Aspects avancés des réseaux

Andrzej Duda
Andrzej.Duda@imag.fr
Ensimag, D311

Claude Castelluccia
Claude.Castelluccia@inria.fr
Inria Rhône-Alpes

Organisation

- Deux parties
- 1ère partie
  - introduction
  - principes du contrôle de trafic
  - qualité de service dans IP
  - contrôle de congestion dans TCP
- 2ème partie
  - communications mobiles
  - GSM, GPRS
  - mobilité dans l'Internet

Bibliographie

- W. Stallings "*High Speed Networks* ", Prentice-Hall, 1998
- J.F. Kurose and K.W. Ross “Computer Networking”, Addison-Wesley, 2000
- S. Keshav "*An Engineering Approach to Computