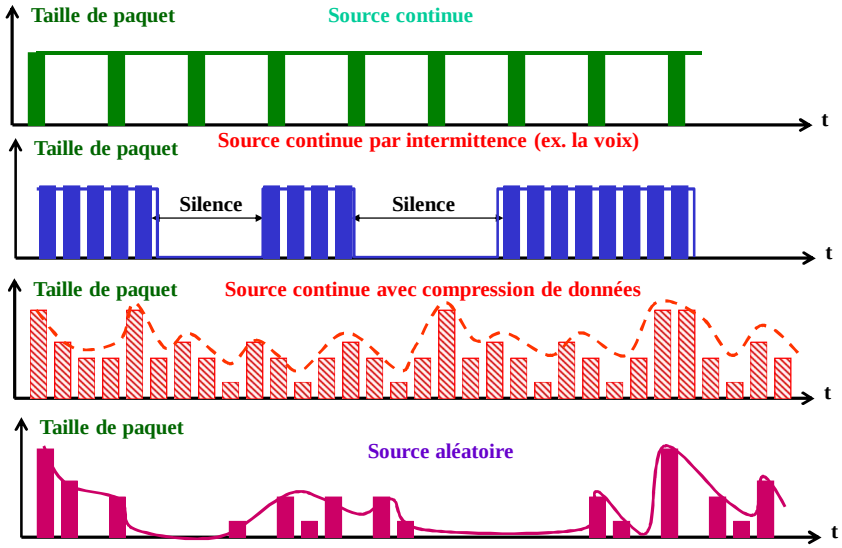
Chapitre 3

Modèles de trafic et Contrôle d'admission

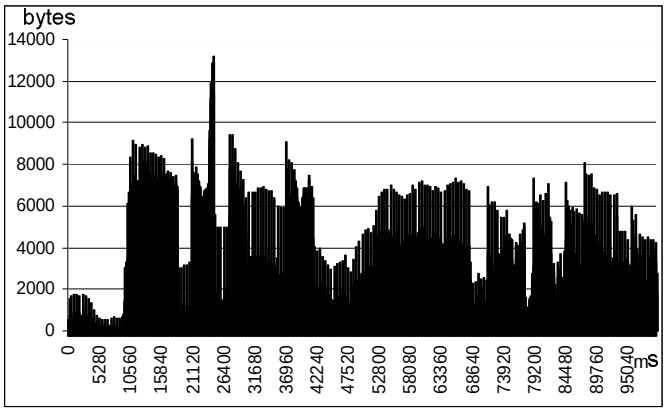
Chapitre 3

Modèles de trafic et Contrôle d'admission

1. Besoins des applications

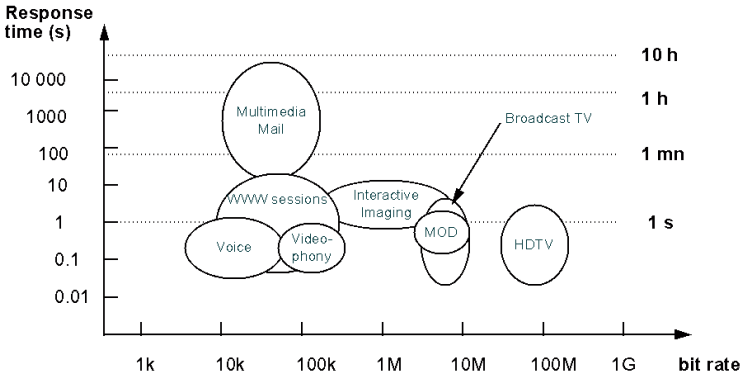


1. Besoins des applications



Distributon de la longueur de trame du film Jurassic park I codé en MPEG-4

1. Besoins des applications



Quelques exemples de contraintes de débit et délai
(à prendre avec précaution car ces chiffres changent sans cesse)

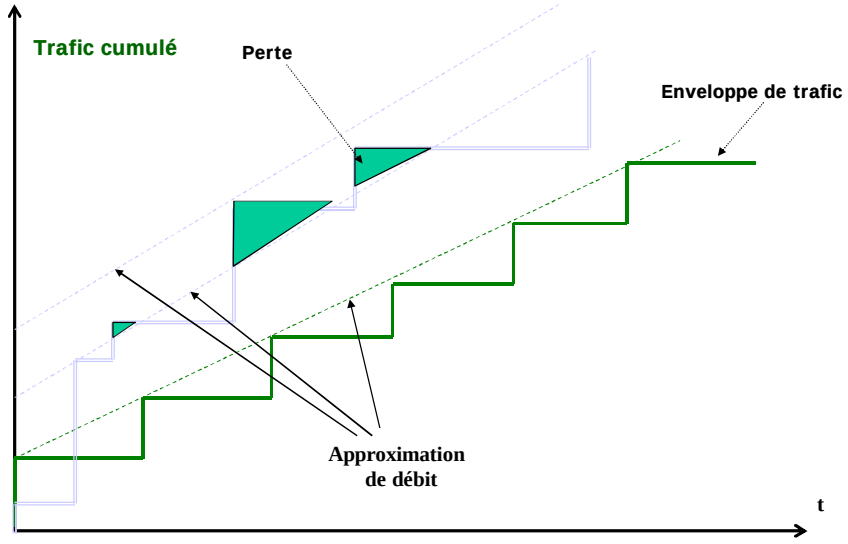
1. Besoins des applications

Type	Service	Débit	délai	Gigue	Taux de perte
Conversationnel / Temps réel	Voix	4-25 kbit/s	<150 ms	<1 ms	<3% FER
	Vidéophone	32-384 kbit/s	<150 ms		<1% FER
	Jeux	<1 kbit/s	<250 ms		<3% FER
	Messagerie vocale	4-13 kbit/s	<1 sec	<1 ms	<3% FER
Interactive	Web browsing		4 sec/page		
	e-commerce		4 sec		<0% FER
	Streaming audio	32-384 kbit/s	<10 sec	<1 ms	<1% FER
Streaming	Vidéo	32-384 kbit/s	<10 sec		<1% FER

FER : Frame Error Rate

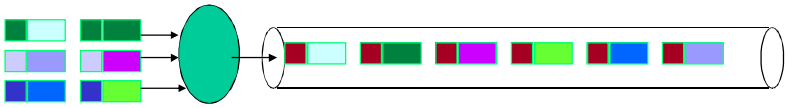
Quelques exemples de contraintes de débit et délai
(à prendre avec précaution car ces chiffres changent sans cesse)

1. Besoins des applications



1. Besoins des applications

Agrégation de flux



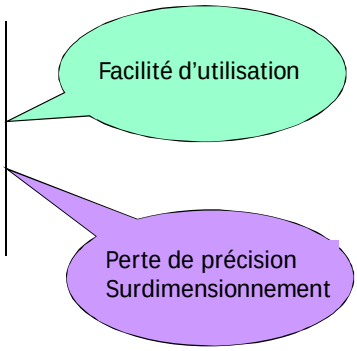
- **Besoins**
  - Plusieurs flux à transporter avec un même niveau de QoS
  - Multiplexage

- **Difficultés**
  - Choix des flux à agréger
  - Précision du trafic agrégé vs complexité de l'agrégation
  - Modèles de dégradation de QoS pour les flux individuels

2. Modèles de trafic

Propriétés des modèles de trafic

- **Simplicité d'expression**
- **Facilité de vérification et de test**
- **Surcoût d'implantation faible**



2. Modèles de trafic

Caractérisation de trafic

- Trafic périodique : aisé
- Trafic apériodique
  - Distribution des instants d'arrivée selon quelle loi (Poisson, …) ?
  - Taille maximale des avalanches ?
  - Durée minimale d'avalanche ?
  - Distribution de la taille des avalanches ?
  - Distribution des pertes de messages ?
  - Corrélation entre les paquets (pour autoriser les pertes) ?
    - Souvent difficile à modéliser :
      choix de paramètres pour "convenance mathématique"
    - Reste beaucoup à faire pour modéliser le trafic aléatoire/sporadique

2. Modèles de trafic

Modèles de trafics fréquemment utilisés (1/2)

➔ **Modèle périodique**
- Période, Longueur maxi de paquet

➔ **Modèle-1 avec rafale (Ferrari)**
- *Lpmax : longueur maxi de paquet*

Xmin (intervalle de temps min entre deux messages successifs)
Xave (intervalle de temps moyen entre deux messages successifs)
I (intervalle de temps sur lequel *Xave* est calculé).

➔ **Modèle-2 avec rafale (Cruz)**
- Débit moyen ρ et taille de rafale σ :

Nombre total de paquets générés n'excède jamais $\sigma + \rho T$ dans tout intervalle *T*.

➔ **Modèle-3 avec rafale (Seau percé)**
- Débit moyen d'écoulement du seau (ρ) et la taille maximale du seau (σ).

Eviter le débordement du seau.

➔ **Modèle-4 avec rafale (Seau à jeton)**
- Débit moyen de génération de jeton (ρ) et nombre maximal de jetons en attente (σ). La source ne peut transmettre que si elle a des jetons.

2. Modèles de trafic

Modèles de trafics fréquemment utilisés (2/2)

➔**Modèle de trafic de l'IETF** (RFC 2215)

- Spécification à l'aide d'un ***TSpec*** :
  - Taille σ et débit ρ de seau percé
  - Débit maximum p
  - Taille maximum de paquet M
- Borne sup, A(T), de trafic par intervalle de temps T :

$$A(T) \leq \min(M + pT,\ \sigma + \rho T)$$

➔ **Autres modèles : probabiliste, stochastique,…**

☞ **Coût et performance du CA dépendent des caractéristiques de trafic**

3. Contrôle d'admission

- **Objectif**
  - Est-ce que tous les nœuds à traverser acceptent le nouveau flux ?
  - Est-ce que le nouveau flux peut affecter la QoS des flux déjà acceptés ?
  - Est-ce que le nœud peut offrir la QoS requise par le nouveau flux ?
  - Est-ce que le nouveau flux a le droit d'utiliser les ressources du nœud ?

- **Informations utilisées**
  - Caractéristiques du nouveau trafic et de la QoS demandée
  - Etat et historique du réseau
  - Dates de fin des trafics déjà acceptés
  - Perturbations éventuelles de la QoS des trafics déjà acceptés
  - Politique d'utilisation des ressources

- Le CA peut se faire sur la base de connexion ou de SLA

3. Contrôle d'admission

Propriétés

(à prendre en compte durant la conception d'un CA)

- Décisions incrémentales (ne pas toujours considérer tous les flux)
- Exactitude (compliquée à cause des phénomènes aléatoires)
- Complexité
  - Problème de la diversité des modèles de flux
  - Utilisation en ligne sans surcoût important
- Flexibilité
  - Problème de la diversité des modèles de flux
- Passage à l'échelle

3. Contrôle d'admission

Exemples de Contrôle d'admission déterministe

- **CA pour WFQ**

$$r_{new} + \sum_{i=1}^{n} r_i \leq r$$

Débit du nouveau flux

Somme des débits des flux déjà acceptés

Débit Max du lien

- **CA pour DEDD**

$$D_{new} \geq \sum_{i=1}^{m} DL_i$$

Délai exigé par le nouveau flux

Somme de délais Dus à tous les flux

3. Contrôle d'admission

Contrôle d'admission statistique (1/5)

- **Pourquoi on en a besoin?**
  - La plupart des flux sont plutôt à caractère aléatoire
  - Eviter de prendre des décisions d'admission pessimistes en rejetant des flux qui pourraient être acceptés si on fait un peu plus attention à l'allocation des ressources
- **Risques d'utilisation de CA statistique**
  - Apparition de situations de congestion
  - Dégradation de la QoS
  - Conséquence : CA statistique non adapté aux applications critiques
- **Difficultés d'utilisation** : maîtrise des probabilités/statistiques

3. Contrôle d'admission

Contrôle d'admission statistique (2/5)

- **Types de CA statistique**
  - **CA basés sur les débits moyen et maximal**
  - **CA basés sur la bande passante effective cumulée**
  - **CA basés sur l'ingénierie de la courbe de perte**
  - **CA basés sur la variance maximale**
  - **CA basés sur la théorie des larges déviations**
  - **Autres types**

3. Contrôle d'admission

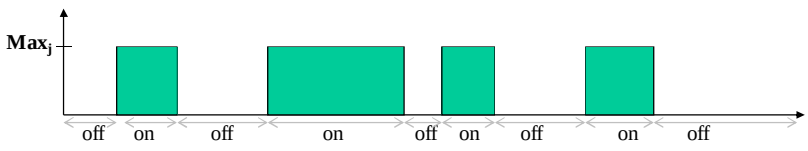
Contrôle d'admission statistique (3/5)

Exemple 1 : CA statistique basé sur les débits moyen et maximum (1/2)

Pour la garantie du taux de perte

- **Notations**
  - **C : capacité du lien considéré**
  - **Max_j : débit max du flux j**
  - **Avr_j : débit moyen du flux j**

- **Hypothèses**
  - **Toute source *j* est de type on-off (soit elle émet à son débit max soit elle est silencieuse)**
  - **Tous les paquets ont la même taille (1 unité)**
  - **Pas de buffer au niveau du lien pour stocker les paquets en attente de transmission**



Contrôle d'admission statistique (4/5)

Exemple 1 : CA statistique basé sur les débits moyen et maximum (2/2)

Pour la garantie du taux de perte

■ **Principe**

– La source *j* étant on/off, la densité de probabilité de son trafic est $f_j(x)$:

$$f_j(x) = \begin{cases} Avg_j / Max_j & si \quad x = Max_j \\ 1 - (Avg_j / Max_j) & si \quad x = 0 \end{cases}$$

– Si on considère *N* flux indépendants qui partagent le même lien, alors la densité de probabilité du flux agrégé, $q(x)$, est la convolution de $f_1, \ldots, f_N$:

$$q(x) = (f_1 * f_2 * \ldots * f_N)(x)$$

– La probabilité de perte de paquets *Pl* pour *N* flux est :

$$Pl^{(N)} = \frac{TraficExcedentaire}{Ch\arg eDeTrafic} = \frac{\sum_x (x - C)^+ q(x)}{\sum_{i=1,\ldots,N} Avg_i}$$

– Test de CA : si $Pl^{(N+1)} \leq$ Taux de perte requis, alors accepter le $N+1^{ème}$ flux, sinon le refuser.

Contrôle d'admission statistique (5/5)

Exemple 2 : CA statistique basé sur la bande passante effective cumulée

Pour la garantie de la bande passante

■ **Notations**

- **C : capacité du lien**
- **B : taille de queue du routeur**
- **$A_j[0, t[$: quantité de bits transmis par la source j dans l'intervalle [0, t[**
- **Pl : taux de perte d'une queue de taille maximale *B***
- **$E_j(Pl)$: bande passante effective du flux *j* (il y a différentes manières de la définir)**
- **N : nombre de flux multiplexés**

■ **Test du CA**

$$\sum_{j=1}^{N} E_j(Pl) < C$$

■ **Exemple de définition de $E_j(Pl)$ si *Pl* peut être définie par une loi exponentielle : $Pl = e^{-\delta B}$**

$$E_j(Pl) = Ave_j + \frac{\delta}{2B} \lim_{t \to \infty} \frac{1}{t} \mathrm{var}(A_j[0,t[)$$

3. Contrôle d'admission

Contrôle d'admission basé sur les mesures

- **Si les caractéristiques de flux sont peu variables**
  - Utilisation de la demande maximale et moyenne pour accepter le flux
  - Décision et réservation définitives
- **Si les caractéristiques de flux sont peu ou pas connues (imprécision de trafic)**
  - Utiliser une estimation initiale du trafic et réserver les ressources
  - Effectuer des mesures sur le trafic et ajuster les réservations en re-estimant le trafic
  - Accepter un plus grand nombre de flux
  - Coût des mesures et efficacité réelle des ajustements
- Problèmes
  - Que faut-il mesurer ? Quand ? Où ?
  - Comment définir progressivement des modèles de trafic ?
  - Comment évaluer l'apport par rapport au CA sans mesure ?

3. Contrôle d'admission

Exemple de contrôle d'admission basé sur les mesures (1/4)

- **Notations**
  - Chaque source est modélisée par un seau à jetons (ρ, δ). Ainsi, le total du trafic généré par la source, pendant U unités de temps, ne peut excéder $\rho U + \delta$ et la source ne peut transmettre que si elle a des jetons.
  - C : capacité du lien
  - v : ratio d'utilisation du lien fixé à l'avance ($v \leq 1$). Ainsi, la bande passante maximale utilisée est vC.
  - $\hat{D}$: pire cas du délai de transfert estimé
  - $\hat{R}$: estimation (en bits) du flux agrégé sur le lien
  - N : nombre de flux déjà acceptés

Exemple de contrôle d'admission basé sur les mesures (2/4)

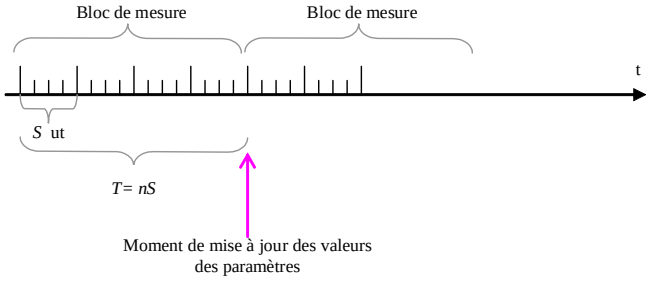
- **Test de CA**
  - **Condition sur le délai**
    - Soient (ρ_{N+1}, δ_{N+1}) les paramètres du seau à jetons du nouveau flux *N+1* et $Dmax_{N+1}$, le délai de transfert maxi exigé par ce flux.
    - Le pire temps d'attente pour un paquet du flux *N+1* est obtenu en supposant que tous les flux transmettent simultanément un paquet de taille maxi égale à leur δ_i :

$$D = \frac{\sum_{i=1}^{N} \delta_i}{\mu}$$

    - Le pire temps de transfert estimé, $\hat{D}$, est utilisé à la place de *D* (*D* est plus pessimiste que $\hat{D}$)
    - Le test de CA obtenu est : $D\max_{N+1} > \hat{D} + \frac{\delta_{N+1}}{\mu}$
  - **Condition sur la bande passante :** $\nu C > \hat{r} + \rho_{N+1}$

Exemple de contrôle d'admission basé sur les mesures (3/4)

- **Processus de mesure**
  - Le délai de transfert et débit du flux agrégé sont mesurés périodiquement et sont adoptés comme valeurs pour les paramètres $\hat{D}$ et $\hat{r}$
  - Pour simplifier, on considère que tous les paquets ont la même taille et que leur temps de transmission est égal à 1.



Exemple de contrôle d'admission basé sur les mesures (4/4)

■ **Processus de mesure**

– Un échantillon de mesure du délai est obtenu pour chaque transmission de paquet.

– Chaque échantillon de mesure du débit du flux agrégé est obtenu sur une période S.

– Chaque bloc de mesure dure T unités de temps ($T = nS$).

– A la fin de chaque bloc de mesure, l'échantillon dont la valeur est la plus élevée est adopté pour estimer $\hat{D}$ et $\hat{r}$.

– Les paramètres $\hat{D}$ et $\hat{r}$ sont mis à jour immédiatement (i.e. avant la fin du bloc) :

* Si un nouveau flux k est accepté, la mise à jour se fait ainsi : $\hat{D} = \hat{D} + \frac{\delta_k}{\mu}$ et $\hat{r} = \hat{r} + \rho_k$

* Quand une mesure dans le bloc actuel est plus élevée que celle déjà estimée alors, les paramètres sont mis à jour immédiatement.

4. Conclusion

Problèmes ouverts

■ **Sur les modèles**

– Modèles statistiques efficaces
– Combinaison de modèles pour l'agrégation de flux
– Compromis : Complexité/Précision/Surdimensionnement

■ **Sur les CA**

– CA efficaces utilisables en ligne
– Caractérisation approximative des flux et complexité du CA
– Compromis entre complexité et performance
– CA adapté aux réseaux sans fil

■ **Contrôle d'admission en cas de AS interconnectés (inter-domaines)**

– Chaque système autonome (AS) peut avoir son CA
– Comment avoir une décision de CA de bout en bout optimale ?
– Comment utiliser le CA basé sur les mesures avec des CA locaux hétérogènes ?

Chapitre 4

Routage

Principes et algorithmes

Chapitre 4

Routage

Principes et algorithmes

1. Principes généraux du routage à QoS

Définition du routage à QoS (*QoS-based routing*)

- "Un mécanisme de routage dans lequel les chemins que doivent emprunter les flux sont déterminés en tenant compte à la fois des connaissances sur la disponibilité des ressources du réseau et des exigences de QoS des flux." [RFC 2386]

- "Un protocole dynamique de routage qui étend sa sélection de chemin à des critères pour inclure des paramètres de QoS tels que la bande passante disponible, l'utilisation des liens, les ressources de calcul des nœuds, le délai et la gigue." [QoS Forum]

Fonctions

- Collecter des informations sur l'état du réseau (fonction vitale et complexe)
- Trouver le meilleur chemin pour un nouveau flux en fonction de la QoS requise
- Changement de chemin avec dégradation progressive de la QoS
- Optimiser l'utilisation des ressources et d'autres critères

1. Principes généraux du routage à QoS

Sélection de chemin

- A la demande (pour chaque paquet, pour chaque flux, …)
  Protocoles réactifs

- Périodiquement et stockage dans une table
  Protocoles proactifs

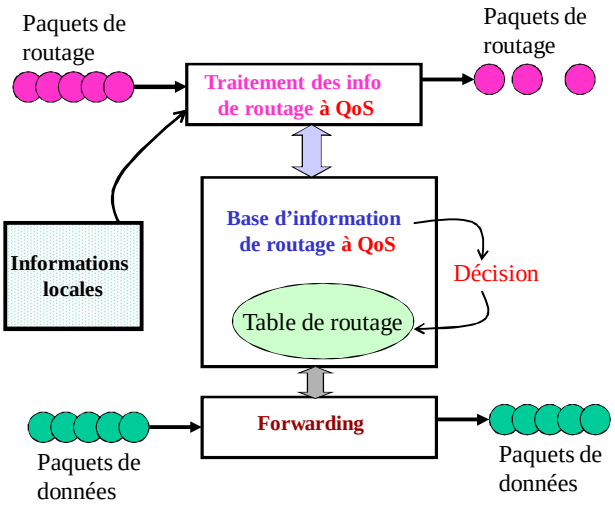
- Pre-calcul basé sur les algorithmes de Bellman-Ford et Dijkstra
  Protocoles hybrides
  (efficace pour les réseaux de grande taille notamment)

Collecte et dissémination d'informations

- A la demande ou périodique (très simples mais pouvant être coûteux)
- A chaque fois que quelque chose change ('idéal', très précis, mais très coûteux)
- Politiques à base de seuil
  - Déclencher la mise à jour si : `| NouvelleValeur – AncienneValeur | > seuil`
  - \+ Evite les M.à.j inutiles
  - \- Réaction lente pour les changements d'état en de nombreux points
- Politiques à base de classe
  - Les capacités des liens sont subdivisées en classes
  - M.à.j déclenchées quand les limites des classes sont atteintes
  - \+ Les tailles des classes permettent d'ajuster/régler le rapport Précision/charge
  - \- Les tailles des classes affectent la sélection des chemins
- Difficulté majeure : optimiser le rapport "Précision des infos d'état/coût"

Composants du routage

- **Protocole de routage**
- **Base d'information de routage**
- **Algorithme de routage**



Classes d'algorithmes de routage (1/4)

➔ **Selon le nombre de participants**

- Unicast
- Multicast
- Anycast

➔ **Selon la manière dont le chemin est calculé**

- Routage par la source
- Routage distribué (hop-by-hop)
- Routage hiérarchique

Classes d'algorithmes de routage (2/4)

➔ **Routage par la source**

– Chaque routeur a une vue locale du réseau (mise à jour périodique ou non)
– Sélection du chemin par la source et notification de ce chemin aux autres nœuds

\+ Simple
\+ plus efficace pour la gestion de QoS temporelle
\- Connaissance approximative
\- Peu efficace pour les réseaux de grande taille

➔ **Routage distribué (hop by hop)**

– Sélection du prochain nœud seulement
– Informations d'état échangées avec les voisins
\+ Plus flexible
\- Difficulté de partage et d'échange d'informations d'état

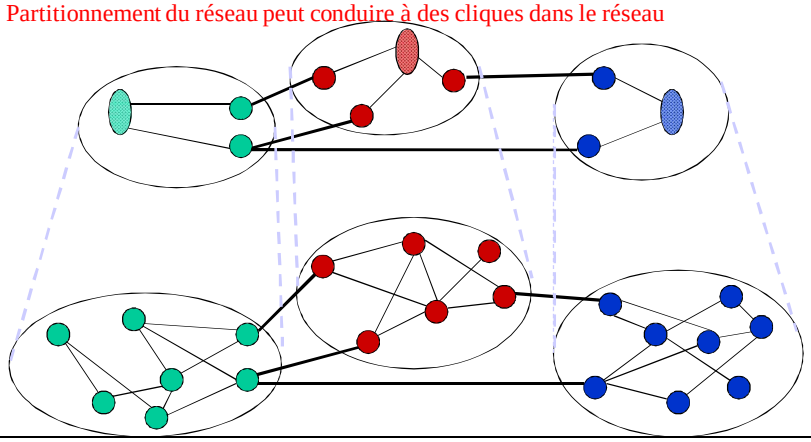
Classes d'algorithmes de routage (3/4)

➔ **Routage hiérarchique**

– Hiérarchisation des nœuds (agrégation)

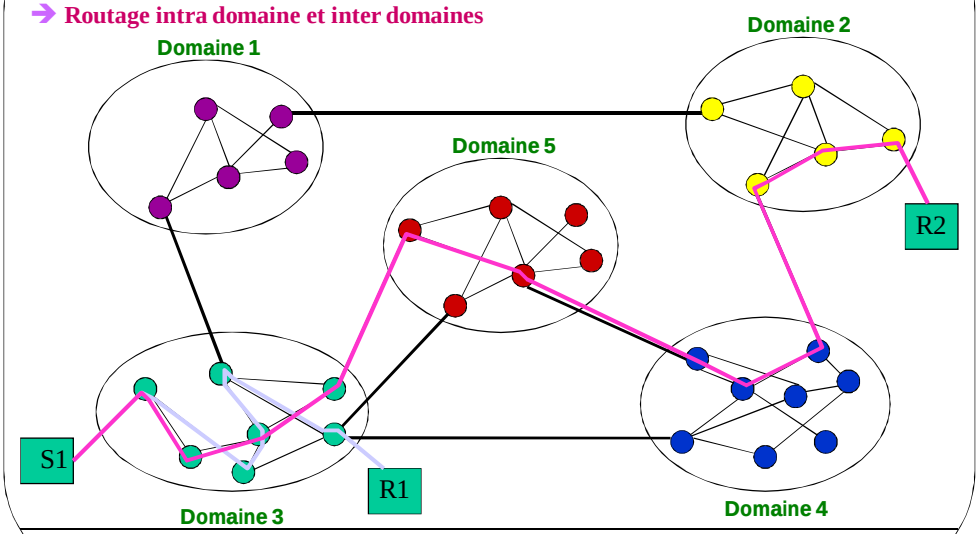
\+ Réduction de la complexité de gestion des états

\- Partitionnement du réseau peut conduire à des cliques dans le réseau



Classes d'algorithmes de routage (4/4)

➔ **Routage intra domaine et inter domaines**



1. Principes généraux du routage à QoS

Algorithmes de routage à QoS

➔ **Des dizaines d'algorithmes de routage sont proposés**
Critères de classement

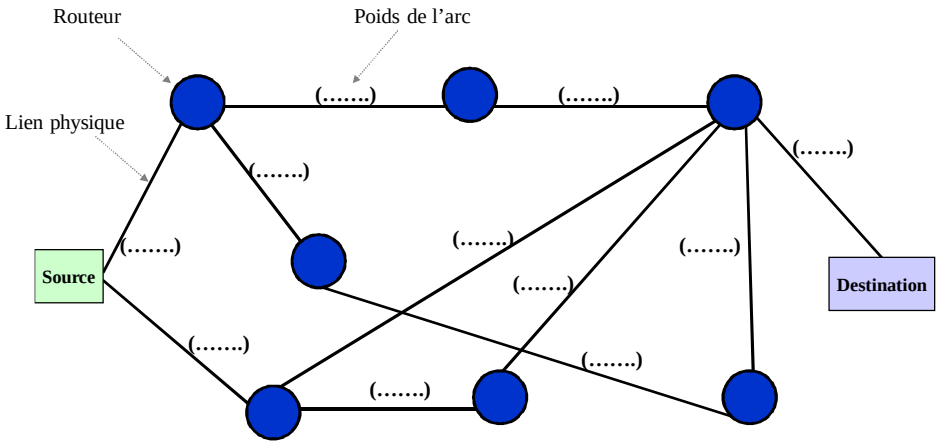
- Contraintes prises en compte (délai, gigue, bande passante, …)
- Stratégie du routage (par la source, distribué, hiérarchique)
- Complexité de l'algorithme
- Complexité de la communication pour maintenir les informations d'état

➔ **Propriétés**

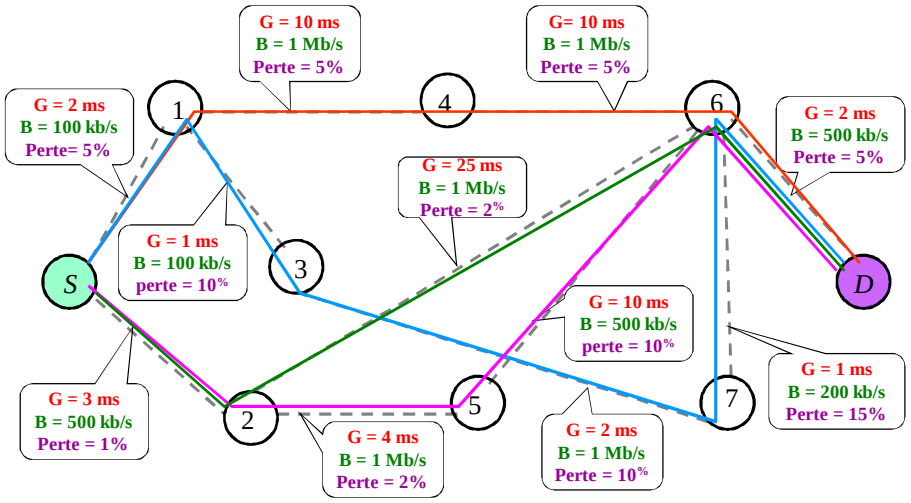
- Complexité (traitement et messages) faible
- Passage à l'échelle
- Coexistence de routage à QoS avec routage best effort

2. Formalisation des problèmes de routage à QoS

Réseau représenté par un graphe G = (V, E)



Exemple de chemins à QoS (1/2)



Exemple de chemins à QoS (2/2)

Chemin	Nbre de sauts	Gigue e2e	BP e2e	Taux de perte e2e
S → 1 → 4 → 6 → D	4	24 ms	100 kb/s	18,5%
S → 1 → 3 → 7 → 6 → D	5	**8 ms**	100 kb/s	38%
S → 2 → 5 → 6 → D	4	19 ms	**500 kb/s**	17%
S → 2 → 6 → D	**3**	30 ms	**500 kb/s**	**7,8%**

Problème du chemin (le plus court) avec contraintes de QoS

➔ **Enoncé du problème dans le cas unicast**

Etant donné : une source *s*, une destination *d*, les vecteurs-poids associés aux arcs, un vecteur de besoins de QoS *B*, trouver un chemin *p* de *s* à *d* tel que :

– (1) Contraintes : $W(p) \{ \leq | \geq \} B$
/* *W(p)* désigne le poids du chemin *p* en terme de délai et … */

– (2) Optimisations :

$Cout(p) \leq Cout(p') \ \forall\ p'$ respectant (1)

$NombreSauts(p) \leq NombreSauts(p'') \ \forall\ p''$ respectant (1)

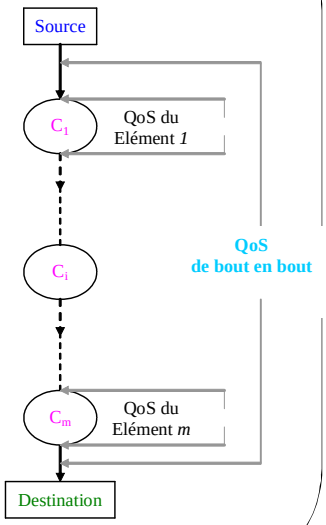
Importance de comprendre les types de métriques de QoS

■ **Métrique additive**
– $QoS(C_1 ; C_2) = QoS(C_1) + QoS(C_2)$
– **ex. Délai**

■ **Métrique multiplicative**
– $QoS(C_1 ; C_2) = QoS(C_1) * QoS(C_2)$
– **ex. Disponibilité**

■ **Métrique concave**
– $QoS(C_1 ; C_2) = \min\{QoS(C_1), QoS(C_2)\}$
– **ex. Débit**

■ **Autres**
– **Cas des spécifications non homogènes**



Problèmes de routage à QoS

➔ Problèmes de satisfaction de contraintes

$$w_i(P) \overset{def}{=} \underset{(u \to v) \in P}{\Omega} w_i(u,v) \ \pi \ L_i \qquad i = 1,...,\#QoSMetrics$$

➔ Problèmes d'optimisation

$$Cost_k(P) \ \pi \ Cost_k(P') \qquad k = 1,..., \#\ OptimizeCriteria$$

➔ Problèmes de satisfaction de contraintes et d'optimisation

Problèmes de routage à QoS à 1 métrique

$$Delay(P) \overset{def}{=} \sum_{(u \to v) \in P} Delay(u,v) \le D\_required$$

$$BWD(P) \overset{def}{=} \min_{(u \to v) \in P} \left(bwd(u,v)\right) \ge B\_required$$

$$Dispo(P) \overset{def}{=} \prod_{(u \to v) \in P} \left(dispo(u,v)\right) \ge Dispo\_required$$

➔ Problèmes résolus

➔ Algorithmes de Dijikstra Shortest-path first de Dijkstra : $O(n^2)$ n : nombre de nœuds

et Bellman-Ford $O(nm)$ m : nombre de liens

Problèmes de routage à QoS à *m* métriques

$$Delay(P) \overset{def}{=} \sum_{(u \to v) \in P} Delay(u,v) \leq D\_required$$

$$\wedge \quad BWD(P) \overset{def}{=} \min_{(u \to v) \in P} \left(bwd(u,v) \right) \geq B\_required$$

$$\wedge \quad \#hops(P) \leq \#hops(P') \qquad \forall P' \in SetFeasiblePaths$$

$$\wedge \quad \ldots$$

➔ **Problèmes NP-complets pour tout vecteur de QoS avec *n* (*n* ≥ 2) métriques**
(délai, débit), (délai, erreur), (délai, débit, gigue), …. [Wang et Crowcroft 96]

➔ Recherche de fonctions de coûts, Heuristiques

Principes de résolution (1/2)

➔ **Prise en compte contrainte par contrainte**

- PS_1 = Sélection des chemins respectant QoS_1
- PS_2 = Sélection parmi l'ensemble PS_1 des chemins respectant QoS_2
- ….
- PS_m = Sélection parmi PS_{m-1} des chemins respectant QoS_m

➔ **Prise en compte d'une contrainte et optimisation de critère(s)**

- Sélection de chemins respectant la contrainte QoS_x
- Optimisation d'un critère simple ou composé

Principes de résolution (2/2)

➔ **Utilisation de métrique composée**

$$ComposedQoS(P) = \left(1 - \frac{Delay(P)}{DelayMax}\right) * \left(1 - \frac{Cost(P)}{CostMax}\right)$$

$$ComposedQoS(P) = \frac{Bwd(P)}{Delay(P) * Loss(P)}$$

$$ComposedQoS(P) = \frac{Delay(P)}{1 - \frac{Cost(P)}{CostMax}}$$

➔ **Autres heuristiques**

3. Eléments sur les coûts du routage à QoS

Coûts du routage

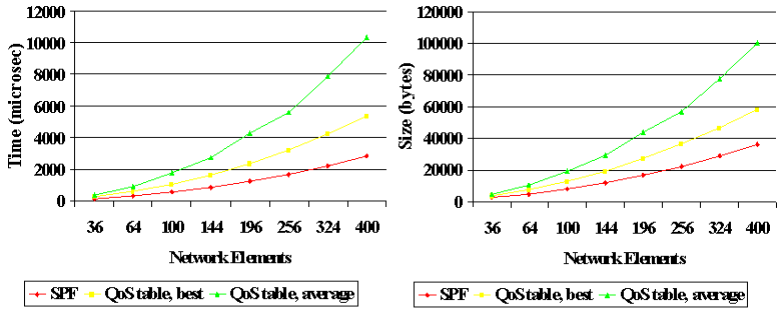
- Traitement/calcul
  - Calcul des chemins (souvent NP-complet)
  - Calcul lié aux échanges d'état
- Stockage
  - Informations de topologie du réseau
  - Informations d'état (sur différentes métriques)
  - Table de routage courante, tables de routage pré-calculées
- Bande passante (paquets liés au routage)

Facteurs de coût du routage

- Fréquence de sélection des chemins
- Métriques
- Facteurs de complexité (nombre de noeuds/liens, entrées de la table de routage,…)
- Compromis précision/surcoût

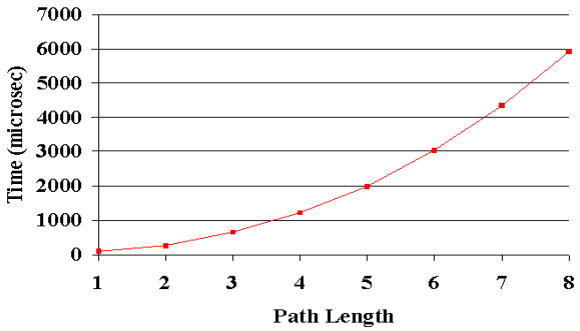
3. Eléments sur les coûts du routage à QoS

Quelques éléments sur le coût de la table de routage [Guérin 1998]

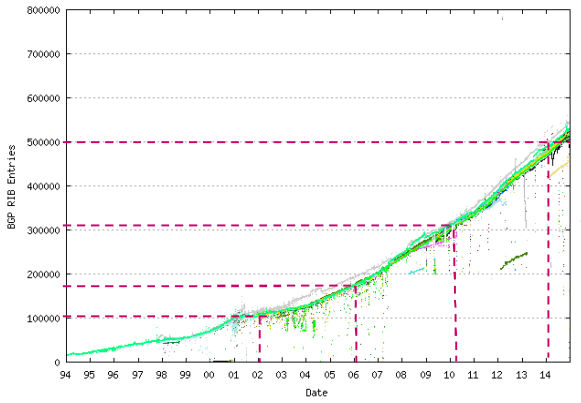


3. Eléments sur les coûts du routage à QoS

Quelques éléments sur le coût de calcul de chemin à la demande [Guérin 1998]



Evolution de la table de routage de BGP depuis 1994
(http://bgp.potaroo.net)



'Double' tous les 4 ans

4. Exemples d'algorithmes de routage à QoS

Rappels sur les protocoles de routage classique (1/2)

➔ **Routage à vecteur de distance (*Distance vector routing*)**

- Porte aussi le nom de Algorithme de Bellman-Ford.
- Chaque routeur transmet (périodiquement) les infos qu'il connaît uniquement à ses voisins immédiats. L'info transmise par chaque routeur est un vecteur qui contient, pour chaque destination, la distance entre ce nœud et cette destination.
- Chaque routeur utilise les vecteurs-infos qu'il reçoit pour construire sa table de routage.
- Comme chaque nœud n'a qu'une partie de la vue de la topologie, le risque d'avoir des boucles (lors de la recherche de chemin) est élevé.
- Pour éviter les boucles, différentes techniques sont possibles, par exemple Limiter le nombre de sauts à 16.

Rappels sur les protocoles de routage classique (2/2)

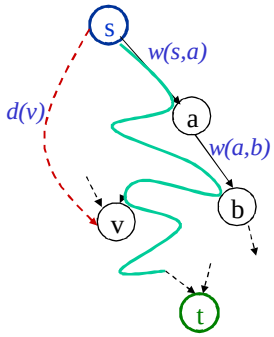
➔ **Routage à état de lien (*Link state routing*)**

- Chaque nœud doit avoir une connaissance globale de la topologie du réseau.
- Les nœuds s'échangent entre eux les différentes métriques de chaque lien de la topologie.
- Chaque nœud doit :
  - Découvrir ses voisins (en envoyant des messages Hello)
  - Mesurer le temps d'acheminement vers chacun de ces voisins (en envoyant des messages Echo)
  - Construire un paquet spécial disant tout ce qu'il vient d'apprendre
  - Diffuser (de manière périodique ou en cas de changement de la topologie) ce paquet spécial à tous les autres routeurs du réseau.
- Chaque routeur utilise l'algorithme de Dijkstra (ou un autre) pour construire sa table de routage contenant le meilleur chemin pour chaque destination.
  Généralement, 'Meilleur' désigne le plus court.

Exemple 1 : Extension de Dijkstra's Shortest path algorithm (Cas Unicast)

➔ **Définitions**

G : ensemble des nœuds du réseau
E : ensemble des arcs du réseau
w(x,y) : poids associé au lien du nœud *x* vers le nœud *y*
d(v) : poids du chemin depuis la source *s* jusqu'au nœud *v*
s : nœud source
t : nœud destination
Pred(u) : prédécesseur du nœud *u* sur le chemin
Adj(u) : ensemble des nœuds adjacents du nœud *u*.



4. Exemples d'algorithmes de routage à QoS

➔ **Algorithme One-QoS_Dijkstra(V,E, w, s, t)**

Step 1 : /* Initialisation */

For each node $v \in V$ **do** $d[v] \leftarrow \infty$; $Pred[v] \leftarrow$ NIL **od**

$d[s] \leftarrow 0$

$Q \leftarrow V$

Step 2 : /* Détermination du chemin le plus court */

while $Q \neq \varnothing$

{ do $u \leftarrow$ Extract-min(Q) /* u tq : d[u] = min{d[y], $\forall y \in Q$} */

if u = t **then exit**

for each $v \in Adj[u]$

{ if $w(v, u) \oplus d[u]$ π $d[v]$

then $Pred[v] \leftarrow u$

$d[v] \leftarrow w(v, u) \oplus d[u]$

}

}

Fonction Poids

QoS = délai ⇒ ($\oplus$, π) = (+, <)

QoS = Perte ⇒ ($\oplus$, π) = (*, <)

QoS = Disponibilité ⇒ ($\oplus$, π) = (*, >)

4. Exemples d'algorithmes de routage à QoS

Exemple 2 : EBSP

(Enhanced Bandwidth Shortest Path) J. Wang et K. Nahrstedt 2002

➔ **Objectif**

- Sélection du chemin ayant la bande passante la plus élevée en minimisant le nombre de sauts
- C'est un algorithme saut par saut.
- Il est utilisable dans des contextes où on cherche à fournir la bande passante la plus élevée possible à certains flux (par exemple, flux premium de DiffServ).

➔ **Fonction Poids de EBSP**

$$d(P) = \sum_{i=1}^{n-1} \frac{2^{i-1}}{BP(i, i+1)}$$

2^{i-1} ⟵ **Facteur qui pénalise les chemins avec plus de noeuds**

$BP(i, i+1)$ ⟵ **Facteur qui pénalise les liens avec une faible BP**

n : nombre de nœuds du chemin P

BP(i,i+1) : bande passante du lien (i, i+1)

Exemple 2 : EBSP (suite)

➔ **Algorithme EBSP(V,E, BP, s, t)**

Step 1 : /* Initialisation */

For each node $v \in V$ **do** $d[v] \leftarrow \infty$; $Pred[v] \leftarrow$ NIL **od**

$d[s] \leftarrow 0$

$Q \leftarrow V$

Step 2 : /* Détermination du chemin le plus court */

while $Q \neq \varnothing$

{ do $u \leftarrow$ Extract-min(Q) /* u tq : $d[u] = \min\{d[y], \forall y \in Q\}$ */

if u = t **then exit**

for each $v \in Adj[u]$

{ if $2d[u] + 1/BP[v,u] < d[v]$

then $Pred[v] \leftarrow u$

$d[v] \leftarrow 2d[u] + 1/BP[v,u]$

}

}

Exemple 3 : DCCR

(Delay-Cost Constrained Routing) L. Guo et I. Matta 2001

➔ **Objectif**

- Sélection du chemin permettant de respecter une borne de délai de bout en bout Δ_{e2e} et un coût max C_{e2e} à ne pas dépasser .
- C'est un algorithme saut par saut.

➔ **Fonction Poids de DCCR**

$$d(P^u) = \begin{cases} D(P^u) & \text{si } D(P^u) \le \Delta_{e2e} \text{ et } C(P^u) \le C_{e2e} \\ 1 - \dfrac{C(P^u)}{C_{e2e}} & \\ \infty & \text{autrement} \end{cases}$$

Si le chemin ne respecte pas la contrainte de délai ou de coût, il est éliminé

Permet de privilégier le chemin ayant le coût le plus faible

D(P) : délai cumulé du chemin P
C(P) : coût du chemin P
P^u chemin allant de la source au nœud u

5. Protocoles de routage

➔ **Objectif :** Définir les formats des messages véhiculant les infos de routage et les règles d'échange de ces messages.

➔ **Protocoles actuels dans Internet : Best effort**

- RIP (Routing Information Protocol) – intra-domaine ; distance vector
- OSFP (Open Shortest Path First) – intra-domaine ; link-state
- IGP (Interior Gateway Protocol) – intra-domaine ; link-state
- IS-IS (Intermediate System - IS – intra-domaine ; link-state
- EGP (Exterior Gateway Protocol) – inter-domaine
- BGP-4 (Border Gateway Protocol v.4) – inter-domaine ; distance vector

➔ **Protocoles pour la QoS**

- QOSFP : extension de OSPF pour tenir compte de la QoS

6. Conclusion

➔ Sans routage à QoS : difficile ou impossible de déployer des applications exigeantes en termes de QoS.

➔ De nombreuses solutions ont été proposées pour avoir du routage à QoS.

➔ Quelques leçons apprises en utilisant le routage à QoS dans les réseaux IP

– Le routage à QoS devrait être utilisé pour des flux et non pour des paquets individuels.

– La gigue est très difficile à garantir dans un réseau IP.

– Il est préférable d'utiliser un routage par la source car elle est la mieux placée pour sélectionner un chemin qui répond à la QoS demandée. Les routeurs intermédiaires font un contrôle d'admission mais ne prennent pas de décision concernant le routage. S'ils rejettent le flux, la source devra tenter un autre chemin.

6. Conclusion

➔ **Des problèmes à résoudre**

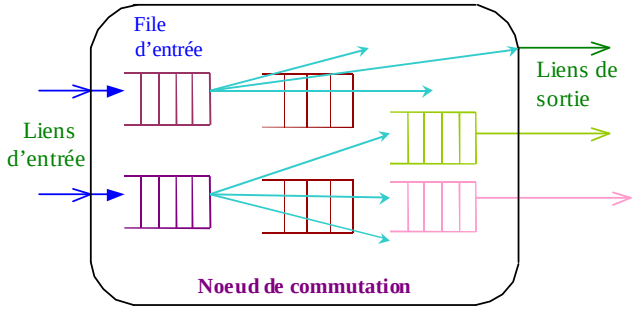
- Métriques à considérer (sensibles à la QoS)
- Précision de la topologie et de la charge du réseau
- Minimisation du surcoût (compromis : efficacité/coût)
  - Echange d'informations d'état (fréquence adéquate)
  - Traitement (pré-calcul de chemins)
  - Table de routage (hiérarchique)
- Maîtrise et prédiction des congestions
- Complexité et efficacité du protocole de routage (QOSPF…)
- Lien routage – réservation de ressources
- Routage inter domaines
  - Représentation commune des états, Politique d'allocation des ressources
  - Interopérabilité des protocoles/algorithmes de routage
- Routage dans les réseaux sans fil
- Intelligence/adaptabilité du routage

Chapitre 5

Ordonnancement de paquets

1. Introduction

Approches de gestion de files d'attente (1/2)



➔ **Attente dans les files d'entrée**

➔ **Attente dans les files de sortie**

➔ **Autres techniques**

1. Introduction

Approches de gestion de files d'attente (2/2)

➔ **Attente dans les files de sortie**
Tout paquet est placé dans sa file de sortie dès son arrivée.
Avantage : c'est la plus performante au niveau débit

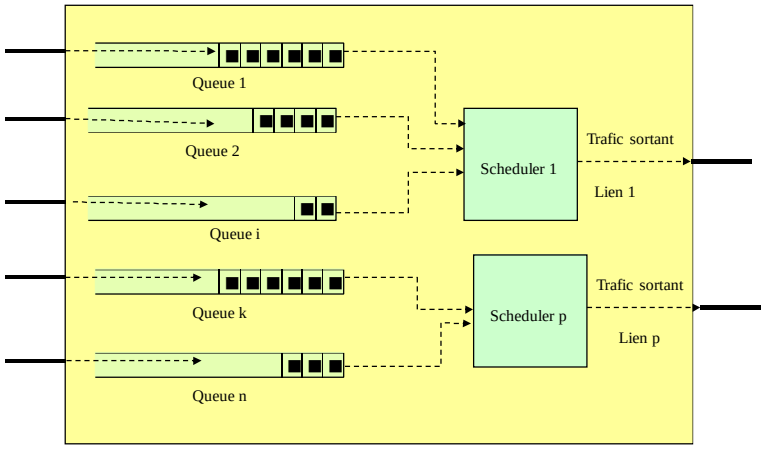
➔ **Attente dans les files d'entrée**
Les paquets arrivant sur un port d'entrée sont placés dans une file d'attente associée à ce port et servis en FIFO.
Inconvénient : « Head-of-line blocking » (lorsque le premier paquet de la file est bloqué, car son lien de sortie est occupé, tous les autres paquets sont bloqués, même si leur lien de sortie sont libres).

➔ Attente dans les files de sortie virtuelles (évite le « Head-of-line blocking »)
A chaque port d'entrée sont associées autant de files d'attente que de liens de sortie utilisés par les paquets arrivant sur ce port. Tout paquet attend dans sa file de sortie virtuelle, avant d'être servi.

➔ **Attente dans une file unique**
Tous les paquets arrivant au routeur sont placés dans une file d'attente unique avant d'être servis. C'est la plus simple, mais la moins efficace pour la garantie de QoS.

➔ **Combinaison d'attente dans les files d'entrée et de sortie**

Restriction : files de sortie



- **Algorithme d'ordonnancement de paquets = Discipline de service**
  (terminologie des files d'attente)

Propriétés

- Facilité d'implantation et surcoût faible
  - Traitement pour chaque paquet ➔ nécessité de traitement très rapide
  - Complexité idéale *O(1)* - Complexité *O(Nombre_paquets)* à éviter
- Garantie (meilleur effort, statistique, déterministe) vérifiable : délai, perte…
- Equité
  - Répartition équitable des ressources entre les flux
  - L'équité conduit au *best effort*, mais pas à la garantie de bornes
- Isolation des flux
  - Un flux qui fonctionne mal ne doit pas perturber les autres.
- Contrôle d'admission
  - Simple à implanter
  - Efficace (pour une meilleure admission et utilisation des ressources)
- Passage à l'échelle (Scalability)

Propriétés contradictoires

Classification des algorithmes

- Garantie déterministe
  - Garantie non déterministe (Probabiliste, meilleur effort)
- Garantie d'un seul paramètre de QoS (Délai, débit…)
  - Garantie de plusieurs paramètres de QoS
- Stratégies fondées sur des priorités fixes
  - Stratégies Round Robin
- Stratégies sans oisiveté (work-conserving)
  - Stratégies avec oisiveté (non-work-conserving)

➔Chaque algorithme de base garantit un type de contraintes (débit, délai, taux de perte) : **Combiner les algorithmes de base**

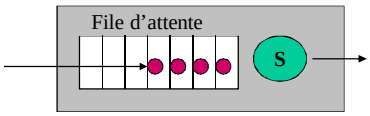
Loi de conservation de Kleinrock (1971) (1/2)

Formule de Little (1956) : $E(t) = E(n) / \lambda$

$E(t)$: moyenne de temps de réponse

$E(N)$: moyenne du nombre de clients dans la file

λ : taux d'arrivée des clients



La loi de conservation de Kleinrock stipule que :

Si l'ordonnanceur est conservatif alors, quelque soit la discipline choisie :

$$\sum_{k=1}^{N} r_k d_k q_k = constante$$

$r_k d_k q_k$ peut être considéré comme un délai pondéré pour la connexion k.

N connexions géré par un ordonnanceur.

r_k le débit moyen de la connexion k.

d_k le délai moyen de traitement par paquet de la connexion k.

q_k le délai moyen de séjour en file d'attente par paquet de la connexion k.

Loi de conservation (2/2)

- Cette loi signifie que, pour tout ordonnanceur conservatif, la somme des délais pondérés est constante. Donc si on offre à une connexion un délai plus court, on le fait au détriment des autres connexions qui vont avoir un délai plus élevé.
- Tout ordonnanceur non conservatif ne peut que conduire à une somme de poids pondérés supérieure à celle d'un ordonnanceur conservatif (à cause des temps d'oisiveté).
- FIFO est la discipline conservative la plus simple. Donc la somme des délais pondérés de FIFO peut être considérée comme une borne inférieure pour toutes les disciplines de service.

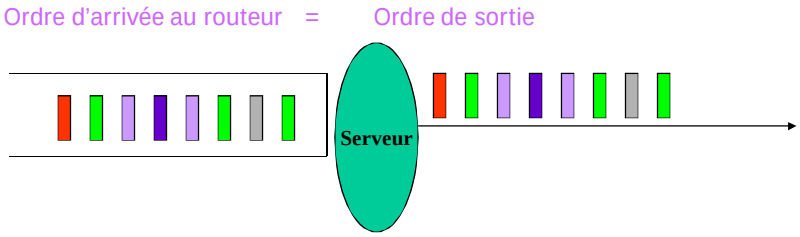
Disciplines de service

- **Conservatives**
  - **FP (Fixed Priority)**
  - **FQ (Fair Queueing)**
  - **WFQ (Weighted Fair Queueing)**
  - **WRR (Weighted Round Robin)**
  - **SCFQ (Self-Clocked Fair Queueing)**
  - **Virtual CLocks**
  - **Delay EDD (Delay Earliest Due Date)**
  - **Autres**
- **Non conservatives**
  - **Jitter EDD**
  - **Stop-and-Go**
  - **HRR (Hierarchical Round Robin)**
  - **RCSP (Rate Controlled Static Priority)**
  - **Autres**

2. Ordonnancement FIFO et FP

FIFO (First in First Out) – FCFS (First Come First Serve)

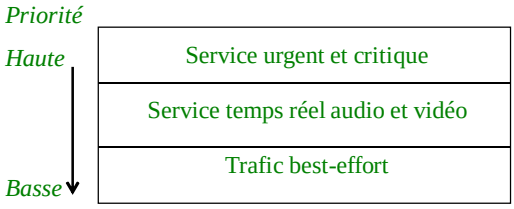
- Naturelle (la première qui vient à l'esprit)
- Non équitable
- Ne permet pas la garantie de QoS (en général)



2. Ordonnancement FIFO et FP

FP (Fixed Priority) (1/2)

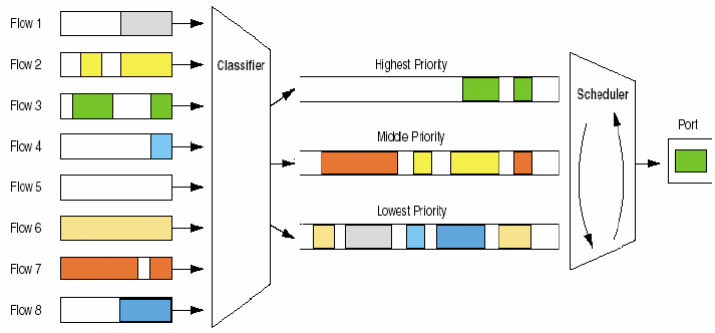
- FP (Fixed Priority) = PQ (Priority Queueing)
- Une priorité fixe est associée à chaque flux (connexion) ou à chaque paquet



- Il y a un mapping entre les priorités initiales et les priorités de l'ordonnanceur.
- Si le nombre de priorités de l'ordonnanceur est faible, cela peut conduire à un service non-conforme aux priorités initiales.

FP (Fixed Priority) (2/2)

- Les paquets de priorité élevée sont servis d'abord

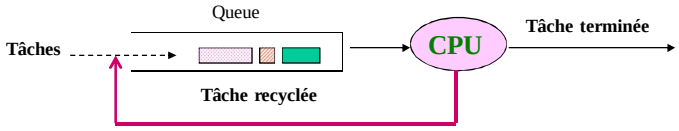


- Risque de famine pour les paquets de priorités faibles

3. Ordonnancement Round Robin

Round Robin (RR) de base pour les tâches

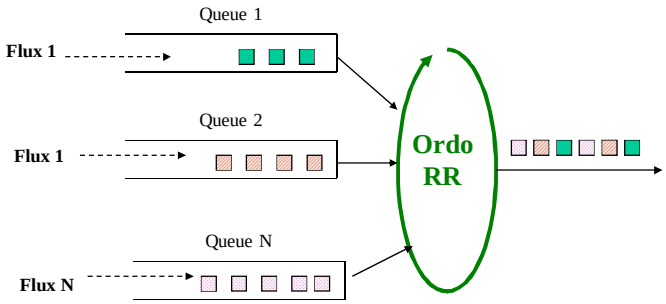
- Une seule queue pour toutes les tâches (processus).
- Servir pendant Δt chaque tâche. Si la tâche n'a pas fini la recycler en queue.
- Ordonnancement largement utilisé dans les systèmes non temps réel.



3. Ordonnancement Round Robin

Round Robin (RR) pour l'ordonnancement de paquets

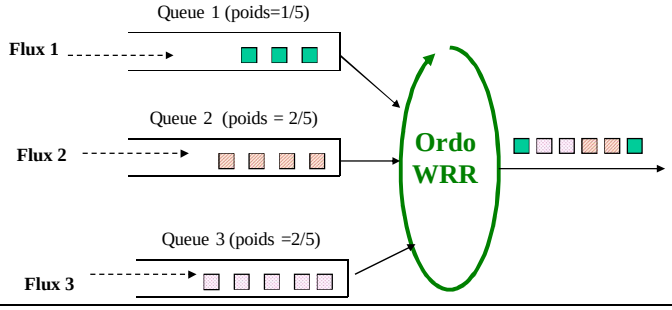
- Associer une queue à chaque flux. Servir les queues à tour de rôle.
- Avantages : simplicité, possibilité de réalisation câblée, équité.
- Inconvénients : ne permet pas la garantie de QoS. Pas d'équité si les paquets sont de tailles différentes.



3. Ordonnancement Round Robin

Weighted Round Robin (WRR) : principe de base

- Associer une queue à chaque flux. Associer à chaque flux un poids normalisé en fonction de la taille moyenne de paquet du flux.
- Servir les queues (non vides) à tour de rôle et en fonction de leurs poids.
- Avantages : prise en compte de l'importance (poids) de chaque flux. Protection des flux les uns contre les autres.
- Inconvénients : pénalise les flux à faibles poids.



WRR pour flux périodiques (1/2)

➔ **Chaque connexion est définie par (P_i, D_i, e_i)**

P_i : intervalle minimum d'arrivée de message sur la connexion i,
D_i : délai de bout en bout et e_i : nombre de paquet par message

➔ **Principe**

- **L'ordonnanceur fonctionne de manière cyclique et chaque tour est défini par un nombre de slots maximum, RL. La longueur d'un slot est égale à 1 et correspond à la durée de transmission du paquet le plus long. A chaque tour, les connexions sont servies à tour de rôle.**

- **Durant la phase d'établissement de connexion, l'ordonnanceur de chaque routeur affecte à chaque connexion un poids wt_i (en fonction de P_i, D_i et e_i contenus dans la demande de connexion) qui indique le nombre de slots affectés à cette connexion, à chaque tour. Si la demande peut être satisfaite, les slots sont réservés, sinon aucun slot n'est réservé et les routeurs ayant déjà réservé des slots pour cette connexion sont avertis pour annuler leur réservation.**

WRR pour flux périodiques (2/2)

- **Trois conditions à respecter pour garantir les contraintes temporelles**

$$\sum_{i=1}^{n} wt_i \le RL \qquad RL \le \min_{i=1,\ldots,n}(P_i) \qquad \forall i\,(1 \le i \le n) \quad wt_i \ge \left\lceil \frac{e_i}{\lfloor P_i / RL \rfloor} \right\rceil$$

➔ **Observations**

- **Avec la discipline WRR, le délai de bout en bout d'un message de la connexion i traversant m routeurs ayant tous la même valeur pour RL est W_i** (en supposant négligeable délai de propagation) :

$$W_i \le \left(\left\lceil e_i / wt_i \right\rceil + m - 1 \right) \times RL \le P_i + (m-1) \times RL \qquad W_i < D_i$$

- **Le contrôle de gigue est difficile à réaliser.**

3. Ordonnancement Round Robin

Problèmes posés par l'utilisation de WRR

- Avec des paquets de tailles et des poids différents, on a besoin de connaître la taille moyenne de paquets à l'avance. Cette taille moyenne est parfois difficile à connaître a priori (ce qui rend WRR non équitable)
- Si la différence entre les tailles (min et max) de paquets ou entre les poids (min et max) est importante, la durée d'un tour peut être élevée, conduisant à de longues périodes de non équité.
  - Eg. On considère un lien à 45 Mb/s utilisé par 500 connexions ayant des paquets de taille fixe égale à 500 octets. 250 connexions ont un poids de 1 et 250, un poids de 10.
    - Chaque paquet dure 500 * 8/45 Mb/s = 88.8 microsecondes
    - Durée d'un tour = (250*1+250*10) * 88.8 = 244.2 ms
    - Cette durée ne permet pas d'avoir des flux audio ou vidéo.
- Conséquence : WRR est une discipline efficace pour des paquets de petites tailles avec des durées de tour petites (c'est le cas d'ATM par exemple).

3. Ordonnancement Round Robin

Deficit Round Robin (DRR) (1/2)

- Idée de base : extension de RR pour des paquets de taille variable.
  Economiser des crédits pour transmettre.
- Principe

```
• Associer un compteur C[k], initialisé à 0, à chaque queue k
• Lorsque la connexion k est visitée par DRR
  ▪ Si la queue k est non vide
        { C[k] = C[k] + quantum ;
        Si Taille(tetequeue[k]) <= C[k]
                { Le paquet est transmis;
                C[k] = C[k] – taille du paquet transmis;
                Si la queue k est vide { C[k] = 0; }
        }
  ▪ Passer à la queue suivante
```

Le quantum est choisi pour permettre la transmission de paquet de taille minimale

3. Ordonnancement Round Robin

Deficit Round Robin (2/2)

- Avantages : facile à implanter ; plus d'équité que RR
- Inconvénients : ne permet pas la garantie de QoS (en général).

➔ Il existe d'autres formes de stratégies RR

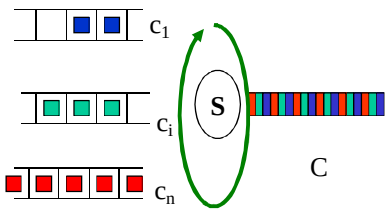
– HRR (Hierarchical RR)

– BWRR (Budgeted WRR)

4. Ordonnancement PGPS et WFQ

PS « Processor Sharing »

- Temps partagé simple du processeur (pour l'ordonnancement de tâches)



$$\forall i, j \quad c_i = c_j = \frac{C}{n}$$

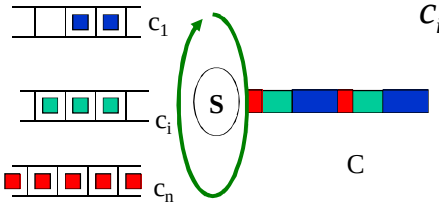
c_k : vitesse d'exécution de la tâche k

C : vitesse d'exécution du processeur

- PS n'est pas implantable pour les paquets (sinon on risque de transmettre des paquets contenant moins d'un bit).

GPS « Generalized Processor Sharing »

– PS + équité en tenant compte de l'allocation préalable des tâches (poids ϕ_i)



$$c_i = \frac{\phi_i}{\sum_j \phi_j} C \qquad \sum_j \phi_j \leq 1$$

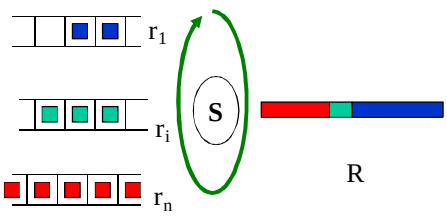
– GPS garantit un temps d'exécution (c_i) selon le poids ϕ_i

Technique « Weighted Fair-Queueing » (Demers, Keshav et Shenker 1989)
PGPS « Packet Generalized Processor Sharing » (Parekh et Gallager 1993)

– GPS signifie que l'interruption de tâche peut se faire à n'importe quel moment
(PGS non applicable directement aux réseaux)

– PGPS = version de GPS appliquée aux réseaux

– PS + équité en tenant compte de l'allocation préalable des connexions (poids ϕ_i)

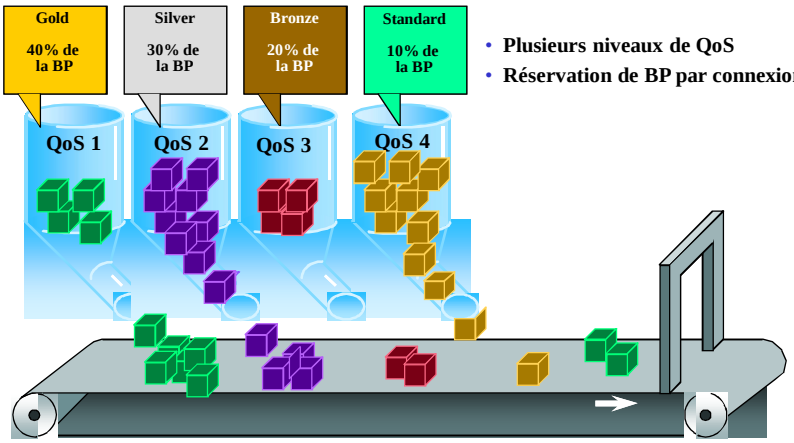


$$r_i = \frac{\phi_i}{\sum_j \phi_j} R \qquad \sum_j \phi_j \leq 1$$

r_k : débit du flux k R : débit du lien

– GPS garantit le débit

Principe général de WFQ



- **Plusieurs niveaux de QoS**
- **Réservation de BP par connexion**

WFQ = une mise en œuvre de PGPS

➔ Principe général de WFQ

- ***V(t)*** : temps virtuel du système qui capte progression de la quantité de service normalisé à l'instant *t* . *V(t) peut être défini par :*

$$V(t_{j-1}+\alpha) = V(t_{j-1}) + \frac{\alpha}{\sum_{i\in B_j}\phi_i} \qquad \frac{dV(t)}{dt} = \frac{R}{\sum_{i\in B_j} r_i}$$

B_j : ensemble des connexions en attente pendant une période d'activité t_{j-1} et t_j. $t \in [t_{j-1} .. t_j]$

V(t) est le plus complexe à calculer dans la pratique.

- Pour tout paquet *k* de la connexion *i* à transmettre, on associe : S_i^k et F_i^k
- Temps de début : $S_i^k = max\{F_i^{k-1}, V(a_i^k)\}$ a_i^k : instant d'arrivée du paquet *k*
- Temps de fin : $F_i^k = S_i^k + \frac{L_i^k}{R\phi_i}$
- L'ordonnancement se fait sur la base des temps de fin (F_i^k)

Performances de WFQ

Hypothèses

- Le flux *c* est conforme à un seau percé (ρ^c, σ^c).
- Tous les routeurs sur le chemin implantent WFQ.

Notations
c : flux/connexion
s : routeur
ϕ_s^c : *poids de la connexion c* au niveau de *s*
r_s^c : débit du flux *c* transitant par *s*
C_s : ensemble des connexions passant par *s*
r_s : débit du lien de sortie de *s*
K^c : nombre de routeurs sur la route de *c*
L^c : taille max de paquet du flux *c*
$Lmax_s$: taille max de paquet transitant par *s*
π : délai de propagation sur le tout le chemin

➔ **Borne de débit garanti**

$$r_s^c = \frac{\phi_s^c}{\sum_{j \in C_s} \phi_s^j} r_s$$

➔ **Borne de délai de bout en bout garanti**

$$\frac{\sigma^c + (K^c - 1)L^c}{\rho^c} + \sum_{s=1}^{K^c} \frac{L\,max_s}{r_s} + \pi$$

Mise en œuvre de WFQ

➔ **On étudie la stratégie WFQ pour chaque lien de sortie dans le réseau considéré.**

n **: nombre de connexions passant par un lien de sortie donné**

u_i **: proportion de la bande passante allouée à la connexion *i* sur le lien de sortie au moment de l'établissement de connexion.**

U **: somme des proportions de bande passante allouées à toutes les connexions sur le lien de sortie considéré (*U* ≤ *1*)**

● **Une connexion est dite inactive si elle n'a aucun paquet en attente dans la file du lien de sortie ou en cours de transmission sur le lien de sortie ; sinon elle est dite active.**

● **Les paquets en sortie associés à une connexion *i* sont placés dans la file du lien de sortie. Les paquets d'une même connexion sont servis selon l'ordre FIFO ; mais l'ensemble des paquets n'est pas servi en FIFO. Un paquet d'une connexion est prêt pour transmission quand il est le premier des paquets en attente pour cette connexion.**

4. Ordonnancement PGPS et WFQ

- **Pour gérer les paquets prêts des différentes connexions actives, une file à priorité *FP* est utilisée par l'ordonnanceur. Chaque connexion active *i* a une entrée (etv_i, *i*) dans la file *FP*. Cette entrée est insérée dans la file *FP* selon son échéance exprimée en temps virtuel (*etv*).**
- **Les paquets prêts sont transmis selon l'ordre donné par la file *FP* (c-à-d, selon l'ordre des échéances virtuelles).**

Ordonnancement de paquets

- Après un temps d'inactivité du lien de sortie (car il n'y avait aucun paquet à transmettre), quand le premier paquet arrive, sur une connexion *i*, l'ordonnanceur calcule l'échéance virtuelle etv_i et la place avec *i* en tête de la file *FP*. La transmission de ce paquet commence immédiatement (car la file du lien de sortie était vide au moment de cette arrivée).

- Quand un lien de sortie est actif (c-à-d qu'il y a un paquet en cours de transmission sur ce lien), l'ordonnanceur calcule etv_i pour tout paquet qui arrive, sur une connexion *i* qui est inactive, et l'insère dans la file *FP*. Si cette connexion *i* était active, le paquet arrivé est mis en file d'attente de la connexion sans traitement.

- Lorsqu'un paquet termine sa transmission, il est retiré de la file d'attente et son entrée est retirée de la file *FP*. Si la connexion *i*, qui est la source du paquet qui vient de terminer sa transmission, est active, l'ordonnanceur calcule l'échéance virtuelle etv_i de son nouveau paquet prêt et l'insère dans la file *FP*. Ensuite, il commence la transmission du paquet dont l'entrée est la première dans la file *FP*.

4. Ordonnancement PGPS et WFQ

Calcul des nombres de fin pour un lien de sortie

U_b = total des proportions de bande passante de toutes les connexions actives pour le lien.

TV : temps virtuel du lien.

t : instant courant ; t_{-1} : instant de la dernière mise à jour de U_b et TV.

e : temps de transmission de paquet (hypothèse : e = 1, pour le paquet de taille maximale)

Règles de calcul

1. Quand le lien est inactif, $TV = 0$; $U_b = 0$; $t_{-1} = 0$ et tous les etv_k ($k=1, \ldots n$) sont à 0.

2. Quand le premier paquet (de temps de transmission e et généré par la connexion *i*) arrive, alors que le lien est inactif, un intervalle d'activité du lien débute. Les opérations suivantes sont effectuées : $t_{-1} = t$; $U_b = U_b + u_i$; $etv_i = etv_i + e/u_i$; insertion de (etv_i, *i*) dans *FP*.

3. Pour tout *i*, quand un paquet de la connexion *i* arrive à l'instant *t*, dans un intervalle d'activité du lien de sortie, si la connexion *i* était inactive avant cette arrivée, alors :

$TV = TV + (t - t_{-1})/U_b$; $etv_i = \max(TV, etv_i) + e/u_i$; insertion de (etv_i, *i*) dans la file *FP* ;
$t_{-1} = t$; $U_b = U_b + u_i$

4. Pour tout *i*, quand un paquet de la connexion *i* termine sa transmission :

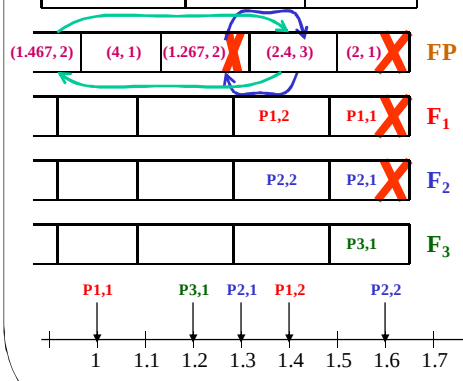
- Si la connexion *i* reste active : $etv_i = etv_i + e/u_i$ (e = temps de transmission du nouveau paquet prêt de la connexion *i*) et insertion de (etv_i, *i*) dans la file *FP*.

- Si la connexion *i* devient inactive : $TV = TV + (t - t_{-1})/U_b$; $t_{-1} = t$; $U_b = U_b - u_i$.

4. Ordonnancement PGPS et WFQ

Exemple

Connexion	e	u_i
C_1	0.5	1/4
C_2	0.1	1/2
C_3	0.2	1/8



	TV	U_b	t_{-1}	etv_1	etv_2	etv_3
t=0	0	0	0	0	0	0
t=1	0	1/4	1	2	0	0
t=1.2	0.8	3/8	1.2	2	0	2.4
t=1.3	1.067	7/8	1.3	2	1.267	2.4
	P2,1 passera avant P3,1					
t=1.4	P1,2 arrive, mais il n 'est pas traité					
t=1.5	P1,1 termine sa transmission P2,1 commence sa transmission					
	1.067	7/8	1.3	4	1.267	2.4
t=1.6	P2,1 termine sa transmission					
	1,067	7/8	1.3	4	1.467	2.4
	P3,1 commence sa transmission					

5. Ordonnancement DEDD et JEDD

Ordonnancement à priorité DEDD

➔ **Delay Earliest Due-Date [Ferrari et Verma]**

- EDF (Earliest Deadline First) pour les paquets
- Garantie de contraintes de délai (de bout en bout en cas de réseau homogène)
- Modèle orienté connexion
- Flux périodique

Principe de Delay EDD

➔ **Etablissement de connexion** (i.e. recherche de chemin)

- Chaque source i demande l'établissement de connexion i en spécifiant P_i, $tmax_i$ et D_i (période, temps maximum de transmission de paquet et délai de bout en bout).

- Chaque routeur par lequel passe la connexion i teste si la connexion peut être acceptée ou refusée. En cas d'acceptation, il détermine le délai relatif qu'il peut garantir et le rajoute dans le paquet de demande de connexion, marque les réservations à faire (ces réservations peuvent être revues à la baisse lorsque le paquet de réponse de connexion revient au routeur) et passe la demande de connexion au routeur suivant.

- A la réception du paquet de demande de connexion, le destinataire du flux de données détermine si la somme des délais intermédiaires est supérieure au délai de bout en bout D_i. Si le test est positif, la demande de connexion échoue. Sinon le destinataire détermine un délai relatif $D_{i,k}$ pour chaque routeur k (ce délai ne pouvant être inférieur à celui déjà déterminé par le routeur au moment où il a passé la demande de connexion) et place tous les délais relatifs calculés dans un paquet d'acceptation de connexion. Ce paquet d'acceptation de connexion est envoyé sur le chemin inverse de la demande, ce qui permet à chaque routeur k de connaître le délai relatif $D_{i,k}$ qui lui a été assigné. Chaque routeur k effectue les réservations (tampon et bande passante sur le lien de sortie) requises pour garantir $D_{i,k}$.

5. Ordonnancement DEDD et JEDD

➔ **Contrôle de débit et ordonnancement**

- Au niveau de chaque routeur k, les paquets sont transmis sur les liens de sortie selon leur délai $D_{i,k}$ (avec la stratégie EDF, c-à-d, le paquet dont le délai est le plus court).

- Contrairement à WFQ, la stratégie EDF ne permet pas de contrôler, de manière intrinsèque, les surcharges générées par des sources qui ne respectent pas leur contrat. Le mécanisme de contrôle mis en œuvre par Delay EDD est le suivant :

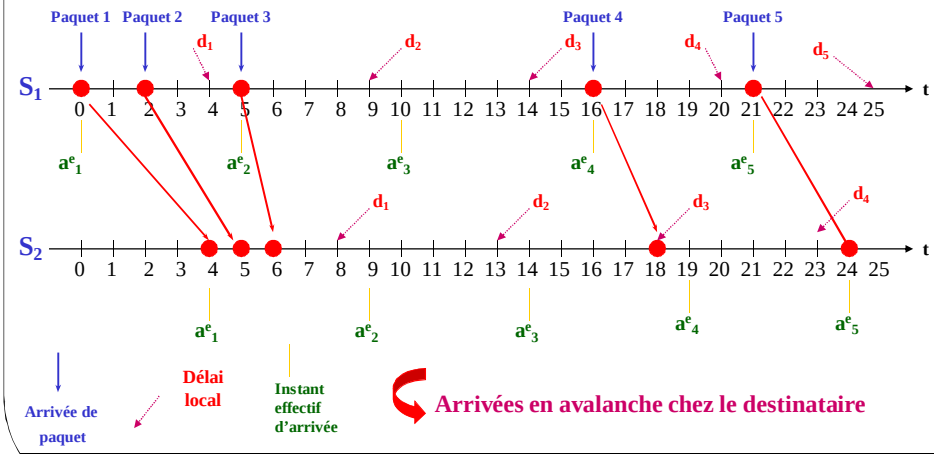
le délai $d_{i,j}$ pour la transmission du paquet j de la source i n'est pas calculé sur la base de son instant d'arrivée $a_{i,j}$, mais sur la base de l'instant d'arrivée effectif du premier paquet de la connexion i, $a^e_{i,1}$. L'instant d'arrivée effectif du paquet j est $a^e_{i,j} = \max (a^e_{i,j-1} + P_i, a_{i,j})$. L'échéance locale du paquet j au niveau du routeur k est : $d_{i,j} = a^e_{i,j} + D_{i,k}$.

➔ **Observation**

Les implantations de Delay-EDD varient essentiellement selon : les hypothèses sur les paramètres de connexion, le test effectué par le CAC, la manière dont le destinataire de connexion répartit les délais intermédiaires sur les routeurs.

Exemple d'application de Delay EDD

➔ On considère une seule connexion définie par $P_i = 5$ et $D_i = 8$ qui passe par deux routeurs S_1 et S_2. Chaque routeur garantit un délai local relatif égal à 4.



Exemple d'implantation de l'algorithme Delay-EDD (Liu 2000)

Détail des opérations d'un routeur *k* selon la stratégie Delay-EDD

(Le temps de transmission de paquet est supposé égal à 1 unité de temps)

- La description suivante de Delay EDD est applicable à des flux périodiques.

- Informations gérées par l'ordonnanceur pour chaque lien de sortie
  - La densité totale Δ de toutes les connexions utilisant le lien k (Δ est initialisé à 0)
  - Le total d'espace mémoire BS alloué aux connexions existantes (BS est initialisé à 0)
  - La période minimale P_{min} de toutes les périodes des connexions utilisant le lien k (P_{min} est initialisée à ∞)

Détail des opérations d'un routeur k selon la stratégie Delay-EDD

1. Phase d'établissement de connexion

- Sur réception d'une demande de connexion avec $(P_i, 1, D_i)$, rejeter la demande si $\Delta + 1/P_i \geq 1 - 1/P_{min}$; sinon :
  - Calculer le délai local à offrir $DL_{i,k}$ ($DL_{i,k} = min(P_i, 1/(1 - \Delta - 1/P_{min})$) et la bande passante requise (égale à $1/DL_{i,k}$) pour garantir ce délai et faire $\Delta = \Delta + 1/DL_{i,k}$.
  - Calculer la taille mémoire requise $BS_{i,k}$, puis $BS = BS + BS_{i,k}$.
  - Mettre les valeurs de $DL_{i,k}$ et $BS_{i,k}$ dans le paquet de demande de connexion et le passer au routeur suivant.
- Sur réception d'un paquet d'acceptation de la connexion i, contenant les valeurs des délais $D_{i,k}$ et l'espace mémoire $BS_{i,k}$ fixées par le destinataire final de la connexion:
  - Remplacer la valeur de $DL_{i,k}$ par celle de $D_{i,k}$ pour la connexion i.
  - Décrémenter Δ de $1/DL_{i,k} - 1/min(D_{i,k}, P_i)$. /* on ajuste la réservation de la bande passante*/
  - Décrémenter BS de $BS_{i,k} - bs_{i,k}$. /* on ajuste la réservation de la mémoire*/.
  - $P_{min} = \min(P_{min}, P_i)$
- A la réception d'un paquet de refus d'établissement de la connexion i : $BS = BS - BS_i$ et $\Delta = \Delta - 1/DL_{i,k}$
  /* On libère les ressources réservées précédemment. */

2. Phase d'échange de données

- A la réception du $j^{ème}$ paquet de données de la connexion i à l'instant $a_{i,j}$, lui affecter une échéance égale à $\max(a_{i,j}, (j-1)P_i + D_{i,k})$ et placer ce paquet dans la file FIFO de la connexion i.
- Les paquets en attente sont transmis selon leur échéance (EDF) : servir le premier paquet parmi toutes les files associées aux différentes connexions qui utilisent le lien de sortie.

Opérations chez le destinataire de la connexion selon la stratégie Delay-EDD

1. Phase d'établissement de connexion

- Si la demande de connexion atteint le destinataire final c'est que tous les routeurs intermédiaires l'ont acceptée et ont offert des délais intermédiaires dont la somme est inférieure au délai de bout en bout D_i.
- Pour répartir équitablement les délais intermédiaires, le site destinataire recalcule les délais intermédiaires effectifs que chaque routeur k devra respecter et les renvoie dans un message d'acceptation de connexion empruntant le chemin inverse du message de demande de connexion pour que les routeurs ajustent leurs délais locaux (ou intermédiaires).

  On note :

  $DL_{i,k}$: délai local accepté le site k pour la connexion i.

  $D_{i,k}$: délai local que devra respecter le site k pour la connexion i.

  m : nombre de routeurs sur lesquels est établie la connexion i.

$$D_{i,k} = \frac{1}{m}\left[D_i - \sum_{j=1}^{m} DL_{i,j} \right] + DL_{i,k}$$

2. Phase d'échange de données : tout paquet reçu est accepté

Ordonnancement « Jitter EDD » (Verma, Zhang et Ferrari 1991)

(Extension de la stratégie Delay EDD pour contrôler la gigue)

Principe de la stratégie Jitter EDD

1. Phase d'établissement de connexion

- **Chaque routeur k calcule le délai local, $D_{i,k}$, et la gigue locale, $J_{i,k}$, qu'il peut garantir à la connexion i et les envoie dans un message au routeur suivant. Ainsi, le routeur k doit garantir un délai local compris dans $[D_{i,k} - J_{i,k}, D_{i,k}]$.**

Pour simplifier l'algorithme Jitter EDD, ses auteurs imposent que les valeurs $D_{i,k}$ et $J_{i,k}$, proposées par le routeur k pour doivent être égales. La technique de calcul de ces valeurs dépend de la politique de de réservation de ressources du routeur.

- **Pour absorber la gigue, le routeur k doit déterminer l'espace mémoire nécessaire à la connexion i, :**

$$BS_{i,k} = Smax_i \times \left(\left\lceil D_{i,k} / P_i \right\rceil + \left\lceil J_{i,k-1} / P_i \right\rceil \right)$$

$Smax_i$ et P_i : taille maximum de paquet et période de transmission de la connexion i

- **A la réception d'un message d'acceptation de connexion, le routeur k réajuste les valeurs de ses paramètres $D_{i,k}$, $J_{i,k}$ et $BS_{i,k}$. Il libère les ressources, s'il reçoit un message de refus de connexion.**

Stratégie Jitter EDD (suite)

➔ **A la réception d'un message de demande de connexion contenant les valeurs de délais et gigues intermédiaires que les routeurs se trouvant sur le chemin entre la source et la destination ont acceptées, le site de destination de la connexion i calcule les délais et gigues intermédiaires pour les répartir de manière équitable sur les routeurs ayant accepté la connexion. Il leur renvoie les valeurs calculées pour réajuster leurs valeurs locales et appliquer ces valeurs.**

$$D_{i,k} = \frac{1}{m}\left[D_i - \sum_{j=1}^{N} J_{i,j} \right] + J_{i,k}$$

$$J_{i,m} = J_i$$

$$J_{i,k} = D_{i,k} \; (\forall k = 1,\ldots,m-1)$$

m : nombre de routeurs traversés par la connexion i

Stratégie Jitter EDD (suite)

2. Phase d'échange de données

- **Marquage de paquets :** Le routeur $k-1$ ajoute à tout paquet j de la connexion i, à passer au routeur k, une marque temporelle $h_{i,j}$ ($h_{i,j} = D_{i,k-1} - DR_{k-1}$).

  DR_{k-1} : délai réel d'attente du paquet j dans le routeur $k-1$.

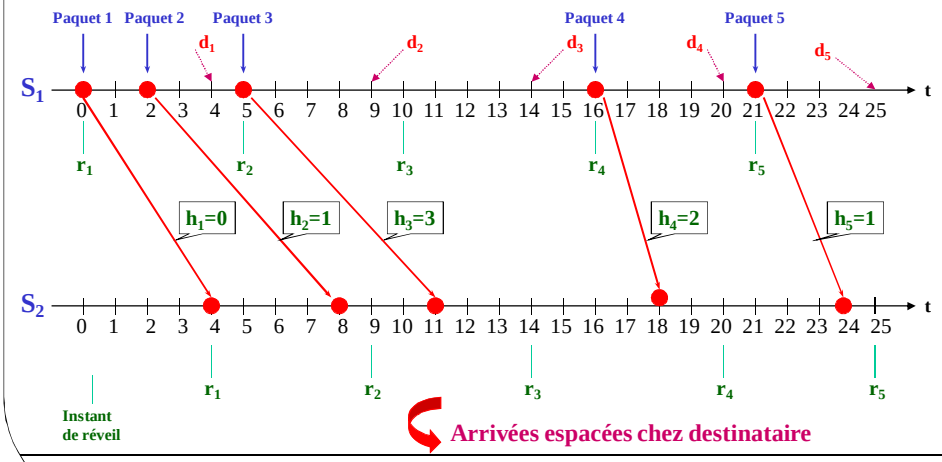
- **Rétention de paquets :** Quand le routeur k reçoit un paquet j de la connexion i, avec une marque temporelle $h_{i,j}$, il calcule l'instant de réveil de ce paquet $r_{i,j}$ ($r_{i,j} = max(a^e_{i,j}, a_{i,j} + h_{i,j})$. Ensuite l'ordonnanceur retient ce paquet dans dans une file spéciale (ou buffer d'élasticité) jusqu'à l'instant $r_{i,j}$. A l'instant $r_{i,j}$, une échéance $d_{i,j} = r_{i,j} + D_{i,k}$ est affectée au paquet et le paquet est mis en file d'attente normale. Ensuite les paquets sont ordonnancés par EDF comme dans le cas de Delay EDD.

  $a_{i,j}$: instant d'arrivée du paquet j de la source i

  $a^e_{i,j}$: instant d'arrivée prévu du paquet j = $\max(a^e_{i,j-1} + P_i, a_{i,j})$.

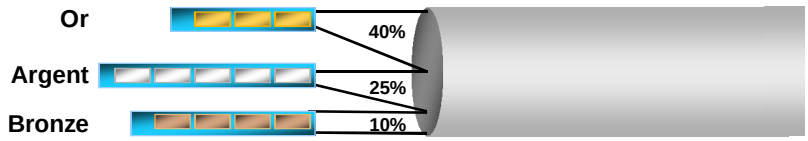
Exemple d'application de Jitter EDD

➔ On considère une seule connexion définie par $P_i = 5$ et $D_i = 8$ qui passe par deux routeurs S_1 et S_2. Ces routeurs appliquent la stratégie Jitter EDD. Chaque routeur garantit un délai max de 4 ms et une gigue de 4 ms.



6. Ordonnancement CBQ

Class-based Queuing (1/3)
[Floyd, Jacobson 95]

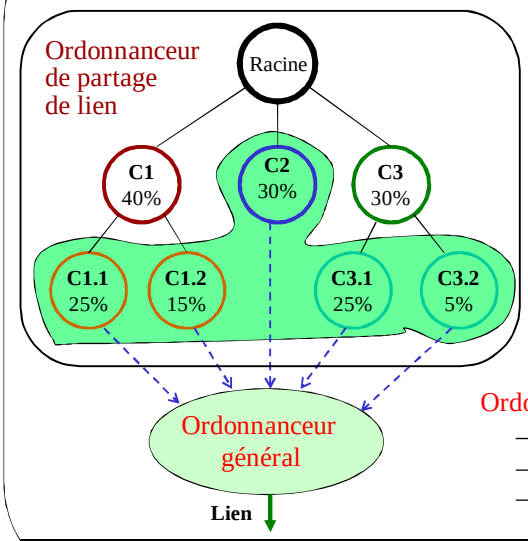


- Chaque classe se voit réserver une part (selon son poids) de bande passante
- Les poids garantissent une bande passante minimum
- Les flux sont regroupés en classes
- La bande passante laissée libre par un flux d'une classe et d'abord utilisée par les flux des classes 'sœurs'.

➔ **C'est la discipline la plus implantée actuellement par les routeurs dits à QoS.**

6. Ordonnancement CBQ

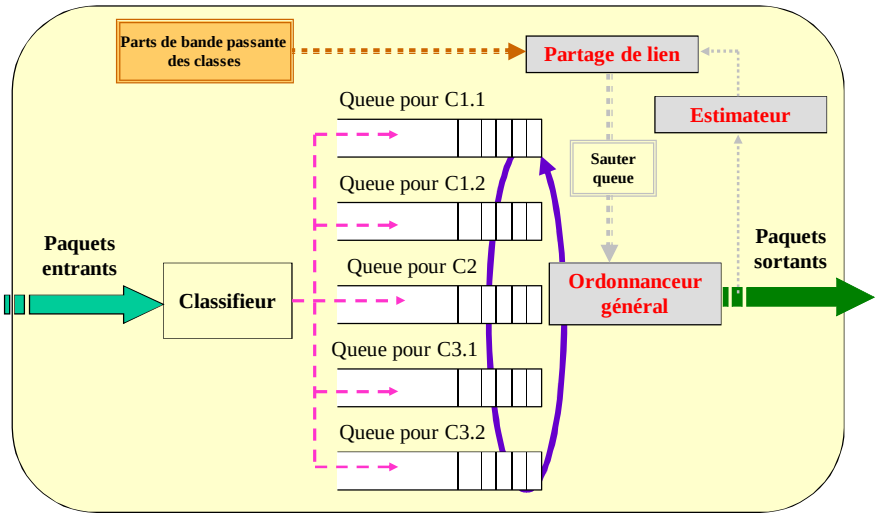
Class-based Queuing (2/3)



Ordonnanceur de partage de lien
- assure le partage du lien selon les poids
- la bande passante laissée libre par une classe est distribuée en priorité parmi les classes sœurs, puis distribuée équitablement entre les autres classes
- utilisation d'estimateurs pour déterminer la bande passante utilisée par chaque classe et réguler les classes (arrêter les classes et distribuer la bande passante en excès).

Ordonnanceur général
- ordonnance les paquets
- ne connaît que les feuilles de l'arbre
- WRR, WFQ, RR, Priority queuing, …

Class-based Queuing (3/3)



Terminologie de CBQ

- **Niveau** : les feuilles de l'arbre constituent le niveau 1 (le plus bas) et la racine le niveau n (le plus élevé).
- **Limite** de consommation
  - Si une classe a récemment consommé plus de bande passante que ce qui lui a été alloué, elle est dite au-dessus de la limite. Si elle en a consommé moins, elle est dite au-dessous de la limite. Sinon elle dite à la limite.
- **Satisfaction**
  - Pour une classe feuille : si elle est au-dessous de la limite et qu'elle a (de manière persistante) des paquets en attente, elle est dite non satisfaite.
  - Pour une classe intermédiaire : si elle est au-dessous de la limite et qu'elle a des descendants qui ont des paquets en attente, elle est dite non satisfaite.
  - Autrement, la classe est dite satisfaite.
- **Régulation**
  - Une classe est dite non régulée si elle est en cours d'ordonnancement par l'ordonnanceur général.
  - Autrement, elle en cours d'ordonnancement par l'ordonnanceur de partage de lien et elle est dite régulée.
  - La régulation de classe est déterminée par les guides de partage de lien.

Guides de partage de lien

- Une classe reste non régulée si l'une des conditions suivantes est satisfaite :
  - La classe n'est pas au-dessus de la limite.
  - La classe n'a pas d'ancêtre de niveau *i* qui est au-dessus de la limite et qu'il n'y a pas de classes non satisfaites dans la structure de partage de lien à des niveaux inférieurs à *i*. Ainsi, la classe emprunte à cet ancêtre.

- Autrement, la classe sera régulée par l'ordonnanceur de partage de lien.

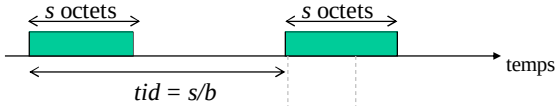
- A cause de problèmes de complexité de calcul des guides de partage de lien, une approximation est proposée par les auteurs :
  - Une classe reste non régulée si l'une des conditions suivantes est satisfaite :
    - La classe n'est pas au-dessus de la limite.
    - La classe a un ancêtre au-dessous de la limite.
  - Sinon la classe sera régulée par l'ordonnanceur de partage de lien.

Estimateur (1/3)

- L'estimateur mesure la bande passante consommée par chaque classe pour savoir si elle est au dessus de la limite afin de la suspendre.

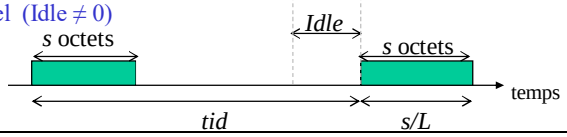
- Une méthode (heuristique) proposée est de considérer les instants d'inter-départ de paquets et d'utiliser une moyenne exponentielle pondérée (EWMA: exponential weighted moving average) pour lisser les valeurs de *Idle*.
  - Soit ***b*** (en b/s) la bande passante allouée à la classe considérée, ***L*** (en b/s) la bande passante totale du lien, ***s*** la taille (fixe) de paquet en octets et ***tid*** (en sec) le temps d'inter-départ.

Cas idéal ***b*** = débit d'émission (Idle = 0)



Idle = tid - s/b

Cas réel (Idle ≠ 0)



Estimateur (2/3)

- Les paramètres sont mis à jour après chaque sortie de paquet.
- Paramètre ***Idle***

  Idle = tid - s/b

  Idle désigne la différence entre le temps théorique d'inter-départ et le temps d'inter-départ réel. Quand *Idle>0*, le dernier paquet a été émis plus tardivement que son instant théorique. Quand *Idle<0*, le paquet a été émis trop tôt et quand il est égal à 0, il a été émis à temps.

- Paramètre ***Avgidle***

  Avgidle ← *(1-w)*Avgidle + w*Idle*

  Avgidle permet d'estimer la consommation d'une classe par rapport à sa consommation nominale.
  - Si *Avgidle* > 0 : la classe est considérée comme étant au-dessous de la limite
  - Si *Avgidle* ≤ 0 : la classe est considérée comme étant au-dessus de la limite

  w est un paramètre de EWMA ; il est généralement égal à 15/16 ou 31/32.

  Avgidle initialisé à *maxidle*.

Estimateur (3/3)

- Paramètre ***maxidle***
  - Lorsqu'une classe est au repos (idle) pendant une longue période, *Avgidle* devient élevé. Si un burst de paquets arrive, il consomme beaucoup de temps avant d'être pénalisé.
  - Pour éviter de monopoliser pendant longtemps le lien après un long silence, ***maxidle*** est introduit pour limiter la taille de burst après une période de silence.
  - La valeur de *maxidle* qui permet d'envoyer jusqu'à *m* paquets, de taille *s*, constituant un burst est la suivante :

$$\max idle = \frac{s}{L} \times \left(\frac{L}{b} - 1 \right) \times \left(\frac{1 - w^m}{w^m} \right)$$

- Paramètre ***minidle***
  - C'est la borne inférieure de *Avgidle* afin d'éviter d'accumuler trop de pénalité lors de sur-consommation de bande passante durant des bursts. En général, ce paramètre est mis à 0.

Régulation

- Quand une classe est au-dessus de sa limite et qu'elle ne peut pas emprunter de ses ancêtres, elle est suspendue pendant une durée *Offtime*.

- La quantité *Offtime* doit être choisie de sorte que le paramètre *Avgidle* de la classe redevienne positif (ou nul) après la suspension et permettre de transmettre ensuite un burst de *m* paquets au maximum :

$$Offtime(m) = \left(\frac{1}{1-w} \times \frac{s}{L} \times \left(\frac{L}{b} - 1 \right) \times \frac{\left(1 - w^{m-1}\right)}{w^{m-1}} \right) + \left(\frac{s}{L} \times \left(\frac{L}{b} - 1 \right) \right)$$

$$Offtime(1) = \frac{s}{L} \times \left(\frac{L}{b} - 1 \right) = \frac{s}{b} - \frac{s}{L}$$

***s/b* temps de transmission au débit *b* (débit alloué)**
***s/L* temps de transmission au débit *L* (débit du lien)**

Ordonnaneur général

- Généralement l'ordonnanceur général est WRR avec des poids qui correspondent à la bande passante allouée à chaque classe feuille. D'autres ordonnanceurs peuvent aussi être utilisés.

- Lordonnanceur général coopère avec l'ordonnanceur de partage de lien en contrôlant la valeur de *Offttime* :
  - Si *Offttime=0* ou si un temps égal à *Offttime* s'est écoulé depuis la suspension, alors effectuer l'ordonnancement en WRR normal.
  - Sinon, tester si la bande passante d'un ancêtre peut être prêtée à la classe
    - Si oui: ordonnancer avec WRR normal.
    - Sinon : sauter la classe pour le tour courant.

7. Conclusion

Récapitulatif

FIFO	Pas de distinction entre les paquets – Pas de réservation de ressources
FP	Les queues de haute priorité sont servies d'abord. Garantie de transmission de trafic critique/urgent
RR	Les queues sont servies à tour de rôle et de la même manière
WRR	Les queues sont servies à tour de rôle et en tenant compte de leur poids
DRR	Les queues sont servies à tour de rôle en fonction de la taille de leurs paquets et du crédit accumulé.
WFQ	Garantie d'équité entre les flux – Les queues sont traitées en RR
D-EDD	Paquets servis selon EDF. Garantie de délai
J-EDD	Paquets servis selon EDF + rétention de paquets. Garantie de gigue
CBQ	Classification, réservation et service selon les classes. Garantie de bande passante et éventuellement de délai

7. Conclusion

Comparaison des disciplines de service

	WFQ	Delay EDD	Jitter EDD	WRR (1)
Test d'acceptation de connexion	O(1)	O(1)	O(1)	O(1)
Complexité d'ordonnancement	O(n)	O(log n)	O(log n)	O(1)
Délai de bout-en-bout maximal	(2) $E/u + m(e+1)$	D_i	D_i	$P_i + (m-1)RL$
Gigue de bout-en-bout maximale	Constante × m (Pas de garantie)	Constante × m (Pas de garantie)	J_i	$P_i - e_i + (m-1)(RL-1)$
Besoin en espace mémoire	Constante × m	Constante × m	Constante	(3) $(1 + \lceil (k-1)(RL-1)/P_i \rceil) * e_i$
Files d'attente exigées	Oui	Oui	Oui	Non

n : nombre de connexions partageant un lien de sortie
m : nombre de routeurs
e_i : Durée de transmission du paquet i
e : Durée maximale de transmission de paquet

(1) Hypothèse : tous les routeurs utilisent les même paramètres RL et wt_i.
(2) Si le trafic est conforme un seau percé de paramètres (E, u).
(3) k indique le rang (k = 1, …, m) du routeur sur le chemin emprunté par la connexion.

7. Conclusion

Des problèmes à résoudre

- Choix de discipline de service pour chaque routeur en fonction des applications.
- Chaque routeur peut avoir ses propres stratégies et algorithmes d'ordonnancement ⇒ Difficulté de garantie de QoS de bout en bout
- Fonctionnement avec QoS dégradée en cas de surcharge
- Complexité des algorithmes en cas de contraintes multiples (maîtrise des heuristiques)
- Modèles analytiques de bornes
- Adaptation des algorithmes aux changements de trafic
- Modèles de conception et d'analyse de composants combinés
  Routage/ordonnancement/CAC/Réservation de ressources
- Algorithmes d'ordonnancement et réseaux sans fil avec capacités de lien dynamiques
- Effets des mécanismes de sécurité sur les performances de l'ordonnancement

Chapitre 6

Contrôle de congestion et gestion de buffers

1. Principes d'allocation de ressources

Allocation de ressources (1/4)

➔ Ressources = CPU, mémoire, bande passante…

➔ QoS fournie dépend des ressources allouées pour le service.

➔ Allocation de ressources ⇒ Politique d'allocation
(droits d'utiliser des ressources, coûts, …)

➔ Allocation de ressources

– Sans Négociation : rigide (tout ou rien), sûre

– Avec Négociation : à la connexion, flexible, complexe

– Avec Renégociation : s'adapter au réseau à tout moment,
transmettre à moindre coût, (très) complexe

1. Principes d'allocations de ressources

Allocation de ressources (2/4)

➔ Stratégies d'allocation de ressources : Allocation non statistique ou statistique

➔ Allocation non statistique ("peak bandwidth allocation")

- Allouer une capacité maximale s'il reste encore assez de bande passante
- Adaptée au trafic à rythme fixe (CBR)
- Risque de sous-utilisation du réseau

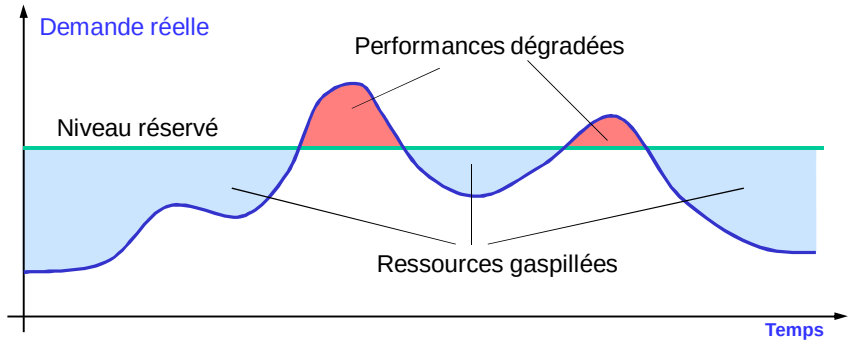
➔ Allocation statistique

- L'allocation ne se fait pas sur la base du débit maximum de connexion
- La somme des débits correspondant aux connexions acceptées peut être supérieure à celui des ports de sortie du nœud.
- Adaptée à des flux variables (VBR)
- Difficulté de prédire la garantie de QoS

Allocation de ressources (3/4)

➔ **Problèmes de la réservation de ressources**

- Des ressources réservée mais non utilisées : Perte
- Ressources minimales/moyennes à réserver : difficiles à déterminer



Allocation de ressources (4/4)

➔ **Gestion de buffers (tampons ou queues)**

- Gestion de buffers (un flux / une file, une file partagée entre plusieurs flux)
- Mémoire de stockage des paquets limitée ⇒ Contrôler son utilisation
- Deux décisions majeures : Quand rejeter les paquets ?
  Quels paquets rejeter ?

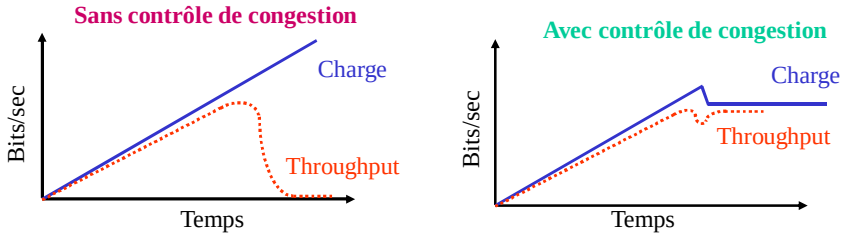
➔ **Contrôle et traitement de trafic**

- Contrôle de congestion
- Contrôle de trafic de l'utilisateur
- Façonnage du trafic de l'utilisateur (« traffic shaping »)
- Marquage de paquets

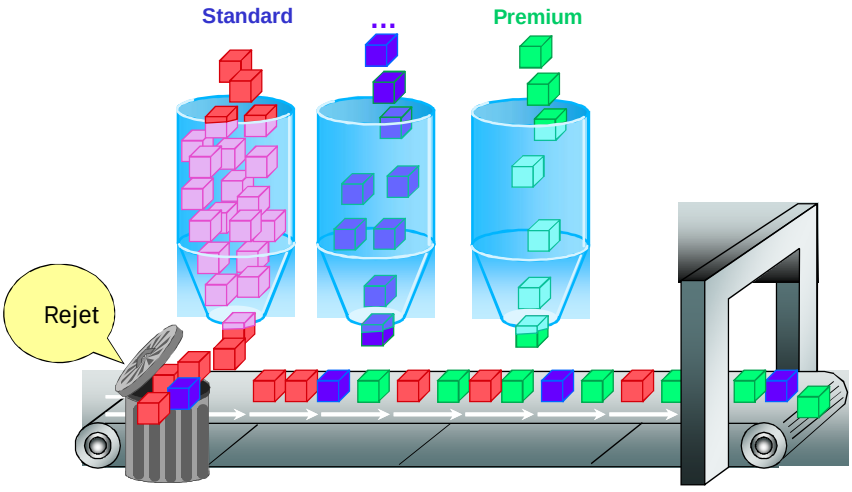
2. Problème de congestion

2. Problème de congestion

- Flux aléatoires + Mémoire limitée + bande passante limitée ⇒
  Possibilité de congestion
- Effets négatifs : taux de perte élevé, latence élevée
- Nécessité de contrôle de congestion



Suppression de paquets



Stratégies de suppression de paquets

- Politique de suppression de paquets
  (Quels paquets supprimer et dans quelles conditions ?)
  - Rejeter sans distinction les paquets arrivant en fonction de l'état courant des buffers
  - Utiliser des seuils de congestion (anticiper les situations de congestion)
  - Utiliser les données des contrats utilisateur pour rejeter les paquets hors profil
  - Affecter des priorités basses aux paquets hors profil

- Méthodes : réactives vs préventives

- **Techniques de contrôle de congestion**
  - RED ("Random Early Detection")
  - Variantes de RED (FRED, WRED…)
  - ECN ("Explicit Congestion Notification")
  - Autres

Compromis :
Complexité/ Performance/Prévision
&
Détermination des seuils

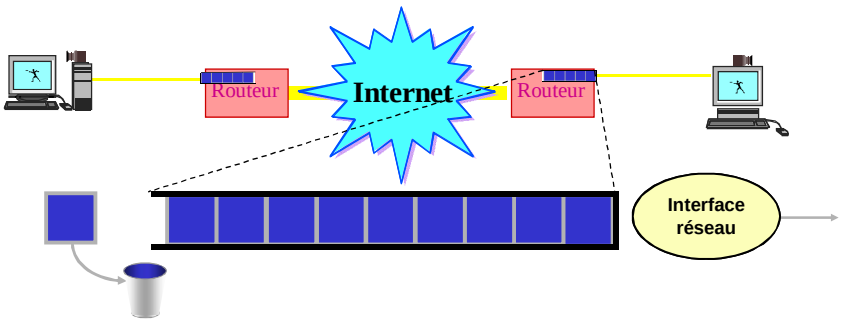
2. Problème de congestion

Propriétés de stratégies de contrôle de congestion

- Anticipation des situations de congestion (aspects probabilistes)
- Notification aux sources qui persistent à dépasser leur limite
- Rendement du réseau (‘throughput’) élevé
- Latence limitée
- Adaptation aux bursts (rafales de paquets)
- Taille des queues appropriée (pas de surdimensionnement inutile)
- Équité au niveau du rejet de paquets
- **Complexité d’implantation réduite**

3. Technique FIFO

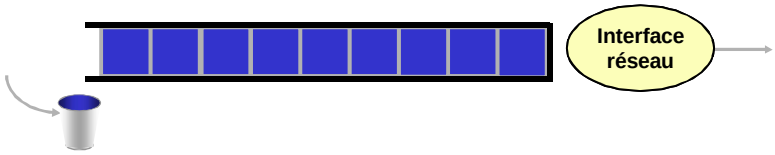
Mise en queue (‘bufferisation’) dans les routeurs IP



- Taille de queue (buffer)
  - Suffisamment d’espace pour les rafales
  - Latence
- Rejet de paquets
  - Quand ?
  - Lesquels ?

Technique FIFO pour la gestion de queue

- FIFO : technique la plus simple pour gérer les files d'attente de routeurs.
  - une seule queue par interface de sortie
  - servir le paquet en tête
  - mettre en queue le paquet arrivant si file non pleine
  - rejeter le dernier paquet si queue pleine



- FIFO a le mérite de ne pas poser de problème au fonctionnement d'Internet fondé sur le principe du meilleur effort. Pas de contraintes pour l'implantation de routeurs IP.

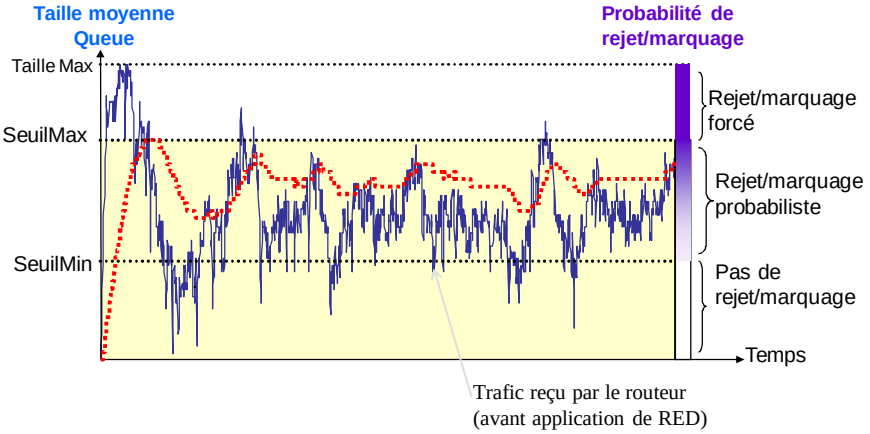
- Limites (inconvénients) de FIFO :
  - Impossible de différencier les trafics (car on a une seule queue) : les paquets arrivant sont rejetés sans distinction en cas de queue saturée. Donc on ne prend pas en compte l'importance des paquets.
  - Il n'y a pas d'isolation de flux. Un flux qui persiste à envoyer plus de paquets finit par avoir plus de service par rapport aux autres qui s'auto limitent.

- Inconvénients du rejet en fin de queue ('drop-tail'):
  - Les routeurs doivent avoir des queues de taille importante pour maintenir un taux d'utilisation élevé.
  - Des buffers larges impliquent des latences importantes.

⇒ Nécessité d'utiliser des méthodes de gestion actives

4. Technique RED (Random Early Detection)

- RED : technique la plus populaire pour l'évitement de congestion
- Proposée par Floyd et Jacobson (1993) pour gérer les flux TCP
- Gestion de queue avec des seuils - technique de gestion active (préventive)



4. Technique RED

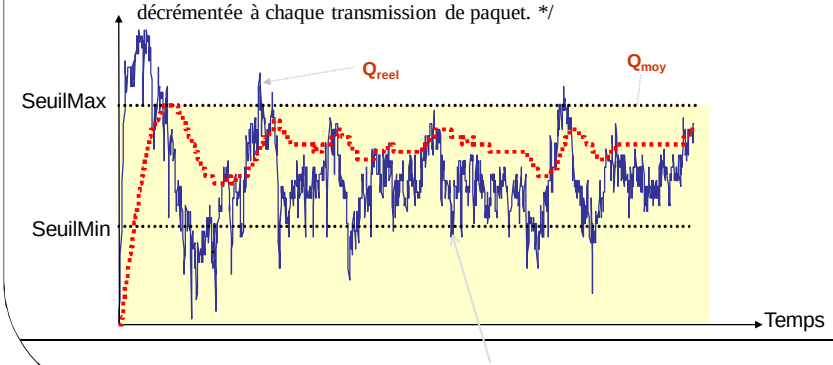
Algorithme de la technique RED (1/2)

➔ Pour chaque paquet arrivant

- Estimation d'une taille moyenne de queue Q_{moy} :

$$Q_{moy} = (1-w)Q_{moy} + w*Q_{reel} \qquad w<<1 \ \ (ex.\ w=0.002)$$

/* Q_{reel} : taille réelle de la queue de paquets acceptés et non encore transmis. Cette variable est incrémentée à chaque acceptation de nouveau paquet et décrémentée à chaque transmission de paquet. */



Algorithme de la technique RED (2/2)

- Rejet/marquage probabiliste en fonction de la taille moyenne de queue
  - Si $Q_{moy} < SeuilMin$: pas de rejet/marquage du paquet arrivant
  - Si $Q_{moy} \geq SeuilMax$: rejet/marquage systématique du paques arrivant
  - Si $SeuilMin \leq Q_{moy} < SeuilMax$: rejet/marquage avec une probabilité p_a calculée comme suit :

$p_b = Pmax*(Q_{moy} - SeuilMin)/(SeuilMax - SeuilMin)$

$p_a = p_b/(1 - p_b*compte)$

/ * *Compte* = nombre de paquets reçus depuis le dernier paquet marqué/rejeté pendant que la condition $SeuilMin \leq Q_{moy} < SeuilMax$ reste satisfaite. */

Explication des paramètres de RED

➔ Si l'on admet que l'on peut accepter des rafales de L paquets au maximum, alors le poids w doit respecter :

$$L + 1 + \frac{(1-w)^{L+1} - 1}{w} < SeuilMin$$

Par exemple : $SeuilMin = 5$ et $L = 50$, conduit à choisir $w \leq 0.0042$

La variation de Q_{moy} (i.e. la sensibilité de Q_{moy} aux rafales) dépend de la valeur de w. Si w est très petit alors, Q_{moy} répond lentement aux changements de la queue réelle.

➔ Les valeurs de *SeuilMin* et *SeuilMax* dépendent de la taille moyenne de queue désirée. Cette taille influe sur le délai moyen d'attente en queue. En général, *SeuilMax* est fixé au moins au double de *SeuilMin*.

➔ *Pmax* désigne une borne supérieure pour la probabilité de marquage de paquet. Le nombre de paquets marqués/rejetés avant d'atteindre le seuil de congestion dépend de cette valeur. *Pmax* est fixée selon les besoins (est-ce que l'on veut anticiper de manière forte ou non les situations de congestion). Les auteurs de RED (Floyd et Jacobson) préconisent que la valeur de *Pmax* ne doit pas être supérieure à 0.1.

Algorithme de la technique RED détaillé

```
/* Initialisation */
Q_reel := 0 ; /* Taille -en nombre de paquets- de la queue réelle */
Q_moy := 0 ; /* Taille moyenne estimée de la queue */
Compte := -1 ; /* Nombre de paquets reçus depuis le dernier paquet marqué. Ce nombre
          s'incrémente uniquement quand la condition SeuilMin ≤ Q_moy < SeuilMax  est satisfaite */
s := /* durée de transmission d'un paquet typique */
Répéter indéfiniment :
  Chaque fois que Q_reel passe à 0 alors /* i.e. quand la queue réelle devient vide */
     Q_instant := InstantCourant ; /* Instant lu à partir de l'horloge du système */
        /* Q_instant indique l'instant de début de période d'oisiveté de la queue */
  ||   /* en parallèle avec ce qui suit */

  Pour chaque arrivée de paquet :
     /* calculer la nouvelle valeur de taille moyenne de queue */
     Si Q_reel > 0 alors {Q_moy := (1 – w)Q_moy + Q_reel*w ;}
     Sinon { m = f(InstantCourant – Q_instant) ; /* Généralement f(x) = x/s */
             Q_moy := (1-w)^m Q_moy ; }    /* m représente le nombre de paquets qui auraient pu arriver
                                           pendant la période d'inactivité de la queue */
     FinSi
     Si SeuilMin ≤ Q_moy < SeuilMax alors
               { Compte := Compte + 1 ;
               p_b := Pmax*(Q_moy - SeuilMin)/(SeuilMax - SeuilMin) ; p_a := p_b/(1 – Compte*p_b) ;
               Avec une probabilité p_a { Rejeter le paquet arrivant ; Compte := 0 ; }
     Sinon
               Si SeuilMax ≤ Q_moy alors {Rejeter le paquet arrivant ; Compte := 0 ; }
               Sinon { Accepter le paquet ; Compte := -1 ; }
               FinSi
     FinSi
   Fin Pour
Fin Répéter
```


4. Technique RED

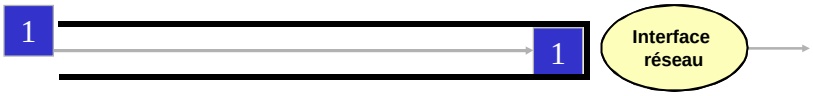
Exemple de fonctionnement de RED (1/6)

SeuilMin = *2*
SeuilMax = *7*
Pmax = *0.1*
w = *1/4*

On suppose qu'il y a une rafale et qu'il n'y a pas de transmission de paquets acceptés

Compte = -1 Q_{moy} = 0 Q_{reel} = 0

Queue vide à l'initialisation

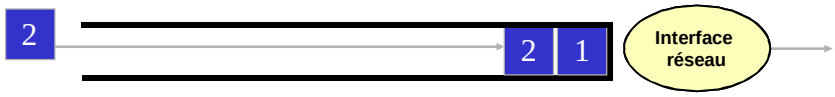


Compte = -1 Q_{moy} = 0 Q_{reel} = 1

Mise en queue d'un paquet

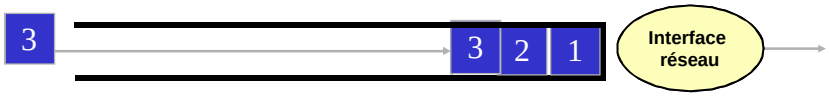
4. Technique RED

Exemple de fonctionnement de RED (2/6)



Compte = -1 Q_{moy} = 0.25 Q_{reel} = 2

Après le 2ème paquet

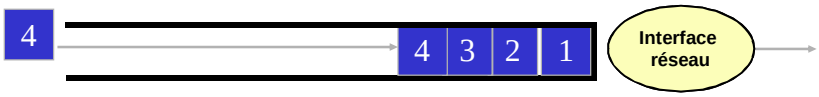


Compte = -1 Q_{moy} = 0.69 Q_{reel} = 3

Après le 3ème paquet

4. Technique RED

Exemple de fonctionnement de RED (3/6)



Compte = -1 Q_{moy} = 1.27 Q_{reel} = 4

Après le 4ème paquet

Compte = -1 Q_{moy} = 1.95 Q_{reel} = 5

Après le 5ème paquet

4. Technique RED

Exemple de fonctionnement de RED (4/6)

Compte = 0 Q_{moy} = 2.71 Q_{reel} = 6

Le *SeuilMin* est dépassé, on calcule $p_a = 0{,}014$. On tire une valeur aléatoire pour savoir si on rejette ou non le paquet. On suppose ici que l'on ne rejette pas le paquet.

Après le 6ème paquet

Compte = 1 Q_{moy} = 3.53 Q_{reel} = 6

Le *SeuilMin* est dépassé, on calcule $p_a = 0{,}03$. On tire une valeur aléatoire pour savoir si on rejette ou non le paquet. On suppose ici que l'on ne rejette pas le paquet.

Après le 7ème paquet

4. Technique RED

Exemple de fonctionnement de RED (5/6)

Compte = 2 Q_{moy} = 4.40 Q_{reel} = 8

Le *SeuilMin* est dépassé, on calcule $p_a = 0{,}05$. On tire une valeur aléatoire pour savoir si on rejette ou non le paquet. On suppose ici que l'on ne rejette pas le paquet.

Après le 8ème paquet

Compte = 3 Q_{moy} = 5.30 Q_{reel} = 9

Le *SeuilMin* est dépassé, on calcule $p_a = 0{,}087$. On tire une valeur aléatoire pour savoir si on rejette ou non le paquet. On suppose ici que l'on ne rejette pas le paquet.

Après le 9ème paquet

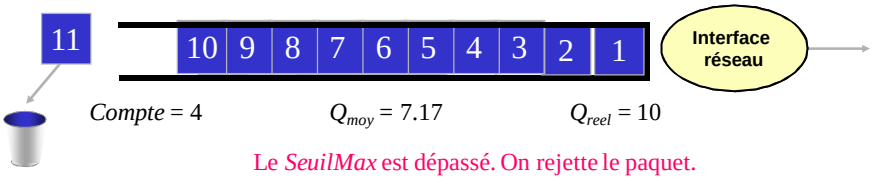
4. Technique RED

Exemple de fonctionnement de RED (6/6)

Compte = 4 Q_{moy} = 6.23 Q_{reel} = 10

Le *SeuilMin* est dépassé, on calcule $p_a = 0{,}127$. On tire une valeur aléatoire pour savoir si on rejette ou non le paquet. On suppose ici que l'on ne rejette pas le paquet.

Après le 10ème paquet



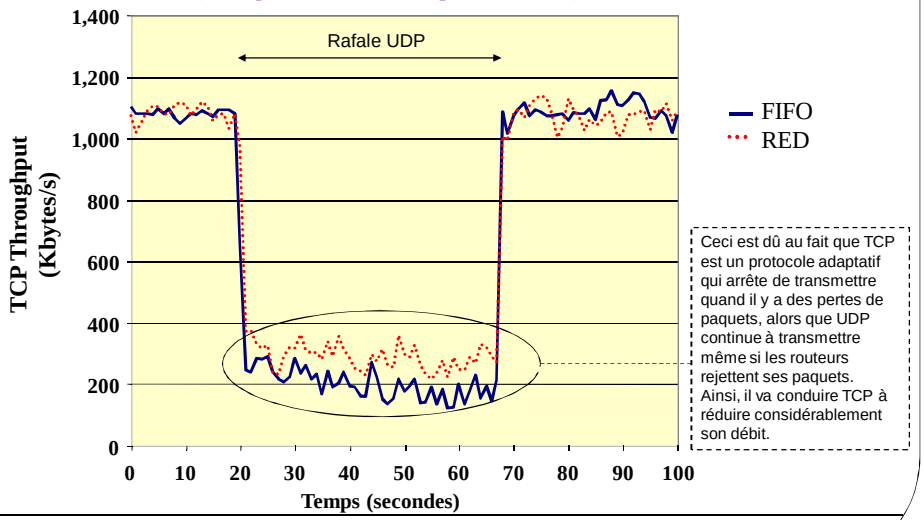
Compte = 4 Q_{moy} = 7.17 Q_{reel} = 10

Le *SeuilMax* est dépassé. On rejette le paquet.

Après le 11ème paquet

4. Technique RED

RED est vulnérable en présence de flux agressifs (exemple de résultat d'expérimentation)



Ceci est dû au fait que TCP est un protocole adaptatif qui arrête de transmettre quand il y a des pertes de paquets, alors que UDP continue à transmettre même si les routeurs rejettent ses paquets. Ainsi, il va conduire TCP à réduire considérablement son débit.

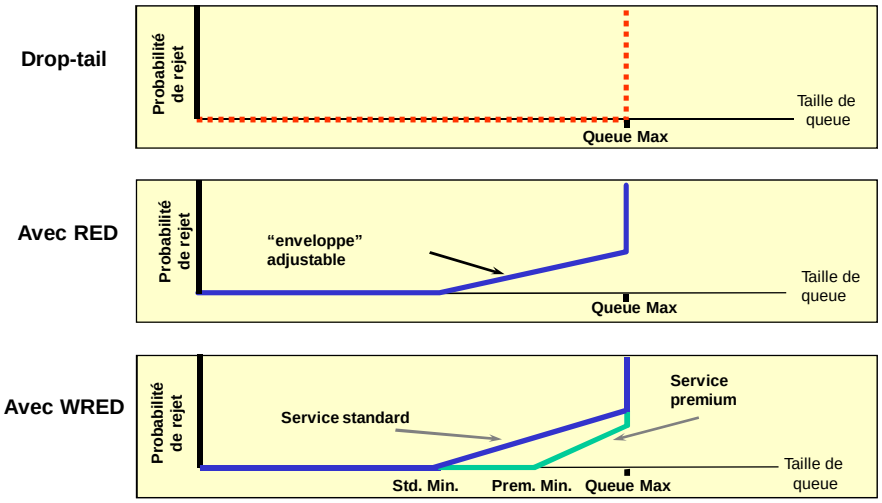
Limites (inconvénients de RED)

- Perturbation par les flux qui se comportent mal
- Sensible au nombre de sources
- Difficultés de choisir les paramètres (seuils et Pmax)
- ...

Variantes de RED

- FRED (Flow RED)
- CBT RED (Class Based Threshold RED) : analogue à FRED
- SRED (Stabilized RED)
- DRED (Dynamic RED)
- WRED (Weighted RED)
- ARED (Adaptative RED)

Rejet selon Drop-tail, RED, WRED

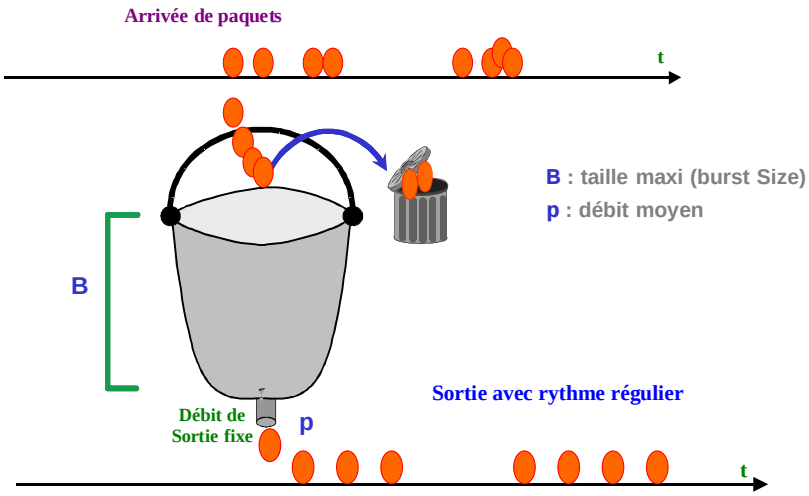


5. Contrôle de trafic utilisateur

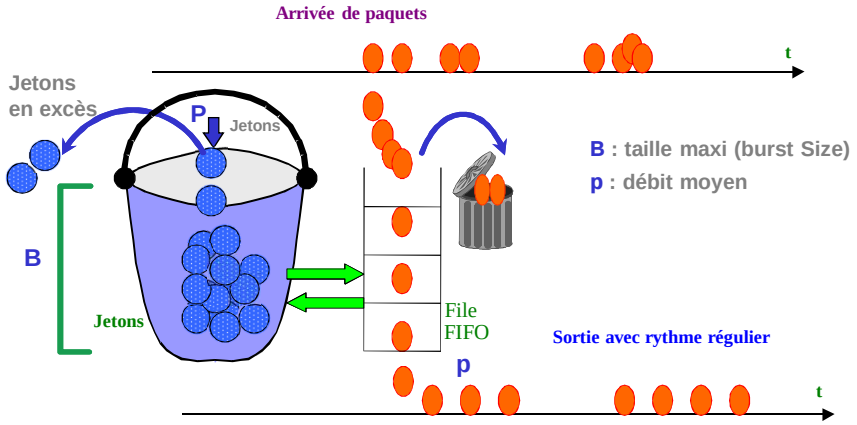
- Protection du réseau contre les abus des utilisateurs
- Contrôle de conformité du trafic de l'utilisateur
- **Deux techniques d'implantation du contrôle du trafic**
  - Seau percé (« leacky bucket »)
  - Seau à jetons (« token bucket »)

5. Contrôle de trafic utilisateur

Contrôle du trafic de l'utilisateur – Seau percé ('Leacky Bucket')



Contrôle du trafic de l'utilisateur – Seau à jetons ('Token bucket')



Mise en œuvre du seau percé pour ATM (1/2)

– ATM véhicule des cellules de taille fixe égale à *53* octets.

– Les cellules ATM sont déposées dans un tampon à *M* places au maximum (*M* correspond au paramètre *MBS* – 'Maximum Busrt Size').

– Les cellules sont vidées du buffer avec un débit constant de *S* cellules/s (*S* correspond au paramètre *SCR* – 'Sustainable Cell Rate').

– Dans le cas où certaines rafales contiennent trop de cellules pour tenir dans le tampon, il y a rejet de certaines cellules (on parle dans ce cas de débordement du seau).

– Un seau est défini par deux paramètres :
I (incrément) : $I = 1/S$
L (Limite) : $L = (M-1)*(1/S)$ $M \geq 1$

Mise en œuvre du seau percé pour ATM (2/2)

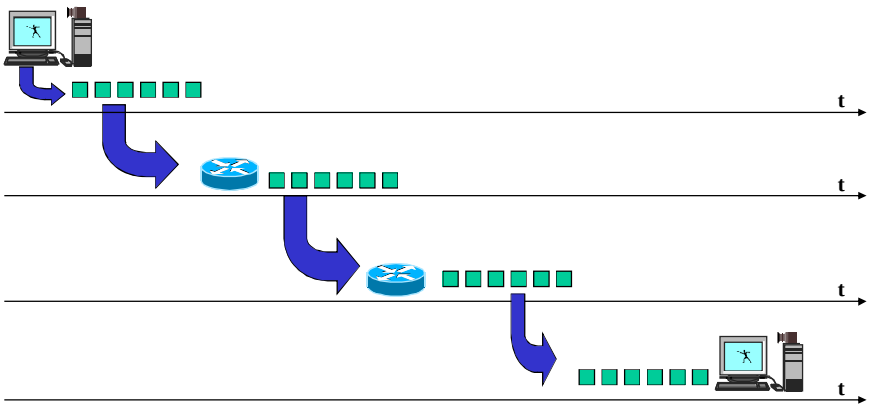
Notations :

- TTA : temps théorique d'arrivée (c-à-d temps d'arrivée normalement prévu)
- ta(k) : temps d'arrivée effectif de la k^{ème} cellule
- I : Incrément
- L : Limite

```
TTA = ta(1) + I   /* TTA initialisé à l'instant de l'arrivée de la
                     première cellule */
Répéter indéfiniment
    Arrivée de la cellule k (k>1)
    Si TTA < ta(k)
            alors TTA = ta(k) + I
            Sinon   Si TTA > ta(k) + L
                            Alors cellule non conforme
                            Sinon TTA = TTA + I
                    FinSi
    FinSi
FinRépéter
```

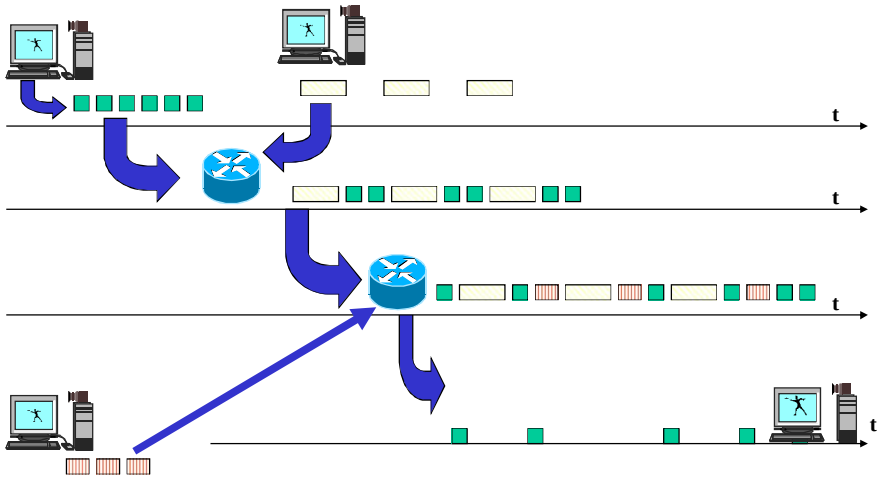

Façonnage de trafic (« traffic shaping ») (1/3)

- Arrivée avec rythme régulier



Façonnage de trafic (2/3)

- Arrivée avec rythme irrégulier



Façonnage de trafic (3/3)

➔**Objectif :** Contrôle du rythme de la circulation des informations entre liens
- Eviter la congestion
- Absorber la gigue

➔**Deux techniques d'implantation du façonnage du trafic**
- Seau percé
- Seau à jetons

➔**Problèmes**
- Surcoût du « traffic shaping » : maintient d'info d'état sur les flux
- Où placer les « shapers » ?
- Agrégation de flux et traffic shaping

6. Conclusion

- Gestion de ressources et contrôle de congestion : fonctions vitales pour la QoS
- Internet actuel : techniques utilisées simples, mais vulnérables

Des problèmes à résoudre

- Utilisation des techniques de prévention
  - Choix des indicateurs de charge et leur mesure
  - Choix des seuils de congestion (statique, dynamique)
  - Agrégation et dégradation de QoS pour les flux individuels
  - Maîtrise de l'analyse statistique de la charge dynamique
- Gestion efficace de buffers pour les flux agrégés
- Techniques d'isolation de flux et pénalisation des flux agressifs
- Applications adaptatives à l'écoute du réseau
- Gestion de buffers adaptive en fonction des applications (différenciation)
- Combinaison : CA/Ordonnancement/Contrôle de congestion

Chapitre 7

Architectures et protocoles de qualité de service dans Internet

1. Introduction

➔ **Objectifs 'initiaux' de l'Internet**

- Fournir les moyens de connectivité globale (au niveau mondiale)
- Permettre à toute personne possédant un ordinateur de communiquer avec les autres
- Traiter les paquets aussi vite que possible

➔ **Aujourd'hui : Internet n'offre quasiment qu'un service *best effort***

➔ **Internet et la QoS**

- Internet doit évoluer pour offrir de la QoS
- Mais toute extension doit garder le sacro saint Best effort

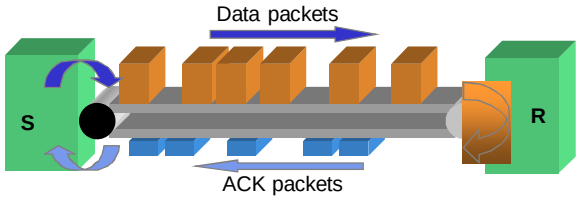
1. Introduction

Internet aujourd'hui

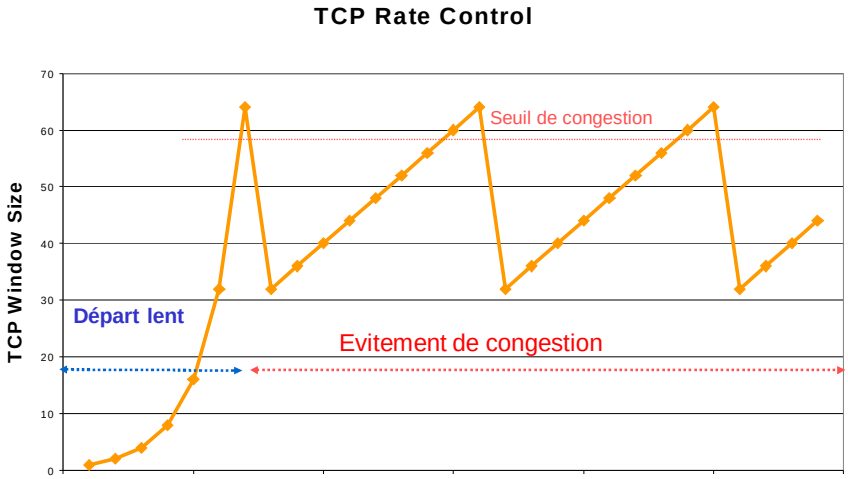
- Trois familles de protocoles
  - TCP (Transmission Control Protocol) en 1981
  - UDP (User Datagram Protocol) en 1981
  - DCCP (Datagram Congestion Control Protocol) en 2006
- Trois applications courantes
  - WWW (TCP)
  - Transfert de données (TCP)
  - Transfert d'audio & vidéo (UDP)
- Consomment 90% de l'Internet actuel

Performance de TCP (1/2)

- Principe de contrôle de flux :
  - Envoyer des segments de données et attendre les ACK
  - Augmenter le débit si les ACK reviennent à temps
  - Réduire le débit en cas de perte des ACK.



Performance de TCP (2/2)



1. Introduction

➔ IETF (Internet Engineering Task Force) a proposé de multiples moyens pour prendre en compte les besoins de QoS

➔ **Deux approches de fourniture de QoS**

- Integrated Services (IntServ) [RFC 1633 - Juin 1994]
  Orienté connexion
- Differentiated Services (DiffServ) [RFC 2475 - Décembre 1998]
  Orienté agrégation de flux et marquage de paquets

➔ **Protocoles pour la QoS**

- RSVP (Resource reSerVation Protocol) [RFC 2205 – Septembre 1997]
- MPLS (MultiProtocol Label Switching) [RFC 3031 – Janvier 2001]
- **Autres**

2. Architecture IntServ

IntServ (Integrated Service) (1/5)

➔ Flux de données = paquets ayant la même source et destination et même port

➔ IntServ adopte le modèle orienté connexion

➔ IntServ se base sur RSVP pour la réservation de ressources

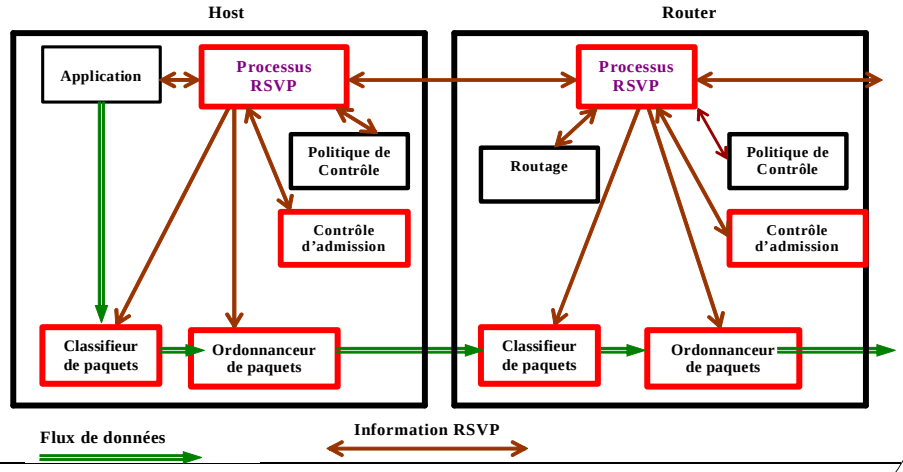
➔ Les routeurs qui n'implantent pas IntServ ignorent les nouvelles classes de QoS

➔ Trois classes de service selon les besoins des applications
- Service garanti (garantie absolue)
- Service à charge contrôlée (garantie statistique)
- Services Best-effort de trois types (pour le Web, FTP, Mail, etc.)

IntServ (Integrated Service) (2/5)

➔Implantation de IntServ : 4 composants

Protocole de signalisation, Contrôle d'admission, Classifieur, Ordonnanceur de paquets



IntServ (3/5)

➔ **État actuel** : Deux services IntServ ont été standardisés

- Guaranteed Service (RFC 2212 - septembre 1997)

  Il fournit une garantie absolue pour le délai de transfert et la perte de paquet.

- Controlled Load Service (RFC 2211 – septembre 1997)

  Il fournit un service équivalant à celui d'un réseau sous-chargé.

  Il est utilisable surtout par les applications qui s'adaptent à la charge du réseau et qui tolèrent la perte de paquet.

IntServ (4/5)

➔Paramètres de caractérisation de QoS et trafic [RFC 2215 – 1997]

- `NON-IS-HOP` : indique si, sur le chemin du flux, il y a des nœuds qui n'implantent pas IntServ.
- `NUMBER_OF_IS_HOPS` : compte le nombre de nœuds qui implantent IntServ sur le chemin du flux.
- `AVAILABLE_PATH_BANDWIDTH` : fournit l'information sur la bande passante disponible sur le chemin du flux.
- `MINIMUM_PATH_LATENCY` : fournit la latence minimale (i.e. avec délai de séjour en file d'attente -*queuing delay*- égal 0) pour les nœuds traversés.
- `PATH_MTU` : fournit la *Minimum Transmission Unit* sur le chemin du flux.
- `TOCKEN_BUCKET_TSPEC` : décrit les caractéristiques du trafic par le débit du seau percé (*r*), la capacité maximum du seau (*b*), le débit de crête (*p*), la taille minimum de paquet (*m*) et la taille maximum de paquet (*M*).

IntServ (5/5)

➔Limites/inconvénients

- Le nombre de flux individuels peut être très important. Par conséquent, le nombre de messages de contrôle peut être élevé et nécessite beaucoup de ressources au niveau de chaque routeur.
- Des politiques doivent être mises en place pour déterminer quand, où et pour qui les ressources peuvent être réservées.
- Des règles de sécurité doivent être mises en place pour garantir que les utilisateurs non autorisés ne peuvent pas effectuer des réservations de ressources.
- Peu d'industriels ont implanté IntServ à grande échelle.

➔ IntServ : convient seulement aux petits réseaux (problème de passage à l'échelle)

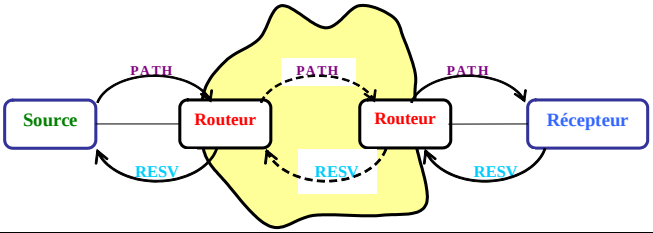
ReSerVation Protocol (RSVP) RFC 2205 (1/5)

➔ **C'est quoi RSVP?**

- RSVP est un protocole de signalisation pour demander la réservation de ressources dans un réseau IP.
- Principales caractéristiques de RSVP
  - C'est l'application qui initie le processus de réservation (granularité fine de réservation) au moment du démarrage d'un flux.
  - Modèle de réservation orienté Récepteur
  - Les réservations sont faites pour chaque flux individuel
  - Les flux peuvent être *unicast* ou *multicast*
  - Il supporte les réservations hétérogènes et permet la renégociation de réservation
  - Il permet un bon partage des ressources réservées pour de multiple flux.
  - La gestion des réservations s'effectue en mode état *soft*.

RSVP (2/5)

- Un chemin *unicast* ou *multicast* est déterminé par un algorithme de routage (sans être sûr que ce chemin réponde à la QoS exigée).
- La source du flux transmet un message PATH pour indiquer les caractéristiques de son flux. Chaque routeur traversé garde la trace du flux et des ressources demandées.
- Chaque récepteur a ses propres capacités et spécifie dans un message RESV ses exigences. Chaque routeur sur le chemin du message RESV confirme les réservations ou les annule. Quand le message RESV atteint la source, les réservations sont acceptées le long du chemin et le flux de données peut commencer.



RSVP (3/5)

➔ **État *soft* de RSVP**

- Signifie l'état associé à RSVP (états PATH et RESV) est temporaire
  - Il faut le rafraîchir pour qu'il reste valide
  - Il est (éventuellement) détruit s'il n'est pas rafraîchi.
- Rafraîchissement et suppression d'état
  - Par envoi périodique de messages (PATH/RESV), toutes les *t* sec.
  - Si un routeur ne reçoit pas de message après *n*t* sec, il supprime l'état.
- Comme on est dans IP et pour éviter le surcoût de gestion des états, les messages sont envoyés de manière non fiable.

☞ Inconvénients

La maintenance des états soft implique que tous les routeurs doivent constamment contrôler et mettre à jour les états pour chaque flux individuel. La conséquence peut être une congestion de réseau.

RSVP (4/5)

➔ **7 Messages RSVP**

- PATH : Mise en place d'un état de réservation le long d'un chemin.
- RESV : Requête de réservation le long d'un chemin.

```
<Path Message> ::= <Common Header>
[ <INTEGRITY> ] <SESSION> <RSVP_HOP>
<TIME_VALUES> [ <POLICY_DATA> ... ]
[ <SENDER_TEMPLATE> <SENDER_TSPEC>
[ <ADSPEC> ] ]
```

Définit les caractéristiques du trafic.

```
<Resv Message> ::= <Common Header>
[ <INTEGRITY> ] <SESSION> <RSVP_HOP>
<TIME_VALUES> [ <RESV_CONFIRM> ]
[ <SCOPE> ] [ <POLICY_DATA> ... ]
<STYLE> <flow descriptor list>
```

Définit la QoS demandée par le récepteur.

☞ RSVP est un protocole général et ne spécifie ni les types de ressources demandées, ni la quantité de ressources à réserver.

RSVP (5/5)

➔ 7 **Messages RSVP (suite)**

- PATH_TEAR : Suppression explicite d'un état de réservation.
- RESV_TEAR : Suppression explicite de réservation.
- PATH_ERR : Message d'erreur sur un chemin.
- RESV_ERR : Message d'erreur de réservation.
- RESV_CONFIRM : Confirmation de réservation

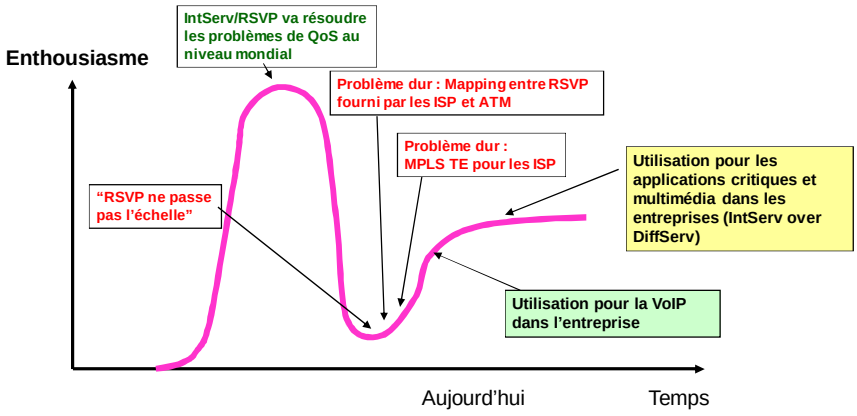
➔ **Extensions RSVP**

- RSVP pour DiffServ, RSVP pour les réseaux ad hoc, RSVP sans état…

➔ **Extension souhaitée non encore acceptée par l'IETF**

- RSVP est orienté Récepteur or dans certaines applications c'est l'émetteur (source) qui souhaite imposer les exigences en terme de QoS pour le flux qu'il veut transmettre.

Acceptation de IntServ/RSVP



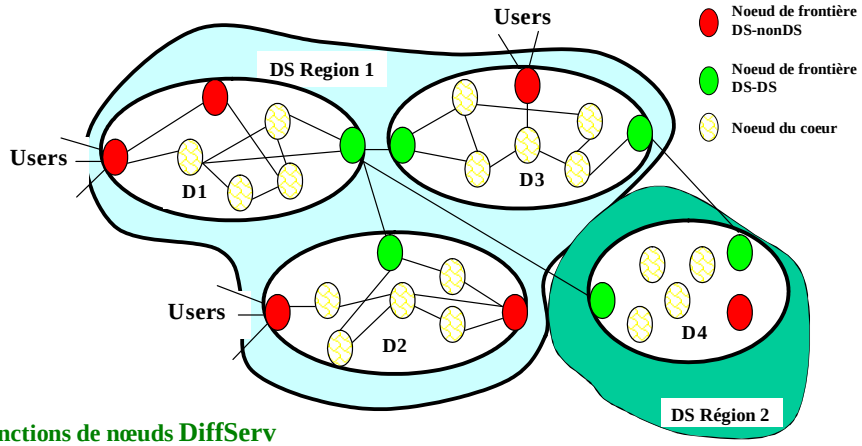
3. Architecture DiffServ

Principes de base et caractéristiques de DiffServ

- Pallier les inconvénients de passage à l'échelle de IntServ
- Introduction du concept d'agrégation de flux pour simplifier les traitements de classification et marquage de paquets
- Caractéristiques importantes
  - Pas de gestion d'état par flux
  - Pas (ou peu) de classification de paquets à l'intérieur du réseau
  - Gestion sur la base de SLA et non de connexion
  - Réalisable par des mécanismes simples (câblés)

3. Architecture DiffServ

Routeurs de bordure et routeurs de cœur

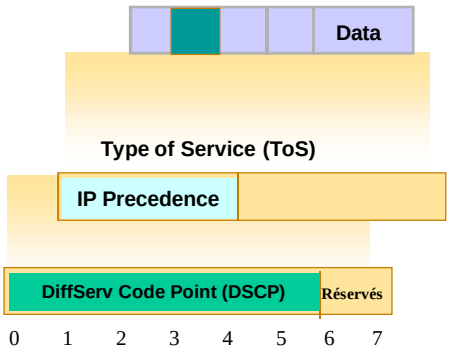


➔ **Fonctions de nœuds DiffServ**

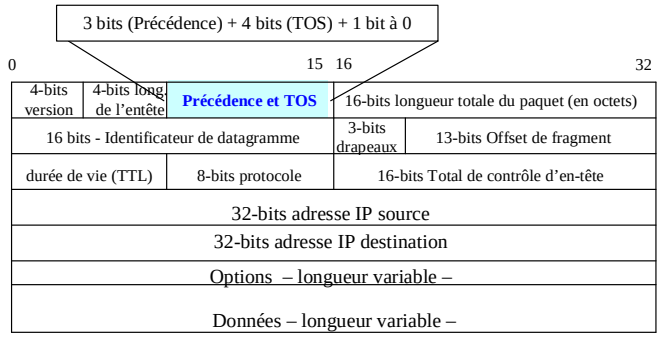
- Nœuds de frontière : Classification, marquage,… destruction de paquets
- Nœuds du cœur : Commutation, accélération des traitements de paquets

Code DiffServ (1/3)

- Champ DiffServ (or DSCP) = 6 bits dans l'entête IPv4 ou IPv6
  Il indique le niveau de QoS pour traiter le paquet



Code DiffServ (2/3)

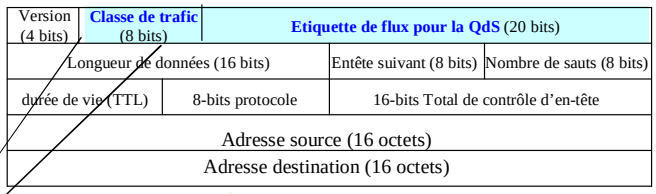


Entête de paquet IPv4
[RFC 791 ; Sep. 1981]

TOS = 1xxx : Minimiser le délai
01xx : Maximiser le rendement
001x : Maximiser la fiabilité
0001 : Minimiser le coût
0000 : Service normal

On ne peut pas exiger deux types de QoS en même temps

Code DiffServ (3/3)



Entête de paquet IPv6
[RFC 2460 – Déc. 1998]

4 bits de priorité/classe

Les priorités de 0 à 7 : pour le trafic dont la source dispose d'un mécanisme de contrôle de congestion : 0 (trafic non caractérisé), 1 (NNTP), 2 (SMTP),

4 (FTP et NFS), 6 (Telnet et terminal X), 7 (SNMP)

Les priorités de 8 à 15 : pour le trafic qui ne prend pas en compte les situations de congestion (trafic temps réel).

Marquage/re-marquage de paquet

- **Marquage**
  - **Par la source**
    * Par un dialogue approprié avant l'échange de données, la source obtient le DSCP
    * Le DSCP est fixé par le réseau au moment de l'établissement du contrat (SLA) et notifié à la source
  - **Par réseau**
    * Lorsqu'un paquet arrive au premier routeur de bordure, celui-ci choisit le DSCP en fonction des adresses source et destination, numéro de port et informations sur le SLA.

- **Re-marquage**
  - Dans les routeurs de coeur (rarement)
  - Lorsque le paquet traverse des systèmes autonomes différents

Traitement standardisé : PHB (Per Hop Behavior) (1/3)

- Agrégation de comportement : un ensemble de paquets avec le même DSCP
- Comportement par saut (PHB): traitement subi (en terme de QoS) par un paquet marqué par un DSCP
- Un PHB est une notion abstraite (boite noire)
  - Un PHB décrit le comportement observable lors de l'expédition (forwarding) de paquets avec le même DSCP
  - Un PHB peut être implanté de manière différente selon les systèmes autonomes
  - L'IETF n'indique ni comment les PHB sont implantés, ni quelle est la nature exacte (QoS) du traitement subi par les paquets.
- Correspondance entre DSCP et PHB : réalisée (connue) de manière statique.
- Région DiffServ = ensemble de domaines DiffServ avec des objectifs communs

Traitement standardisé : PHB (Per Hop Behavior) (2/3)

- Groupes de PHB standardisés
  - *Expedited Forwarding* (RFC 2598 – Juin 1999)
  - *Assured Forwarding* (RFC 2597 – Juin 1999)
  - *Best effort* (RFC 2474 – Décembre 1998) - Code DSCP = '000000'

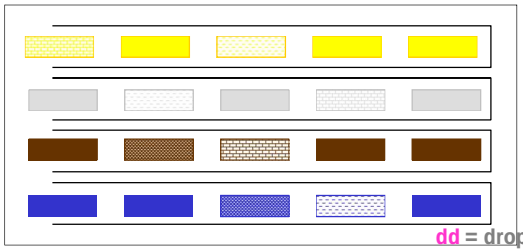
Note : Chaque PHB a un DSCP recommandé par IETF, mais les ISP peuvent choisir d'autres codes pour leurs réseaux (mais attention l'interopérabilité !).

- Expedited Forwarding (DSCP ='101110')
  - Offre un service avec : bande passante garantie, taux de perte faible, faible latence, faible gigue
  - Adapté à la voix sur IP et les lignes virtuelles louées
  - EF fonctionne correctement à condition que le trafic des sources ne dépasse pas une borne maximale connue.

Traitement standardisé : PHB (Per Hop Behavior) (3/3)

- Assured Forwarding
  - Offre différent niveaux de service en termes de débit (*gold, silver, bronze*…)
  - Les contraintes de temps (délai et gigue) non garanties
  - Le taux de perte est non quantifiable mais dépend du DSCP.
  - Actuellement 4 classes AF indépendantes sont définies. 3 niveaux de rejet (*drop precedence*) en cas de congestion sont définis pour chaque classe.

AF Classe 1 > AF Classe 2 > AF Classe 3 > AF Classe 4



DSCP recommandés

AF Classe 1: 001dd0

AF Classe 2: 010dd0

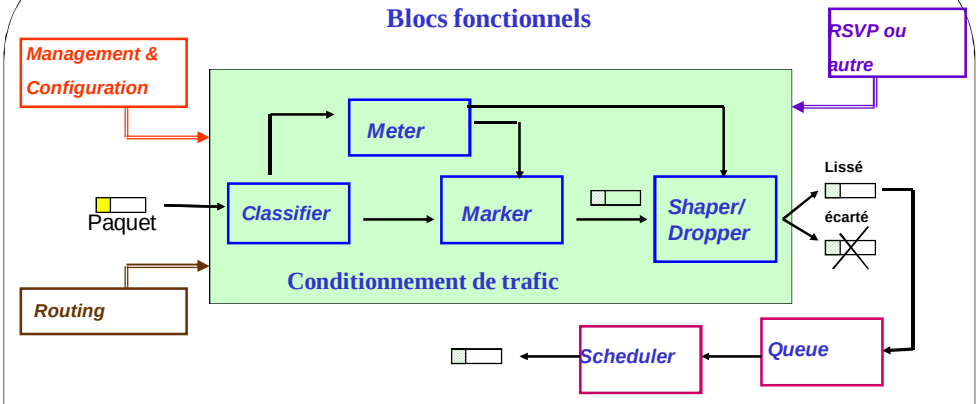
AF Classe 3: 011dd0

AF Classe 4: 100dd0

dd = drop precedence

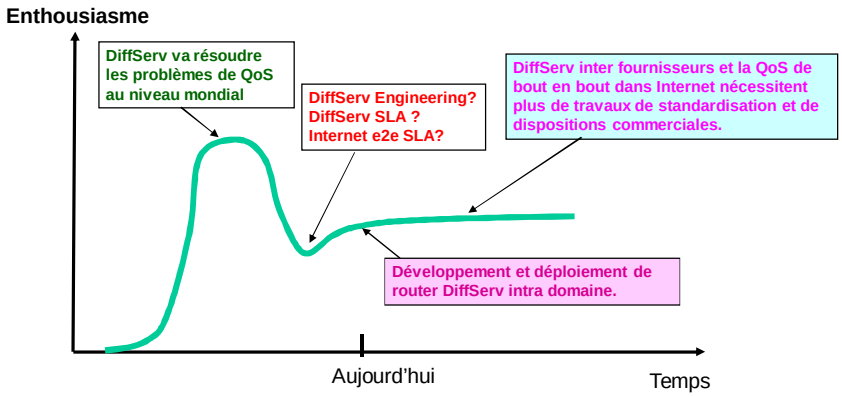
dd= 01 : taux de perte faible ; dd=11 : taux de perte élevé

Blocs fonctionnels



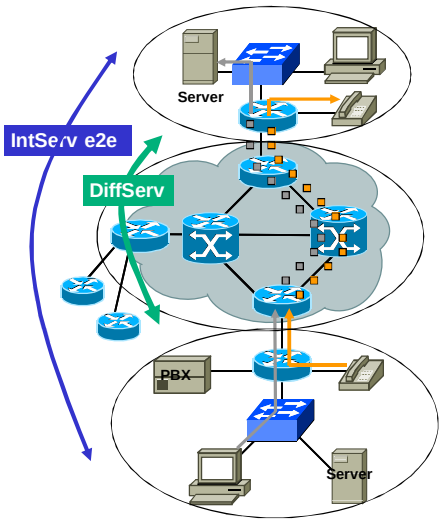
- *Classifier :* sélectionne les paquets en fonction de leur entête.
- *Meter :* effectue des mesures pour savoir si le paquet est conforme au contrat de trafic
- *Marker* : réécrit ou change le DSCP
- *Shaper :* retarde certains paquets pour les rendre conformes à certain rythme.
- *Dropper* : écarte (élimine) certains paquets non conformes ou ayant un taux de rejet exigé plus élevé que celui des autres paquets (seulement en cas de congestion)

Acceptation de DiffServ



'IntServ over DiffServ'

- IETF a défini un cadre pour le déploiement de IntServ au dessus de DiffServ.
- Une solution pour le passage à l'échelle de RSVP dans l'entreprise et SP.
- Mapping flux RSVP et PHBs
- Multiple alternatives pour le contrôle d'admission au niveau des AS DiffServ :
  – Pas de CA
  – CA au niveau des routeurs de bordure
  – CA au niveau de chaque routeur
  – CA au niveau de chaque routeur mais sur des flux agrégés.



Limites de DiffServ

– Fournir de la QoS à des flux agrégés ne permet pas (sauf pour la classe premium) toujours d'offrir la QoS de bout en bout pour chaque flux pris individuellement.

– DiffServ suppose des SLA configurés de manière statique. Or les besoins utilisateur et la topologie du réseau peut évoluer dans le temps.

– DiffServ est orienté Emetteur. Or parfois il faut tenir compte des capacités et besoins du récepteur.

– Le nombre de DSCP n'est pas assez élevé pour différencier réellement les flux. On surdimensionne souvent (eg. Mettre ensemble deux flux qui demandent deux latences de 10 et 50 ms).

Problèmes ouverts sur DiffServ

– Analyse et maîtrise de l'impact de l'agrégation sur les performances de bout en bout.

– Conception et validation de PHB avec une approche orientée composant à QoS

– Mapping entre les besoins en QoS, DSCP et PHB.

– Implantations efficaces des blocs fonctionnels (*scheduling*, *routing*, *meter*, *shaping*…)

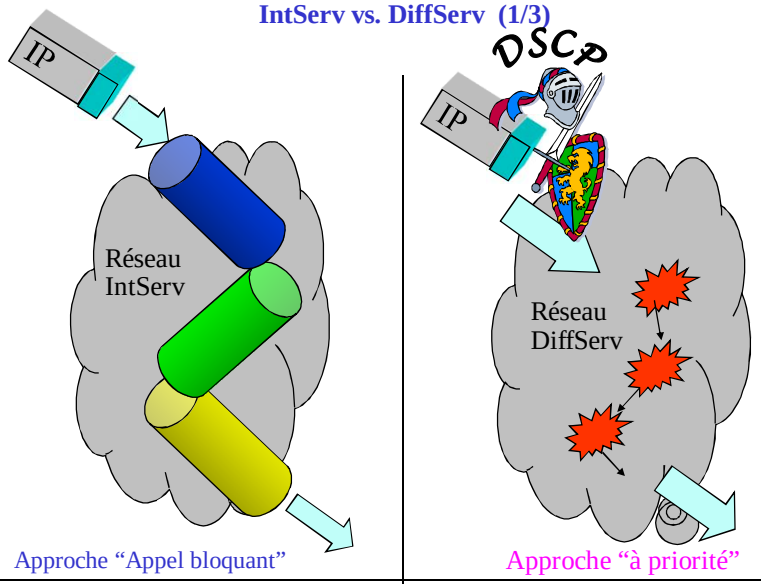
– Réservation de ressources dynamiques : stratégies et signalisation

– DiffServ sur MPLS, GMPLS

– DiffServ et réseaux ad hoc, réseaux de capteurs

– Autres

IntServ vs. DiffServ (1/3)



Approche "Appel bloquant"

Approche "à priorité"

IntServ vs. DiffServ (2/3)

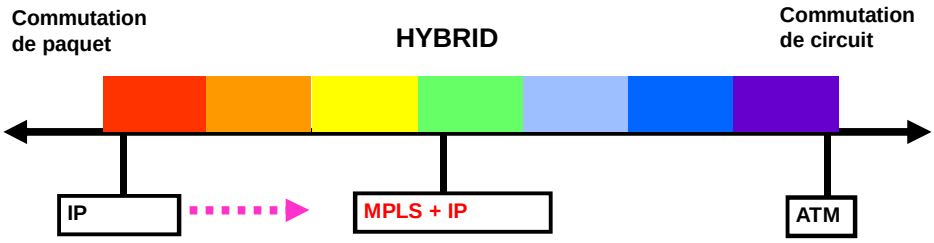
	Intserv	Diffserv
Granularité du service	Flux individuels	Agrégation de flux
Etats dans les routeurs (eg. Ordonnancement…)	Par flux	Par agrégation
Base de classification de trafic	Plusieurs champs dans l'entête	Champ DSCP
Type de différenciation de service	Garantie déterministe ou statistique	Assurance absolue ou relative
Contrôle d'admission	Obligatoire	Requis pour l'assurance absolue
Protocole de signalisation	Obligatoire (RSVP)	Non requis pour l'assurance relative

IntServ vs. DiffServ (3/3)

	Intserv	Diffserv
Coordination pour la différenciation de service	End-to-end	Local (PHB)
Etendue de service	Chemins unicast ou Multicast	Partout dans le réseau
Sclabilité	Limité par le nombre de flux	Limité par le nombre de classes de service
Comptabilité	Basée sur les caractéristiques de flux et exigences de QoS	Basée sur l'utilisation de classe
Gestion de réseau	Similaire aux réseaux à commutation de circuits	Similaire aux réseaux IP
Déploiement interdomaine	Accords multilatéraux	Accords bilatéraux

4. MPLS

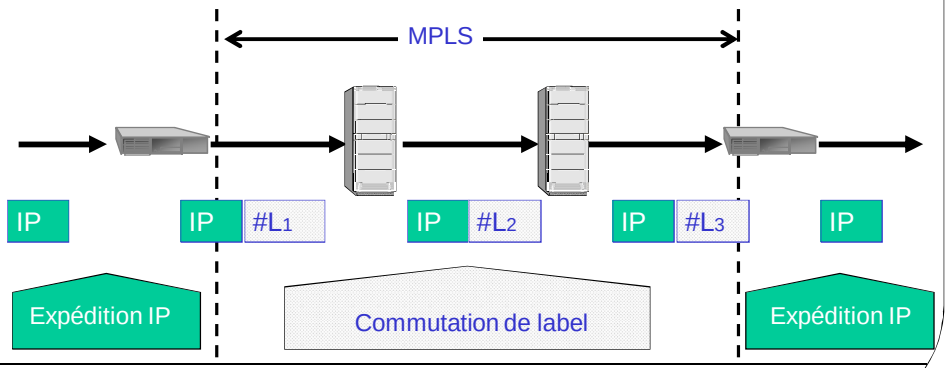
Le meilleur des deux 'mondes'



- **ATM et Frame Relay ne peuvent pas se substituer directement à IP pour être utilisés comme technologies pour l'Internet.**
- **MPLS + IP forme un mécanisme puissant combinant les points forts des deux technologies de commutation (circuit et paquets IP).**
- **Objectifs de MPLS : permettre une expédition (forwarding) ultra rapide afin d'offrir de la QoS.**

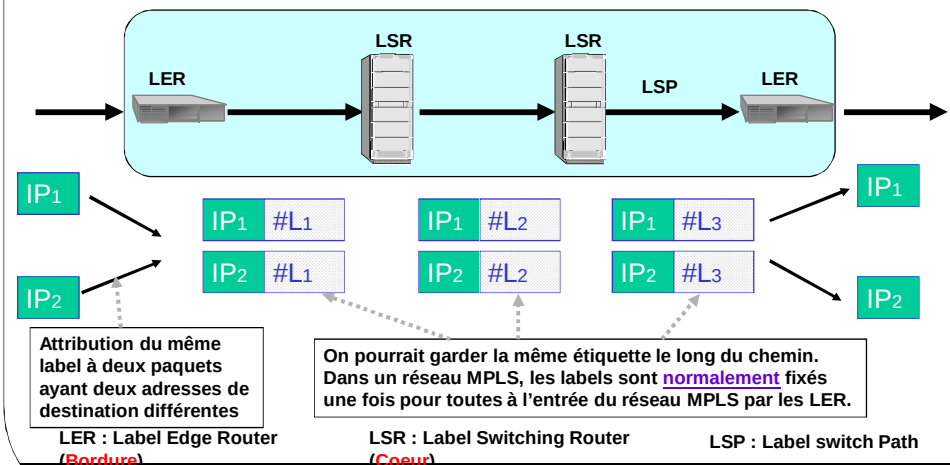
Terminologie MPLS (1/3)

- **MPLS = MultiProtocol Label Switching**
- **Label = un tag ('étiquette') permettant d'identifier un flux (FEC).**
- Expédition de paquet (bordure) + Commutation de label (cœur)



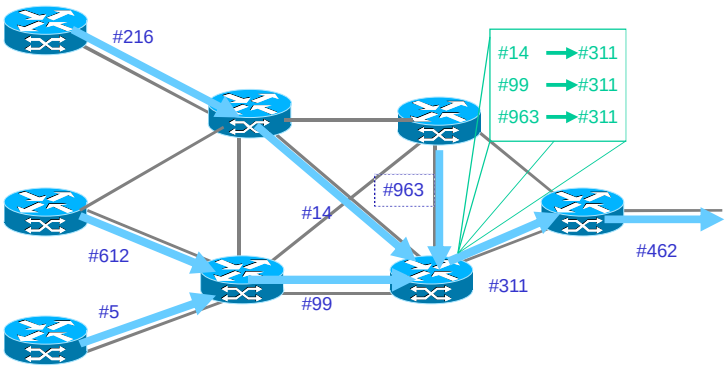
Terminologie MPLS (2/3)

- **FEC (Forwarding Equivalence Class) : groupe de paquets IP subissant le même traitement d'expédition. Il y a une correspondance entre FEC et classe de QoS fournie.**



Terminologie MPLS (3/3)

■ **LSP (Label Switching Path) : unicast ou multicast.**



Exemple de LSP multicast

Commutation dans MPLS (1/2)

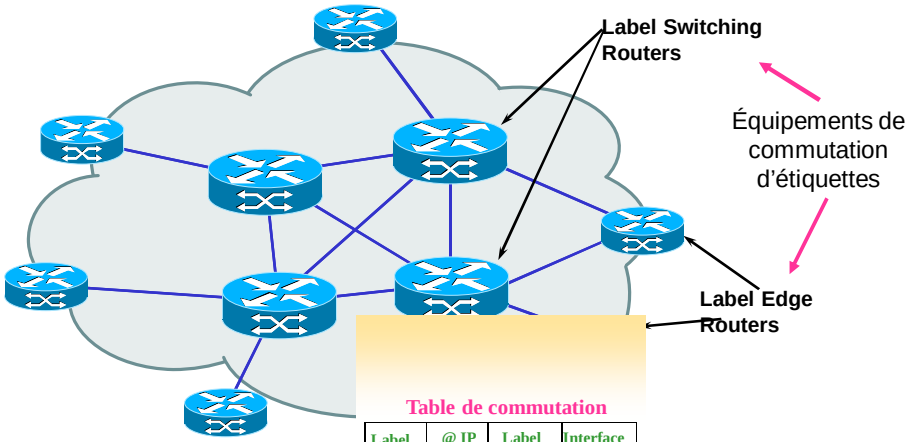


Table de commutation

Label entrant	@ IP dest	Label sortant	Interface de sortie

Commutation dans MPLS (2/2)

- Principes de base de la commutation de label :
  - Combine le routage de niveau Réseau avec l'expédition basée sur l'échange de label
    - La simplicité de l'expédition de la couche 2 offre de meilleures performances
    - Le routage de niveau 3 a une meilleure scalability.
  - Séparation nette entre l'expédition et le contrôle/routage
    - Module d'expédition : une simple fonction d'échange de label.
    - Module de contrôle : fonctions de collection des données pour la maintenance et distribution des labels.
    - La séparation facilite l'évolution graduelle du contrôle simplicité.

- Algorithme d'expédition :
  - Extraire le label du paquet entrant
  - Trouver l'entrée dans la table de commutation avec "label entrant = label du paquet"
  - Remplacer le label du paquet avec le label sortant
  - Envoyer le paquet sur l'interface de sortie.

- Algorithme d'expédition indépendant de la couche Réseau et de la distribution des labels.

Distribution des labels (1/3)

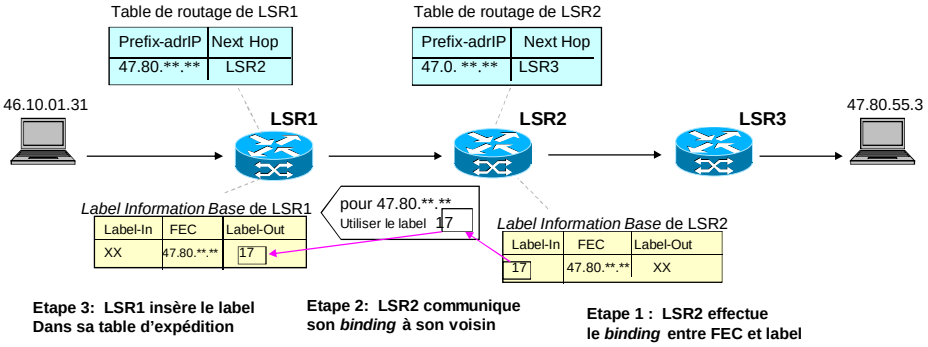
- Objectif
  - S'assurer que des routeurs voisins ont une vue commune des FEC pour traiter les paquets de manière cohérente.
  - Permettre à chaque routeur de trouver la correspondance entre le numéro de label entrant et le numéro de label sortant

- Techniques de distribution de labels
  - Manuellement
  - Par LDP (Label Distribution Protocol) dédié
  - Par *Constraint-based routing* LDP /* Pour tenir compte de la QoS */
  - Par RSVP-TE (RSVP avec extension TE –traffic engineering)
  - Par BGP (Border Gatway Protocol)
  - Autres

- Protocole de distribution de label dits aussi Protocoles de signalisation MPLS

Distribution des labels (2/3)

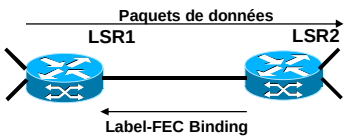
- Avant de fixer les labels et de les communiquer entre routeurs MPLS, les tables de routage de ces routeurs doivent être construites (par OSFP, RIP…)
- Chaque LSR associe (binding) un label à chaque FEC (décision locale). Ensuite les labels sont communiquées aux autres LSR en prenant le chemin inverse des données.



Distribution des labels (3/3)

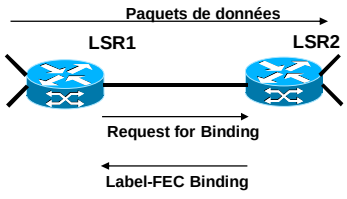
- Deux manières de distribuer les labels par LDP : à la demande ou sans demande

LDP en mode sans demande



- LSR2 et LSR1 ont une relation d'adjacence-LDP si LSR2 est le successeur de LSR1 sur le chemin pour une FEC donnée.
- LSR2 découvre le next-hop pour le FEC (s'il n'y a pas de next-hop, LSR2 ne pourra pas traiter les paquets de la FEC car ils n'ont pas de destination).
- LSR2 choisit un label qu'il associe à la FEC et le communique à LSR1.
- LSR1 insère le label reçu dans sa table.

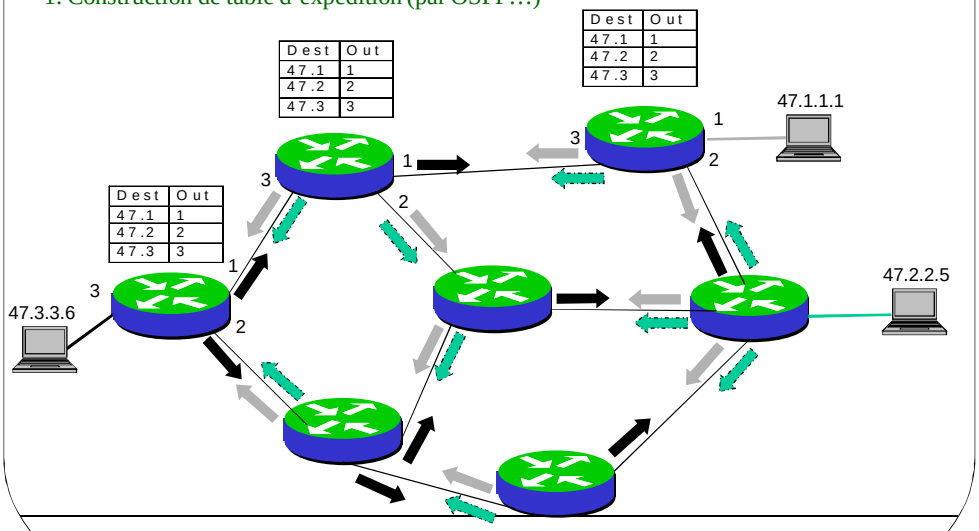
LDP en mode avec demande



- LSR1 reconnaît LSR2 comme étant le prochain saut pour une FEC donnée.
- LSR1 envoie une requête à LSR2 pour obtenir son *binding*.
- Si LSR2 reconnaît le FEC et qu'il a un next-hop pour elle, il crée le binding et répond à LSR1.
- Les deux LSRs connaissent les labels à utiliser.

Exemple de distribution des labels (1/3)

1. Construction de table d'expédition (par OSPF…)



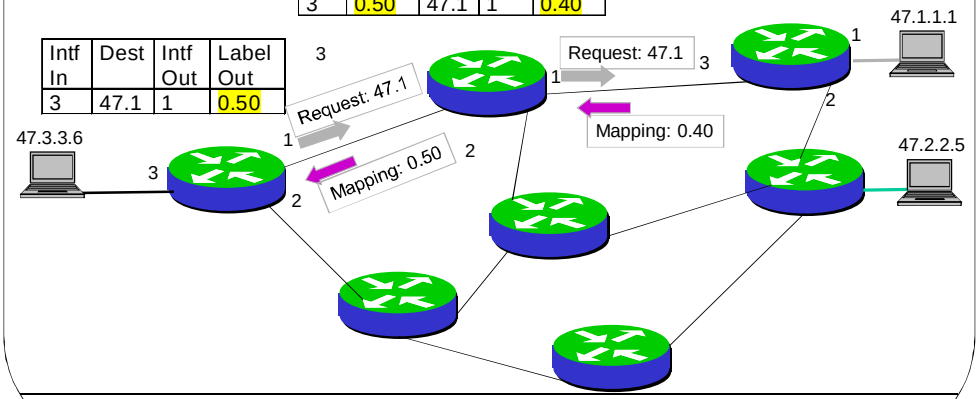
Exemple de distribution des labels (2/3)

2. Distribution de label

Intf In	Label In	Dest	Intf Out	Label Out
3	0.50	47.1	1	0.40

Intf In	Label In	Dest	Intf Out
3	0.40	47.1	1

Intf In	Dest	Intf Out	Label Out
3	47.1	1	0.50



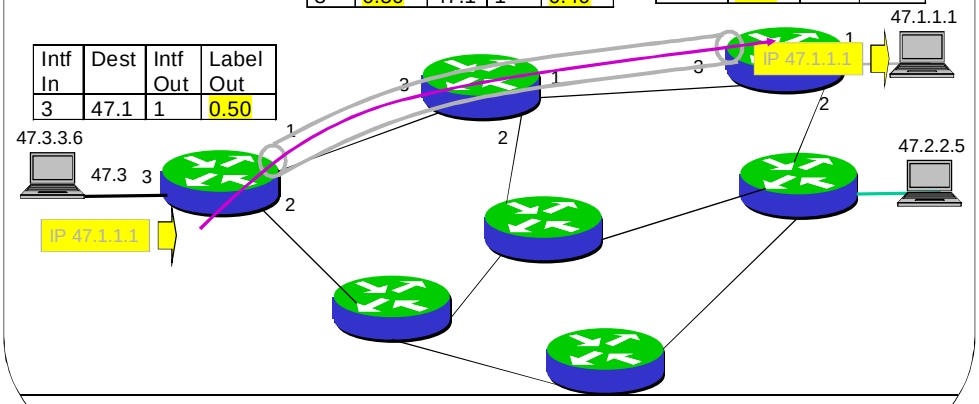
Exemple de distribution des labels (3/3)

3. Les paquets empruntent le LSP

Intf In	Label In	Dest	Intf Out	Label Out
3	0.50	47.1	1	0.40

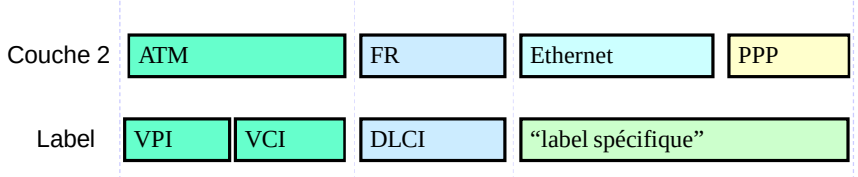
Intf In	Label In	Dest	Intf Out
3	0.40	47.1	1

Intf In	Dest	Intf Out	Label Out
3	47.1	1	0.50



Encapsulation de label (1/2)

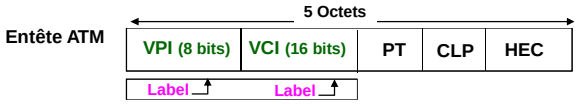
- MPLS est censé fonctionner au dessus de diverses couches 2
- Les spécifications pour les couches 2 suivantes sont définies:
  - ATM : label contenu dans le champ VCI/VPI de l'entête de cellule ATM
  - Frame Relay : label contenu dans le champ DLCI dans l'entête de cellule FR
  - PPP/LAN : utilise un entête inséré entre les entêtes des couches 2 et 3



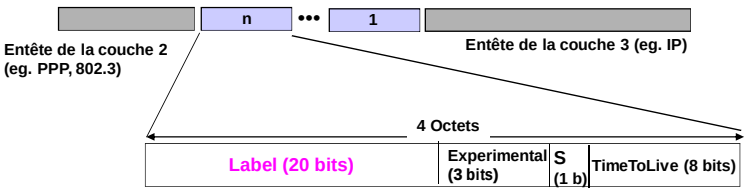
VPI: virtual path identifier DLCI: data link connection identifier
VCI: virtual channel identifier

Encapsulation de label (2/2)

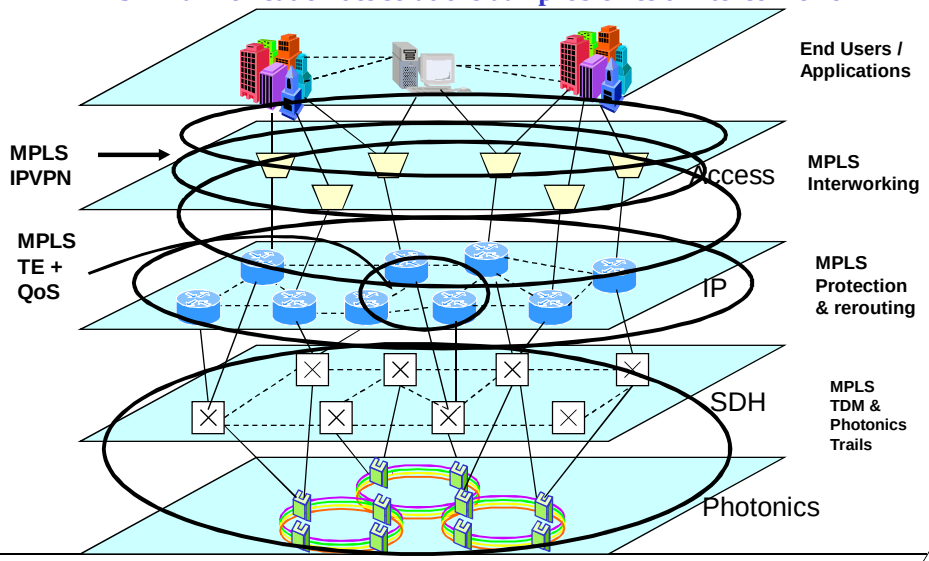
- Encapsulation pour ATM



- Encapsulation pour PPP, Ethernet et autres couches 2 de LAN



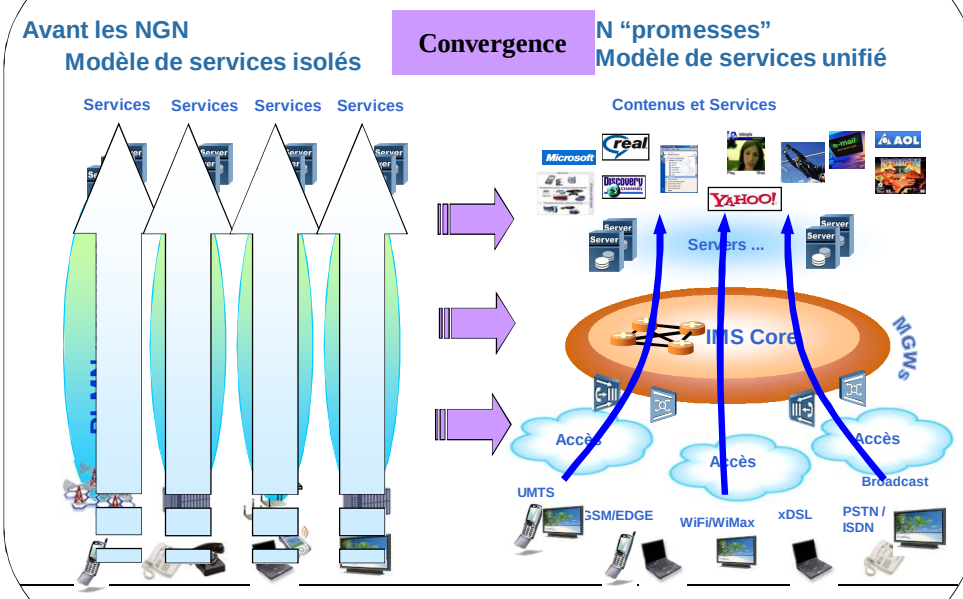
MPLS = Harmonisation des solutions aux problèmes d'interconnexion

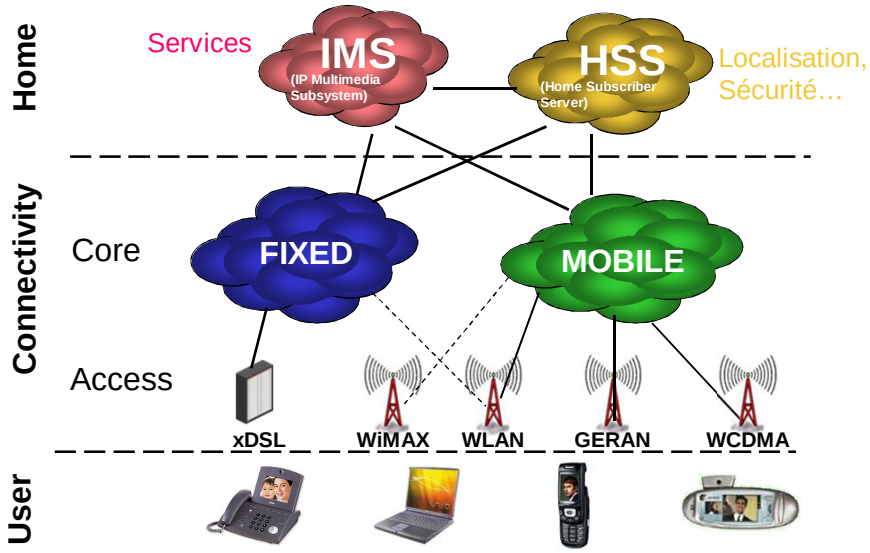


5. Conclusion

- Internet à QoS devient de plus en plus une réalité
  - Les utilisateurs la demandent
  - La technologie permet de l'offrir.
  - Les standards sont disponibles.
- Mais le futur est un peu incertain/trouble
  - De multiple solutions concurrentes
    - + DiffServ vs. IntServ
    - + DiffServ vs. MPLS + traffic engineering
    - + DiffServ over GMPLS
  - Des pièces manquantes (ou à développer davantage)
    - + Signalisation DiffServ et définition des services et SLA
    - + Couplage entre les différentes architectures et protocoles de routage
    - + Interactions entre technologies
    - + Passage à l'échelle des solutions proposées
    - + Politique standards
    - + Interactions entre fournisseurs, entre AS
    - + …

5. Conclusion





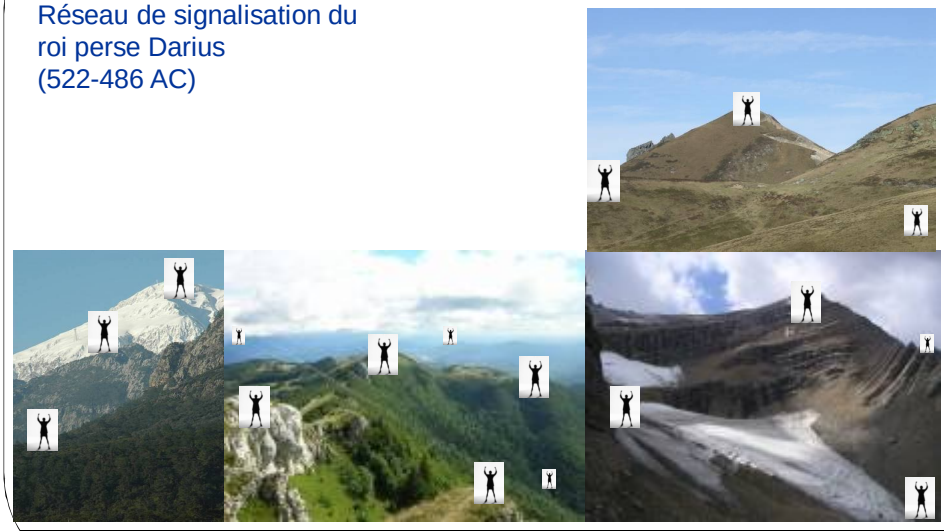
Chapitre 8

Routage à QoS dans les réseaux ad hoc mobiles

1. Introduction aux MANETs

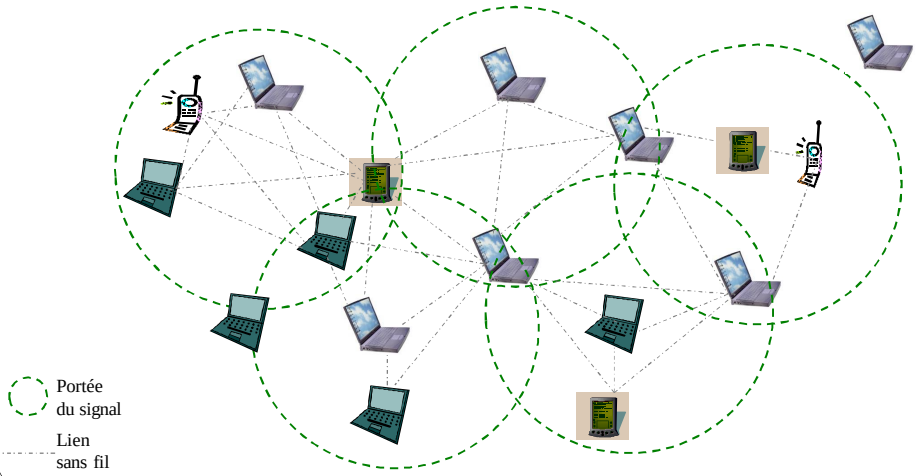
MANET : Mobile Ad hoc NETwork

Réseau de signalisation du
roi perse Darius
(522-486 AC)



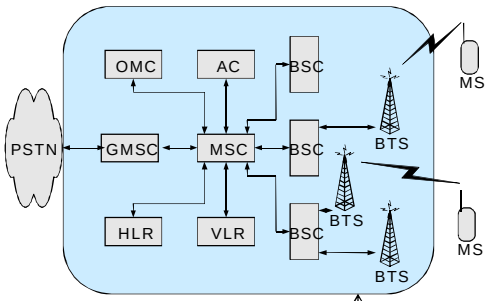
1. Introduction aux MANETs

Topologie de MANETs

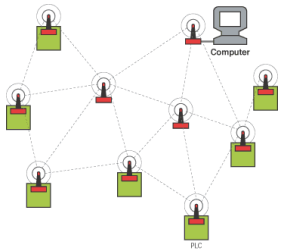


MANETs vs. Réseaux cellulaires

- Route à N sauts vs. route à 1 saut
  - Dans un MANET, chaque nœud est un “routeur”
- Autonomie vs. Administration centralisée
  - Les MANETs sont *self-creating*, *self-organizing*, et *self-administering*



Réseau cellulaire



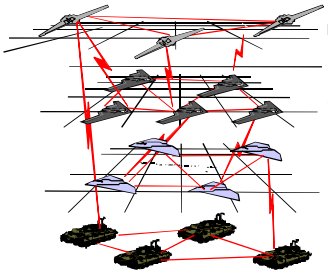
MANET

Infrastructure

Domaines d’utilisation des MANETs

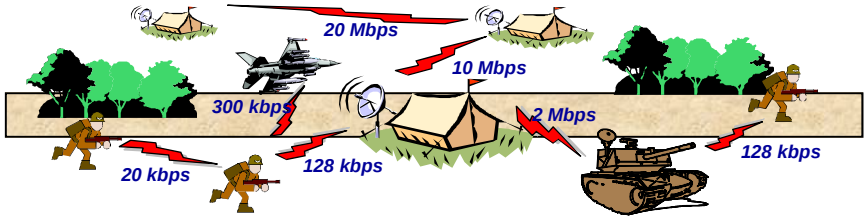
- Partout où il n’y a pas d’infrastructure
- Usages personnels
  - ➔ Téléphone, oreillette, laptop…
- Environnements militaires/champs de bataille
  - ➔ Soldats, chars, avions de chasse
- Environnements civils
  - ➔ Réseau de taxi, de bateaux, petits avions
  - ➔ Salles de conférence, Campus universitaires
  - ➔ Stades, Espaces de jeu…
- Interventions d’urgence
  - ➔ Recherche et secourisme, SAMU
  - ➔ Police, pompiers

Domaines d'utilisation des MANETs



Plusieurs types de BF radio exigés
Plusieurs réseaux interconnectés

"Intelligent Self-Sustaining Distributed Dynamic Networks"



Limites des MANETs

► Limites des réseaux sans fil
- **Pertes de paquets dues aux erreurs de transmission**
- **Variabilité de la capacité des liens**
- **Bande passante limitée**
- **Déconnexions/partitionnements fréquents**
- **Sécurité (diffusion totale !)**

► Limites dues à la mobilité
- **Topologies/routes changeant dynamiquement**
- **Manque de prise en compte de la mobilité par les applications/systèmes (transparence)**

► Limites dues aux équipements/ordinateurs mobiles
- **Durée de batterie limitée**
- **Capacités (calcul et stockage) limitées**

Difficultés/ impossibilité de réservation de ressources

Garantie de QoS !!!

1. Introduction aux MANETs

Travaux et Challenges

- Protocoles MAC
- Routage (unicast, multicast)
- QoS, Réservation de ressources
- Modèles de mobilité
- Gestion et prédiction de
- Localisation des se
- Consommation d'éner
- Couche transport pour MA ETs
- Modèles de simulation de MANETs
- Méthodes d'auto-configuration
- Conception « cross-layer »
- ilware
- rité
- connexion avec d'autres réseaux
- ntennes ("smart antennas")
- …

2. Routage *best effort* dans les MANETs

Propriétés d'un routage idéal pour MANETs

- Totalement distribué (tolérance aux fautes)
- Délai minimal pour la sélection de route (sélection instantanée)
- Convergence vers la solution optimale
- Maintenance de route impliquent un minimum de nœuds
- Reconfiguration rapide : adaptation aux différents changements de topologies
- Fraîcheur des informations d'état stockées (vue précise et instantanée)
- Surcoût minimal (peu calcul, peu de paquets de contrôle, peu d'infos stockées)
- « Scalability »
- Utilisation optimale des ressources du réseau
- Capacités à offrir de la QoS
- Sécurité et vie privée

Sources de difficulté du routage dans les MANETs

- Mobilité des nœuds (topologies changeantes)
- Nombre élevé et/ou très variable de nœuds
- Contraintes de débit (interférence, collision…)
- Collisions entre nœuds (Location-dependent contention)
  plus il y a de nœuds dans la même zone plus il y a de collisions
- Taux d'erreur élevé
- Contraintes d'énergie
- Contraintes de sécurité

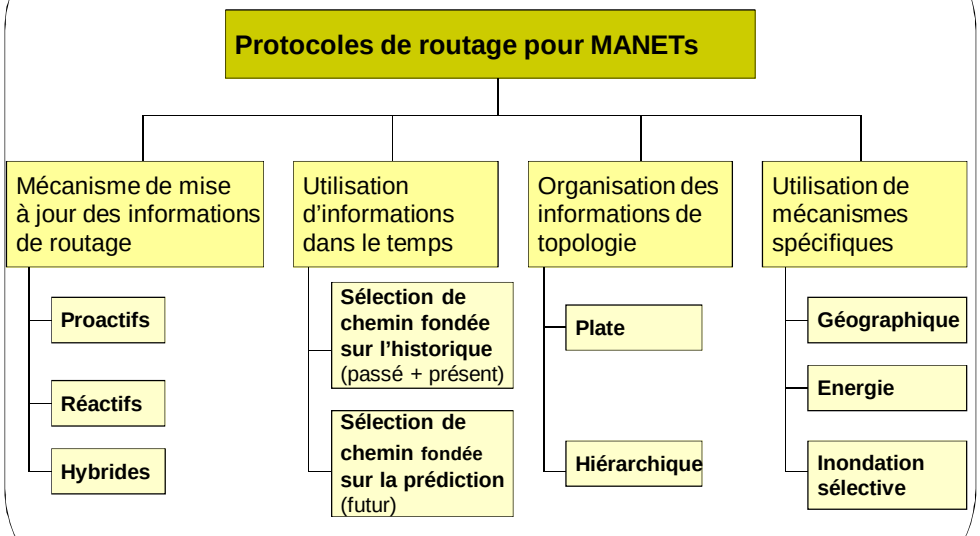
Orientations des travaux

- Routage *Best effort*
- Routage à QoS
- Routage avec optimisation de l'énergie
- Routage et sécurité
- Modèles de mobilité
- Modèles de simulation et analyse

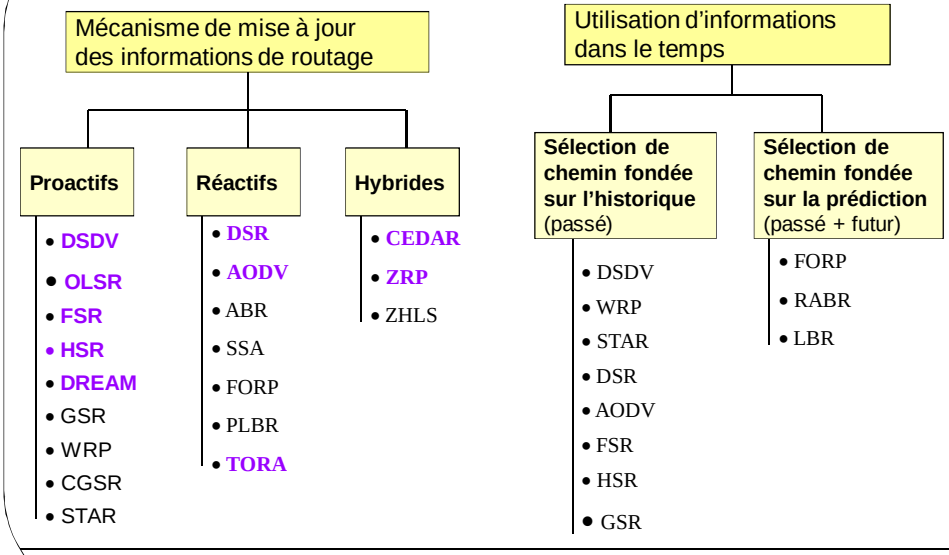
Protocoles de routages pour MANETs

- Beaucoup de propositions
  - Algorithmes de base (originaux) : moins d'une dizaine
  - Leurs extensions : des dizaines
- Aucun protocole n'est parfait

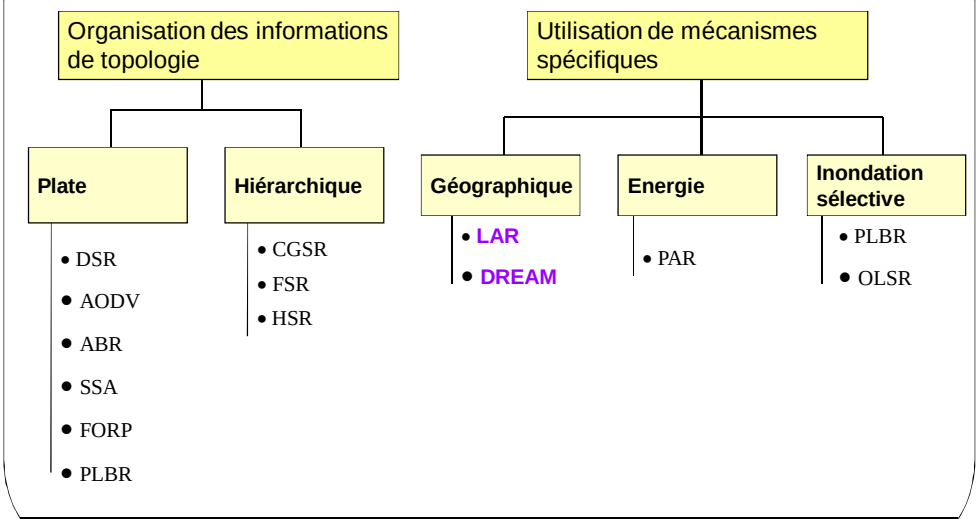
  Tailles variables : Petits (2-100), Moyens-larges (100-10000), très grands (10000 à des millions)
  Besoins différents : débit, délai, disponibilité, sécurité, énergie…
- Le groupe MANET de l'IETF n'a publié que quatre RFC
  - Directives générales (RFC 2501 – Janvier 1999)
  - AODV (RFC 3561 – Juillet 2003)
  - OLSR (RFC 3626 – Octobre 2003)
  - DSR(RFC 4728 – Mai 2007)



2. Routage *Best effort* dans les MANETs



2. Routage *Best effort* dans les MANETs



2. Routage *Best effort* dans les MANETs

ABR	: Associativity-Based Routing
AODV	: Ad hoc On-demand Distance-Vector routing
CEDAR	: Core Extraction Distributed Ad hoc Routing
CGSR	: Cluster-head Gateway Switch Routing
DREAM	: Distance Routing Effect Algorithm for Mobility
DSDV	: Destination Sequenced Distance-Vector routing
DSR	: Dynamic Source Routing Protocol
FORP	: Flow-Oriented Routing Protocol
FSR	: Fisheye State Routing
GSR	: Global State Routing
HSR	: Hierarchical State Routing
LAR	: Location-Aided Routing
LBR	: Link life-Based Routing
OLSR	: Optimized Link-State Routing
PAR	: Power-Aware Routing
PLBR	: Preferred Link-Based Routing
RABR	: Route-lifetime Assessment Based Routing)
SSA	: Signal Stability-based Adaptive routing
STAR	: Source-Tree Adaptive Routing
TORA	: Temporally Ordered Routing Algorithm
WRP	: Wireless Routing Protocol
ZHLS	: Zone-based Hierarchical Link-State routing
ZRP	: Zone Routing Protocol

2. Routage *Best effort* dans les MANETs

- **Protocoles proactifs**
  - Maintenance de routes entre toute paire de nœuds à tout instant
  - Mises à jour continue des tables
  - Pros: + Routes immédiatement disponibles
  - Cons: - Maintenance de routes inutilisées
    - Trafic de contrôle important

- **Protocoles réactifs**
  - Calcul de route quand on en a besoin
  - La source initie la recherche de route
  - Pros: + Faible surcoût car les routes sont établies à la demande
    + Emploi de la diffusion (recherche globale)
  - Cons: - Temps de recherche de route important (inondation)
    - Risque élevé de rafales de trafic

DSDV (Destination Sequenced Distance Vector) routing

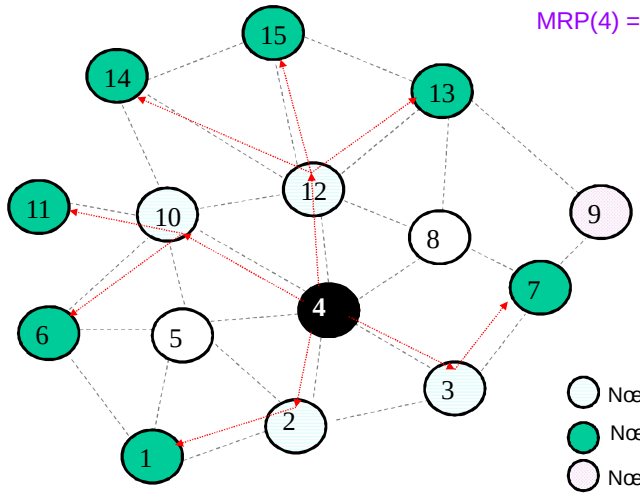
- Un des premiers protocoles proposés pour les réseaux ad hoc [Perkins 1994]
- Protocole proactif
- Basé sur l'algorithme de Bellman-Ford.
- Echange périodique de tables entre voisins.
- Chaque table diffusée par un nœud est accompagnée d'un numéro de séquence.
- Lorsqu'un nœud reçoit une table d'un autre nœud, il vérifie le numéro de séquence de la table reçue pour savoir s'il faut la prendre en compte ou l'ignorer.
- Chaque nœud a une table de routage qui contient pour chaque destination :
  - longueur du chemin le plus court
  - adresse du premier saut suivant vers cette destination
  - numéro de séquence associée à cette destination.

OLSR (Optimized Link State Routing) - [Clausen et al. 2001]

- Protocole proactif à état de lien
- Basé sur les MPR (MultiPoint Relay)
- Réduit l'overhead en réduisant les nœuds qui relayent les informations
- De nombreuses extensions
  - QOLSR (QoS OLSR)
  - S-OLSR (Secure OLSR)
  - MOLSR (Multicast OLSR)
  - PS-OLSR (Power Saving OLSR)
  - FE-OLSR (Fish Eye OLSR)
  - …

Exemple de MRP

MRP(4) = {2, 3, 10, 12}



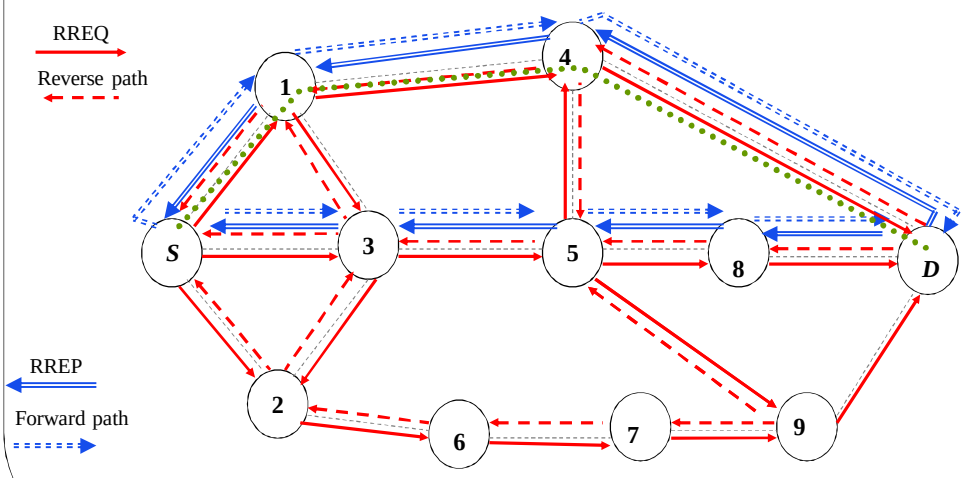
DSR (Dynamic Source Routing) – [Johnson 1996]

- Protocole réactif
- Quand une route est demandée : inondation jusqu'à la destination
- DSR maintient plusieurs entrées dans la table des routes pour chaque destination (reconfiguration rapide en cas de panne, mais coût élevé).
- Chaque requête de route continent un numéro de séquence pour éviter les boucles et la retransmission plusieurs fois de la même requête.
- Les numéros de nœuds traversés par une requête sont rajoutés au fur et à mesure que la requête progresse (i.e. la requête contient le chemin partiel)
- Utilisation de cache de route (éviter aux nœuds intermédiaires de relayer les requêtes si des routes sont déjà connues)

AODV (Ad hoc On-demand Distance Vector) – [Perkins 1999]

- IETF RFC 3561 - July 2003 (rare d'avoir le statut de RFC!)
- AODV est similaire à DSR
  DSR recopie le chemin dans les requêtes; dans AODV, la source et nœuds intermédiaires stockent seulement le prochain saut pour chaque flux
- Caractéristiques
  - Protocole réactif
  - Réduit les diffusions
  - Réduit l'espace mémoire nécessaire à la maintenance de routes
  - Réaction rapide aux coupures de route
  - sans boucle (en utilisant des numéros de séquence)
  - Scalablity

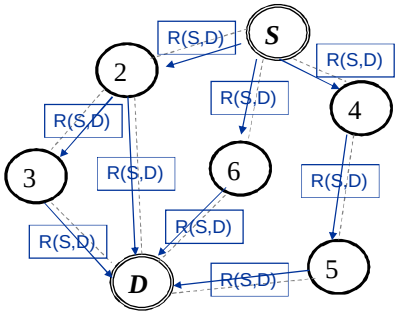
AODV – Exemple



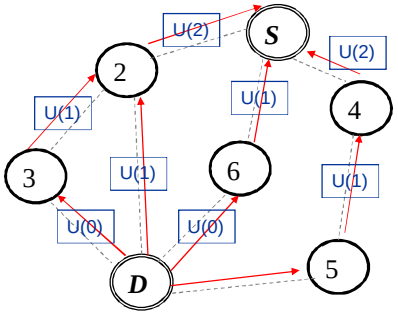
TORA (Temporally Ordered Routing Algorithm) – [Park 1997]

- Protocole réactif
- Point fort : rapidité de recouvrement après coupure de route
- Principe
  - Quand une source *S* veut envoyer des données vers un noeud *D*, elle diffuse une requête de route contenant l'adresse de *D*
  - Chaque nœud intermédiaire relaye la requête jusqu'à ce qu'elle arrive à *D*.
  - *D* diffuse une réponse (Update) contenant une distance égale à 0 (*Dist(D → D)=0*).
  - Quand un nœud reçoit un paquet Update, il teste si la distance qu'il contient est inférieure à celle qu'il connaît déjà. Si oui, il incrémente de 1 la distance contenue le paquet Update reçu et le diffuse et met à jour sa table.
  - Quand le paquet Update arrive à *S*, un graphe orienté de S à *D* est construit (ce graphe donne les routes de *S* vers *D*).

TORA – Exemple (1/2)

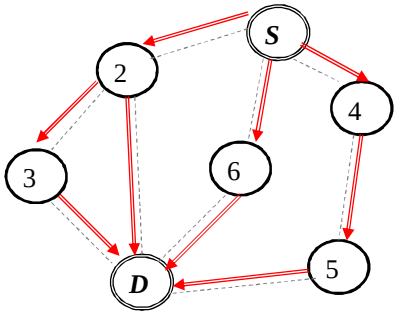


Diffusion de requête (U(S,D))



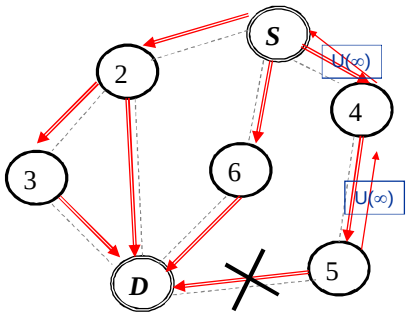
Diffusion de réponse (Update(Distance))

TORA – Exemple (2/2)

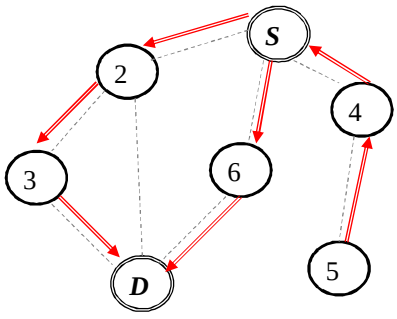


GAO (graphe asymétrique orienté) vers D

- Principe de TORA (suite)
  - Quand un nœud détecte une coupure de lien descendant vers *D*, il diffuse un paquet Update avec une distance supérieure à celles de tous ses voisins, ce qui permet d'inverser le graphe jusqu'à un nœud où il y a un autre chemin descendant vers *D* ou jusqu'à atteindre la source *S*.

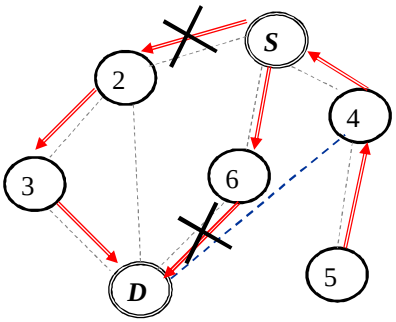


Coupure du lien 5 → D

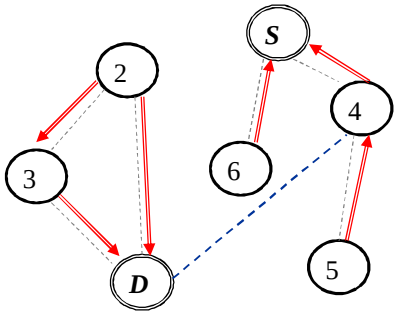


GAO après réparation

TORA – Exemple



Plusieurs coupures de lien



GAO après réparation

- Inconvénient : TORA ne prend en compte de nouvelles routes fraîches que lorsque toutes les routes précédemment calculées sont rompues.

Points critiques ciblés par l'optimisation des protocoles réactifs

► Optimisation de l'inondation

- Limiter les nœuds qui relaient les requêtes
- Solutions : MRP de OLSR, algorithmes basés sur la localisation

► Optimisation de sélection de routes stables

- Ne pas utiliser seulement le nombre de sauts comme critère
- Retenir les nœuds minimisant la probabilité de panne
- Solutions : ABR (Associativity-Based Routing),
  SSR (Signal-Stability-based adaptive Routing)

► Optimisation de la maintenance de routes

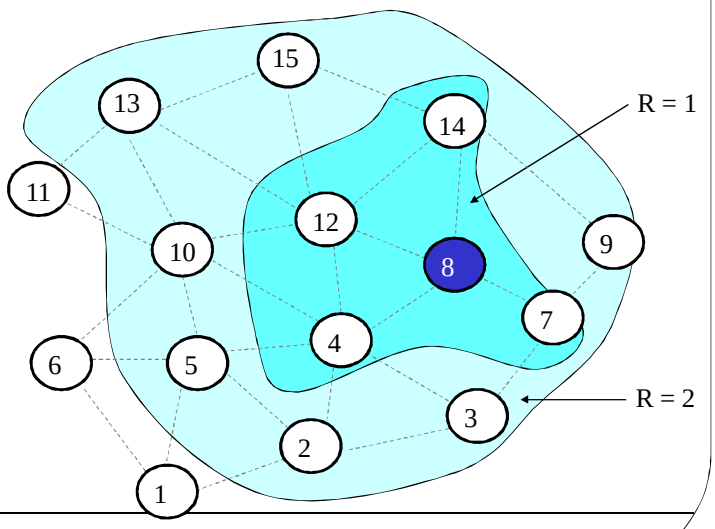
- Minimiser le surcoût de la réparation de route
- Solutions : Maintenance préventive et Réparation locale

(Utilisation de nœuds intermédiaires de réparation au lieu de réparer à la source)

ZRP (Zone Routing Protocol) – [Haas 1999]

- Protocole hybride
- Définir une zone pour chaque nœud *X* composée de nœuds dont la distance -en sauts- est inférieure ou égale à *R* (*R* : rayon de zone)
- Un protocole proactif est utilisé pour que chaque nœud ait connaissance de la topologie du réseau au niveau de sa zone
- Un protocole réactif est utilisé pour trouver une route entre deux nœuds n'appartenant pas à la même zone
- Importance du rayon de zone : un rayon très petit conduit à un pseudo protocole réactif et un rayon élevé conduit à un pseudo protocole proactif

ZRP (ex. Zone du nœud 8)



- Un rayon très petit conduit à un pseudo protocole réactif et un rayon élevé conduit à un pseudo protocole proactif

Protocole hiérarchique – HSR (Hierarchical State Routing)

- Protocole distribué multi-niveaux
- Regroupement des nœuds en niveaux (selon une politique). Un niveau = N clusters
- Chaque groupe a un leader.
- Chaque nœud maintient une table d'état de lien de ses voisins. Echange périodique des infos d'état au sein de chaque cluster.
- Chaque leader de cluster échange la topologie et l'état de lien avec les autres leaders qui lui sont voisins.
- Sélection de chemin : en suivant les niveaux hiérarchiques
- Avantage : réduction significative de la table de routage
- Inconvénients : perte de précision d'état, overhead

HSR - Exemple

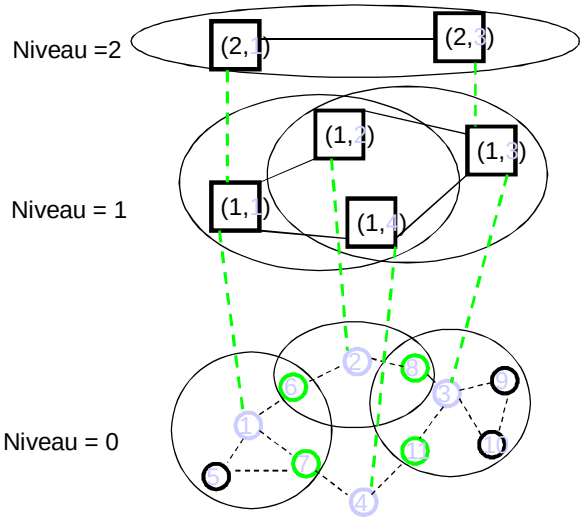


Table du noeud 5

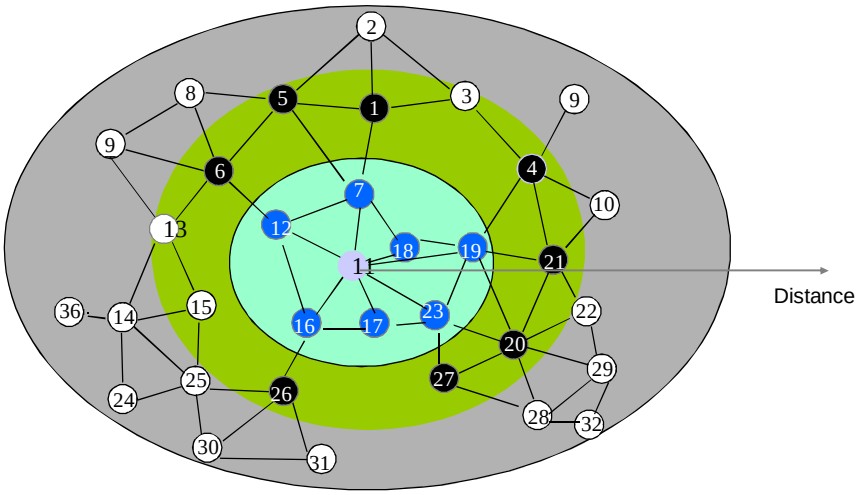
DestID	Chemin
1	5-1
6	5-1-6
7	5-7
(1,2)	5-1-6-(1,2)
(1,4)	5-7-(1,4)
(2,3)	5-7-(1,4)-(2,3)

Protocoles hiérarchiques

FSR (Fish eye State Routing)

- Objectif : réduire le nombre les informations maintenues
- Idée de base : maintenir des informations de moins en moins précises au fur et à mesure que l'on s'éloigne du nœud considéré.
- Chaque nœud maintient des infos très précises concernant ses voisins et des infos moins précises sur les autres nœuds.
- Les infos d'état ne sont pas diffusées vers tous les nœuds.
- Chaque nœud échange avec une fréquence élevée ses infos avec ses voisins et à une fréquence plus faible avec les autres nœuds.

FSR (Fish eye State Routing)



Protocoles basés sur la localisation

- Hypothèse commune aux algorithmes de routage réactifs, proactifs et hybrides :
  Les nœuds découvrent les informations (totales ou partielles) sur la topologie en échangeant des messages d'état. Ces informations guident ensuite les décisions de routage.
- Le routage basé sur la localisation (ou routage géographique) est très efficace pour les réseaux de grande taille et avec un rythme important de changement de topologie
- Chaque nœud connaît sa position <x, y> en utilisant un service de localisation et connaît celles des autres par échange de messages.

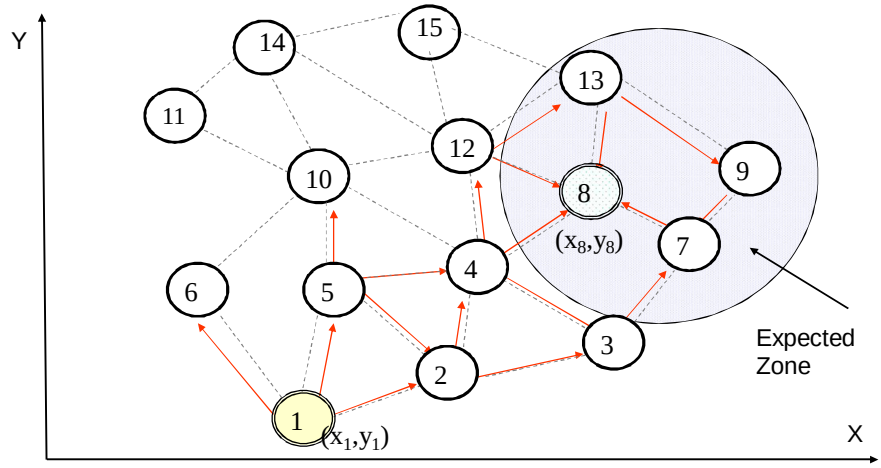
Techniques de localisation de mobiles

- Classification des techniques
  - Localisation à l'intérieur (indoor) : utilisation de moyens du WLAN, UWB…
  - Localisation à l'extérieur (outdoor) : utilisation de satellites (GPS)
  - Pour réseaux cellulaires : utilisation de stations de base…
  - Pour réseaux ad hoc : utilisation d'antennes, GPS
  - Dans la bande (inband)
  - Hors bande (outband)
- Critères d'évaluation des techniques
  - Coût de localisation (délai, nombre de messages, tarif…)
  - Précision (ex. des dizaines de mètres pour le GPS)
  - Disponibilité du service

Protocole LAR (Location-Aided Routing) – [Ko 2000]

- Optimisation de protocole réactif pour minimiser le surcoût de l'inondation
- LAR réduit le nombre de messages en limitant la zone de diffusion de requêtes
- Les informations de localisation obtenues grâce à une requête antérieure sont utilisées pour estimer la localisation du destinataire
- Le service de localisation ne fait pas partie du protocole LAR (point faible)
- La source diffuse sa requête à ses voisins, requête incluant son estimation de la location du destinataire et sa distance estimée par rapport au destinataire.
- Chaque nœud *X* qui reçoit la requête calcule sa distance par rapport au destinataire. S'il est plus proche de la destination que la source, il relaie la requête à ses voisins en remplaçant la distance source-destination par *X*-destination dans la requête.
- Le destinataire *D* qui reçoit une requête renvoie une réponse au nœud qui a relayé la requête. La réponse contient la position actuelle de D et son temps courant, il peut aussi envoyer des informations sur sa vitesse et direction.

- Connaissance de localisation ➔ Meilleure redirection des requêtes



Protocole DREAM (Distance Routing Effect Algorithm for Mobility)

- Chaque nœud communique de manière proactive sa localisation aux autres nœuds
- Surcoût d'échange d'information de localisation réduit en:
  - Envoyant plus fréquemment les infos aux nœuds voisins qu'aux nœuds éloignés
  - Les infos de localisation sont envoyées avec un rythme qui dépend de la mobilité du nœud
- Le service de localisation fait partie du protocole DREAM
- DREAM = adaptation de Fish eye routing + localisation

Débat Réactifs contre Proactifs – Leçons

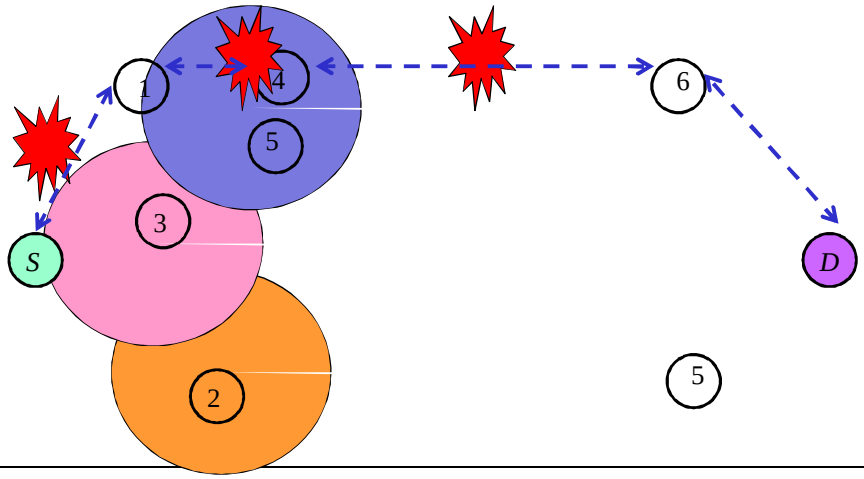
- En général : actuellement pas de gagnant clair
- **Réactif** ou **Proactif** : **dépend du contexte**
  Diversité des flux (Nombre moyen de sources, durées des flux…), Mobilité des nœuds, Contraintes de QoS, énergie
- **Leçons** :
  - Protocoles réactifs plus adaptés quand la diversité du trafic est faible
    Protocoles proactifs plus adaptés quand la diversité du trafic est élevée et en cas d'urgence.
  - En cas de trafic moyen ou élevé : les paquets de contrôle des protocoles proactifs aggravent les situations de congestion, réduisant ainsi le rendement du réseau
  - Vitesse de mobilité : affecte les protocoles proactifs (à cause de toutes les coupures), mais n'affecte pas les avantages des protocoles réactifs (car ces protocoles réagissent uniquement aux coupures des routes utilisées).

Quelques observations communément admises

- Sur les protocoles proactifs
  - DSDV : ne répond pas rapidement aux changements de topologies
  - OLSR semble être le plus efficace. Mais d'autres algorithmes proactifs, comme TBRPF (Topology Broadcast based on Reverse-Path Forwarding) ont des performances semblables à OLSR)
- Sur les protocoles réactifs
  - TORA : considéré comme le plus mauvais en terme de rapidité
  - DSR et AODV sont les plus déployés
  - Choix entre DSR et AODV dépend de la charge du réseau, de la vitesse de mobilité des nœuds, du taux de pannes. Ce n'est pas toujours simple à choisir.

3. Routage à QoS dans les MANETs

Difficulté de fournir la QoS : un exemple simple



Sources des difficultés à traiter le routage à QoS

- Variabilité de la topologie ⇒ Remise en cause des réservations
- Imprécision des informations d'état
- Absence d'infrastructure centralisée et de contrôle centralisé
  ⇒ Difficulté de coordonner les réservations
- Capacités des liens fluctuantes
  ⇒ Difficulté/impossibilité de prédiction de la bande passante disponible
  ⇒ Dégradation de la qualité des applications audio/vidéo
  ⇒ Fourniture de QoS soft seulement
- Limitations de puissance
- Taux d'erreurs élevé sur les canaux radio
- Médium non sûr
- ….

Problèmes de QoS les plus étudiés

- Métriques de QoS
  - **Bande passante**
  - Aspects temporels (**délai** de transfert, gigue…)
  - Taux d'erreurs, taux de perte
  - Disponibilité/robustesse
  - Coût
  - Autres (batterie, espace mémoire,…)
- Niveaux/classes de QoS
  - Meilleur effort
  - Probabiliste/stochastique/statistique (soft QoS)
  - Garantie absolue (hard QoS)

3. Routage à QoS dans les MANETs

Beaucoup de propositions de protocoles

AQOR (Ad hoc QoS on-demand routing)
BR (Bandwidth Routing)
CEDAR (Core Extraction Distributed Ad hoc Routing)
D-LOAR (Delay Load-Aware On-demand Routing)
ODRP (On-demand Delay-constrained Routing Protocol)
OLMQR (On-demand Link-state Multipath QoS Routing)
OQR (On-demand QoS Routing)
PLBQR (Predictive Location-Based QoS Routing)
QOLSR (QoS OLSR)
QoS-AODV (QoS extension to AODV)
QoS-ASR (QoS-Adaptive Source routing)
SBSR (Segmented Backup Source Routing)
TBP (Ticket-Based Probing QoS routing protocol)
TDR (Trigger-based Distributed Routing)

3. Routage à QoS dans les MANETs

Exemples protocoles de routage à QoS

TBR (Ticket Based Routing)

- Idée de base : limiter les paquets de requête et ne les diriger que vers la destination en utilisant des tickets
- Chaque source possède un nombre de tickets de deux types (Jaune et Vert).
- Chaque paquet-requête contient un nombre de tickets. A chaque nœud traversé par la requête, une décision est prise sur le nombre de tickets à retirer du paquet (par exemple, un lien avec un délai faible retire plus de tickets jaunes qu'un lien avec un délai élevé)
- L'objectif du nombre de tickets jaunes est d'accroître la probabilité de trouver un chemin. Ainsi, un paquet-requête avec peu de tickets jaunes, signifie chercher un chemin avec un délai faible.
- L'objectif des tickets verts est de maximiser la probabilité de trouver un chemin avec un coût faible.
- Etat de lien maintenu = BP disponible et délai

Exemples protocoles de routage à QoS

QoS-AODV (QoS-enabled Ad hoc On-demand Distance Vector) – Perkins (1/2)

- Extension des paquets RouteRequet et RouteReply et un paquet rajouté (QoSLost)
- RouteRequest étendu par :
  - Délai maximum (*Dmax*) demandé pour transmettre de *S* à *D*
  - Minimum de bande passante (*BPmin*) demandée
- Table de routage étendue par
  - Liste des sources demandant des garanties de délai
  - Liste des sources demandant des garanties de BP

Exemples protocoles de routage à QoS

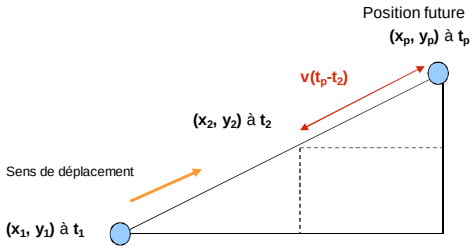
QoS-AODV (QoS-enabled Ad hoc On-demand Distance Vector) – Perkins (2/2)

- RouteReply étendu par :
  - Délai maximum cumulé estimé par le nœud ayant relayé la réponse
    Si délai cumulé est supérieur à *Dmax*, la réponse est écartée
  - Minimum de bande observée entre le nœud qui relaye la requête et *D*
    Si la BP estimée est inférieure à *BPmin*, la réponse est écartée
  - Liste des sources demandant des garanties de délai
  - Liste des sources demandant des garanties de BP
- Paquet QoSLost
  - Chaque nœud qui ne peut plus garantir la QoS envoie un paquet QoSLost aux sources concernées
- Avantage : simple extension de AODV
- Inconvénient : pas de réservation effective de ressources

Exemples protocoles de routage à QoS

PLBQR (Predictive Location-based QoS Routing) (1/2)

- **Principe** : prédiction de la position future de la destination pour sélectionner les routes
- **Protocole de mise à jour** : chaque nœud diffuse périodiquement (ou en cas de changement important) sa position et l'état de ses ressources. Dans certains cas, il peut aussi diffuser sa vitesse et sa direction.
- **Prédiction de position** : Basée sur la similarité de triangles et théorème de Pythagore.



Exemples protocoles de routage à QoS

PLBQR (Predictive Location-based QoS Routing) (2/2)

- **Prédiction de délai** : PLBQR suppose que le délai e2e de *S* à *D* est égal (estimé) au délai e2e du dernier paquet de mise à jour émis par *D* et reçu par *S*. Ce délai est donc variable selon la charge (pas de garantie de borne de délai).
- **Routage à QoS**
  - Découverte de voisins à l'aide de la prédiction position-délai
  - Recherche des routes satisfaisant la QoS
  - Sélection de la route la plus courte géographiquement
  - La route est insérée dans les paquets de données

Exemples protocoles de routage à QoS

CEDAR (Core Extraction based Distributed Ad hoc Routing)

- Protocole réactif

- Trois principes de base
  - Extraction de nœuds dominants
    - Un ensemble de nœuds est choisi (de manière statique ou dynamique) pour former le cœur du réseau (nœuds dominants).
    - Chaque nœud est soit un nœud dominant soit un voisin d'un nœud dominant (dans ce cas il est dit nœud rattaché).
    - Quand un nœud dominant bouge, les nœuds qui lui sont rattachés doivent trouver un nouveau nœud de rattachement.
    - Chaque nœud dominant maintient l'état local de la topologie des nœuds appartenant à son domaine et calcule les routes pour ces nœuds.

Exemples protocoles de routage à QoS

CEDAR (Core Extraction based Distributed Ad hoc Routing)

- Trois principes de base (suite)
  - Propagation d'état de lien
    - Chaque nœud dominant maintient l'état de la BP disponible dans son domaine et l'échange avec les autres nœuds dominants.
    - Optimisation : seules les infos concernant les liens avec BP disponibles significatives et liens stables sont échangées (utilisation de seuils)

Exemples protocoles de routage à QoS

CEDAR (Core Extraction based Distributed Ad hoc Routing)

- Sélection de route en deux phases

1) Sélectionner une route en ne prenant en compte que les nœuds dominants

– La requêté émise par la source *S* est diffusée par le nœud dominant de *S* aux autres nœuds dominants.

– Chaque nœud dominant, qui n'a pas la destination *D* comme nœud rattaché, rajoute son ID à la requête et la relaye à ses nœuds dominants voisins.

– Si un nœud dominant n'a pas de lien direct avec un autre nœud dominant voisin, il utilise les nœuds membres pour lui envoyer la requête (on parle de lien virtuel).

– Quand une requête parvient à *D*, un chemin (core-path) est construit et renvoyé à *S*.

2) Compléter le chemin répondant aux besoins de QoS

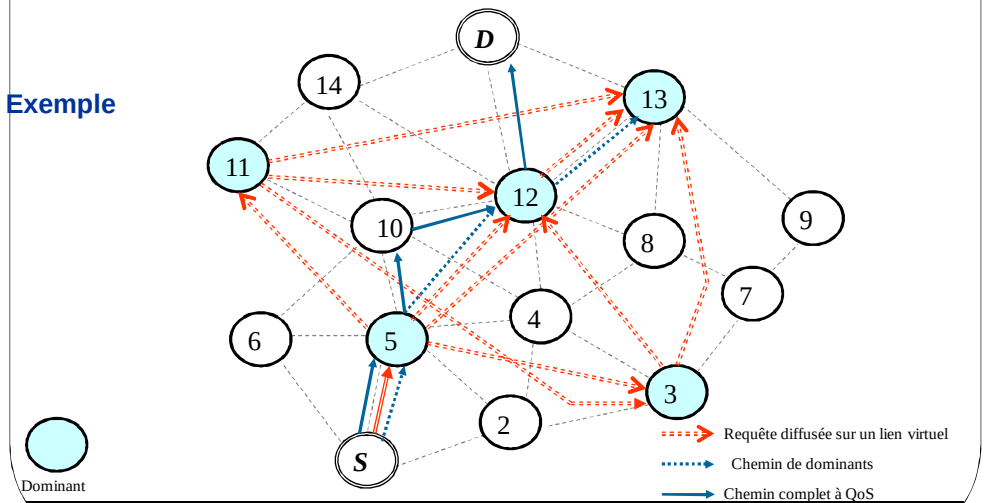
– Le nœud dominant de *S* sélectionne un chemin vers un nœud dominant *X*, le plus éloigné, appartenant au chemin déterminé dans la phase 1) et qui répond au besoin de QoS.

– Le nœud dominant *X* sélectionné précédemment effectue la même opération de sélection que la source *S*. Le processus de sélection se répète jusqu'à atteindre *D*.

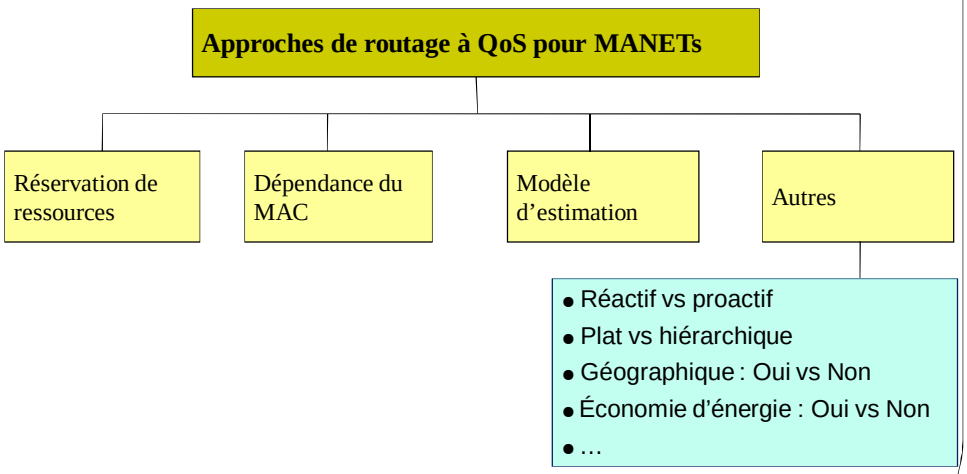
Exemples protocoles de routage à QoS

CEDAR (Core Extraction based Distributed Ad hoc Routing)

Exemple



Classification des approches (1/2)



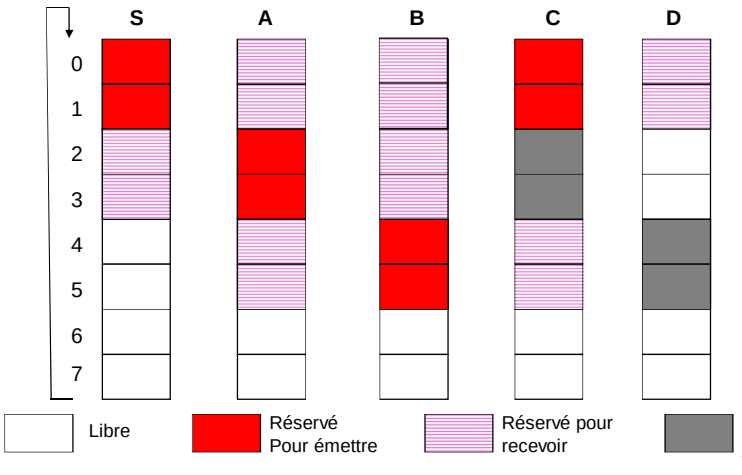
Classification des approches (2/2)

- **Réservation de ressources**
  - Avec réservation
    - par flux (IntServ) ou par marquage de paquet (DiffServ)
    - Avec maintenance
      - * *soft* (MRSVP) et réparation de chemins (la plus répandue)
      - * dure (rare et peu adaptée aux MANETs)
  - Sans (approche optimiste)

- **MAC sous-jacent**
  - CSMA/CA
  - TDMA
  - CDMA-over-TDMA
  - Quelconque

- **Modèle d'estimation du délai**
  - Spécifique
  - Quelconque

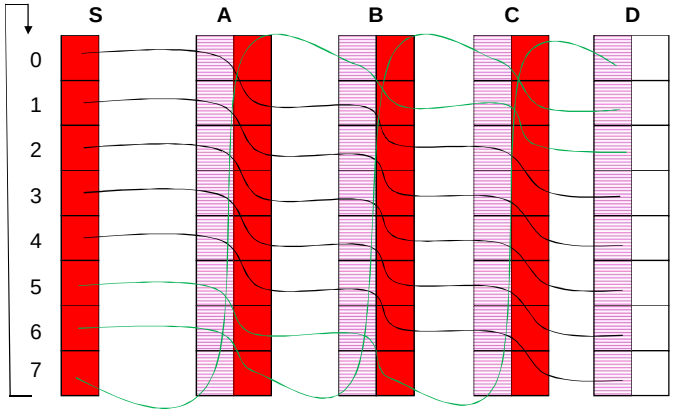
Exemple de réservation de slots synchrone avec TDMA



S demande un chemin à 2 slots (25%)

65% des slots consommés

TDMA (en mode filaire)



S peut avoir un chemin à 8 slots

RTMAC (Real-Time MAC) protocole – [Manoj 2002] (1/6)

- Objectif : support de trafic temps réel (type CBR) sur réseaux ad hoc
- RTMAC : un protocole MAC + un algorithme de routage
- Protocole MAC
  - Extension de 802.11 DCF (Distributed Coordination Function)
  - Deux parties : MAC pour le *Best effort*
    un protocole de réservation pour flux temps réel
- Algorithme de routage = une extension de DSDV
  - Sélection de chemin avec QoS demandée
  - Échange de tables de réservation entre voisins
  - Déclenchement des opérations de réservation/libération de ressources

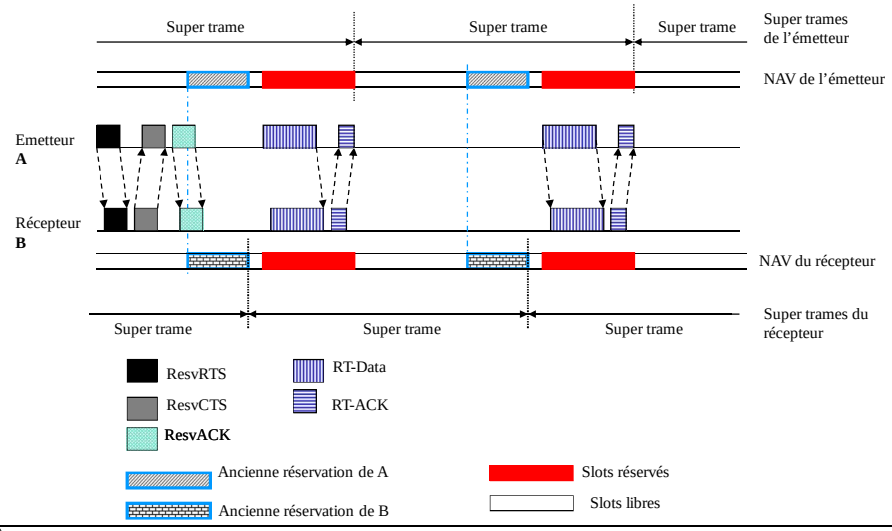
RTMAC (Real-Time MAC) protocole (2/6)

- Procédure de réservation
  - Quatre nouveaux paquets prioritaires (*ResvRTS*, *ResvCTS*, *ResvACK*, *ResvNCTS*) sont utilisés pour réserver
  - Division du temps en super trames
  - Chaque super trame contient un certain nombre de slots à réserver (*resv-slots*)
  - Durée d'un *resv-slot* = 2 fois le délai maximum de propagation
  - Chaque flux réserve un certain nombre de *resv-slots* consécutifs dans chaque super trame sur chaque lien du chemin de la source à la destination
  - Chaque nœud maintient une table de réservation indiquant pour chaque paire <source, destination> : les *resv-slots* et les instants de début et fin de réservation

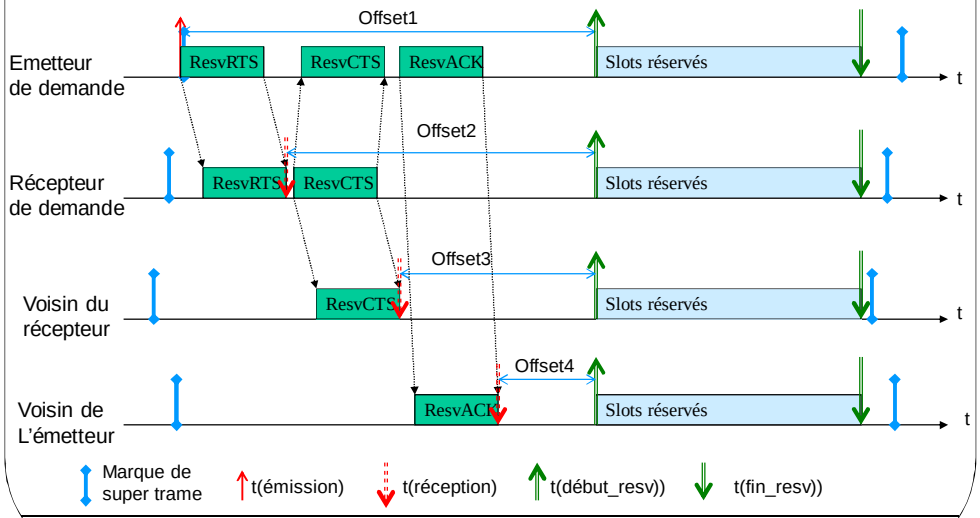
RTMAC (Real-Time MAC) protocole (3/6)

- Procédure de réservation (suite)
  - Soit A le nœud qui veut réserver *Y* slots auprès du nœud B.
    Il envoie un *ResvRTS* contenant *Y* et un temps relatif (offset) qui indique le début de la réservation par rapport à l'instant absolu d'envoi du *ResvRTS*.
  - En recevant le *ResvRTS*, le nœud B teste si les slots demandés sont libres.
  - Si les slots sont libres, le nœud B met à jour sa table de réservation et envoie un *ResvCTS* contenant les mêmes informations que le *ResvRTS*. Ensuite :
    - *** Les nœuds voisins de B mettent à jour leur table de réservation tenant compte des informations du *ResvCTS*.**
    - *** Le nœud A envoie un *ResvACK* contenant les mêmes informations de réservation pour permettre à ses voisins de mettre à jour leur table.**
  - Si les slots demandés ne sont pas libres, un *ResvNCTS* (négatif) est renvoyé obligeant A à modifier ses paramètres et tenter la réservation plus tard.
  - Si le *ResvRTS* est reçu par B durant un slot déjà réservé, il l'ignore. S'il répond par un *ResvNCTS*, il peut causer des collisions avec des réservations des voisins.

RTMAC (Real-Time MAC) protocole (4/6)



RTMAC (Real-Time MAC) protocole (5/6)



RTMAC (Real-Time MAC) protocole (6/6)

- Asynchronisme dans RTMAC
  - Pas d'exigence de synchronisation de super trames
  - RTMAC utilise des temps relatifs pour effectuer les réservation
    - L'émetteur se repère à partir de l'instant d'envoi du *ResvRTS*
    - Le récepteur se repère à partir de l'instant de réception du *ResvRTS*
    - Les voisins du récepteur se repèrent à partir de l'instant de réception du *ResvCTS*
    - Les voisins de l' émetteur se repèrent à partir de l'instant de réception du *ResvACK*

Méthodes d'estimation de bande passante

Méthode de Cansever et al 1999

- **Principe**

$$BPDispo_i = BpNonUtilisee_i - \sum_{j \in N_i} \sum_{k \in N_j} BP(flux_{j \to k})$$

$$BpNonUtilisee_i = CapacitéCanal_i - \sum_{j \in N_i} BP(flux_{i \to j})$$

$$N_x : voisins \quad de \quad x$$

- Difficulté d'utilisation : estimation de la bande passante consommée par les flux du nœud *i* et de ses voisins

Méthodes d'estimation de bande passante

Méthode de Kazantzidis et Gerla 2002

- **Méthode utilisable avec IEEE 802.11 en mode DCF**
- **Principe**

$$BPDispo_{i \to j} = (1-u) * Rendement_{i \to j}$$

$$Rendement_{i \to j} = Moyenne(RendementPaquet_k, k = 1,...,32)$$

$$RendementPaquet = \frac{TaillePaquet}{t\_queue + (t\_trans + t\_CE + t\_OH) * R + \sum_{r=1}^{R} t\_b_r}$$

$$u = 1 - \frac{TempsLibre}{DureeFenetreMesure}$$

- Difficulté d'utilisation : estimation de temps intermédiaires *t_queue*,… et du nombre de retransmissions *R*

Méthodes d'estimation de délai

Principe répandu

- **Estimation de délai moyen par EWMA** (*Exponentially Weighted Moving Average*)

$$D^j_{moy} = (1-\alpha)D^j_{reel} + \alpha * D^{j-1}_{moy}$$

$$D^j_{moy} = \alpha * D^j_{moy} + (1-\alpha) * \beta * \left| D^j_{reel} - D^j_{moy} \right|$$

- Difficulté d'utilisation : valeurs des poids α et β.

4. Architectures et modèles pour les MANETs

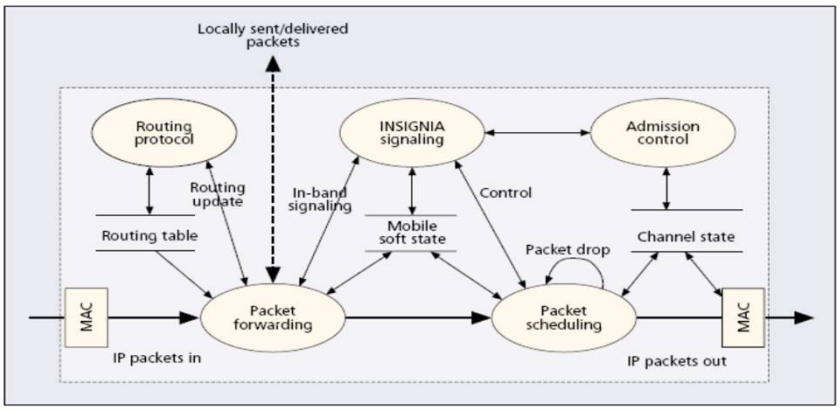
- Equivalent de IntServ ou DiffServ pour MANETs

- Composants fonctionnels
  - Protocole de routage
  - Réservation de ressources et signalisation (MRSVP)
  - Contrôle d'admission
  - Ordonnancement
  - Rejet de paquet, shaping
  - Politique de gestion de ressource

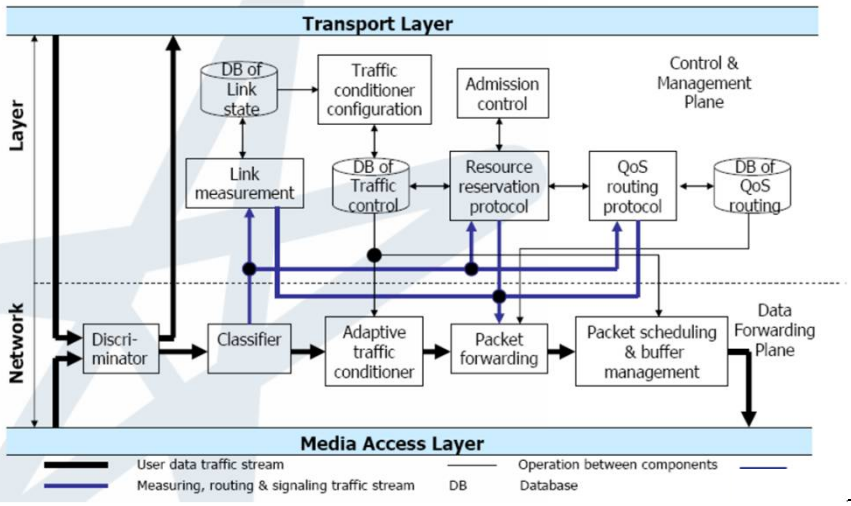
- Architectures/modèles proposés :
  FQMM, iMAQ, INSIGNIA, INORA, PRTMAC, SWAN, 2LQoS

Architecture INSIGNIA

- **INSIGNIA =** IP-Based Quality of Service Framework for Mobile ad Hoc Networks
- Pour applications adaptatives (soft QoS)
- Trois niveaux de service : *Best effort*, *Base QoS* (ou min QoS), *Enhanced QoS*

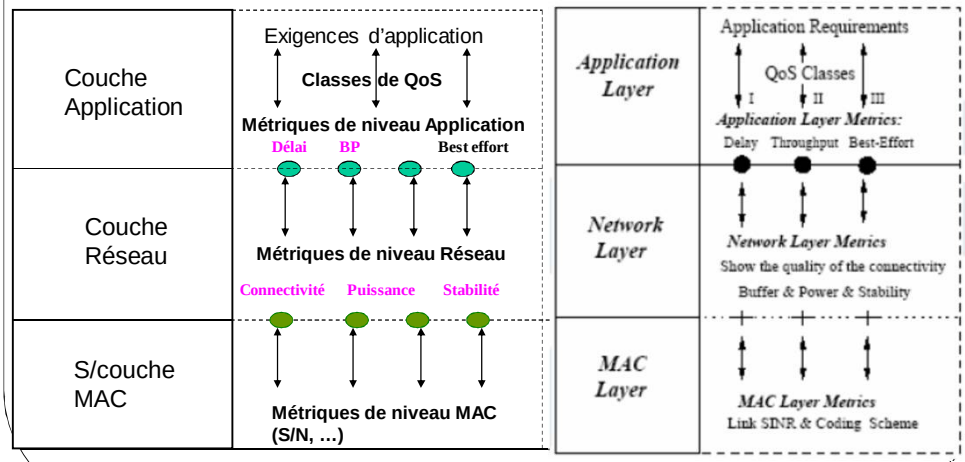


Architecture FQMM (Flexible Qos Model for Manets)

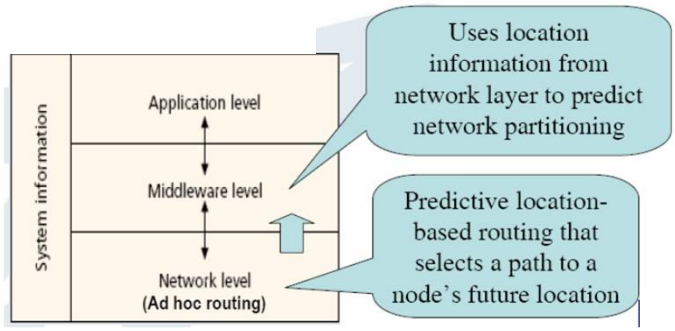


Modèle 2LQoS (Two-Layer QoS model)

- Séparation de la QoS selon trois niveaux : Application, Réseau et MAC



Modèle iMAQ (Integrated Manet Qos)



5. Conclusion

- Routage = fonction clé pour l'utilisation des MANETs
- Beaucoup de protocoles de routage existent
- **Après l'euphorie ! tendance vers : MANETs spécialisés, RCSF, Réseaux Mesh**
- **Challenges**
  - Approches statistiques pour l'estimation des métriques
  - Fonctions Poids génériques et configurables
  - Ingénierie de trafic pour les réseaux ad hoc
  - Méthodes de réservation de ressources
  - Méthodes de gestion de clash de réservation de ressources
  - Contrôle d'admission et adaptabilité des applications
  - Messagerie d'urgence dans les MANETs

5. Conclusion

- Auto-configuration
  - Unification des approches, modélisation des fonctions de routage
  - Choix guidé de protocole de routage (en fonction des flux, densité du réseau, mouvements des nœuds)
  - Reconfiguration automatique du protocole de routage (adaptation)
  - Combinaison efficace des protocoles (routage sensible à la charge)
  - Maîtrise de l'imprécision de l'information d'état de lien pour anticiper
  - Meilleure exploitation des infos de localisation (3D, obstacles…), modèles de déplacement, modèles de réservation de ressources
  - Problème d'optimisation : surcoût du routage – QoS attendue (consommation d'énergie, débit délai, perte, gigue…)
  - Contrôle d'admission plus intelligent pour limiter les congestions (à la source, aux niveaux des nœuds intermédiaires) pour augmenter le througput du réseau : avoir un réseau dont les capacités des liens sont contrôlables.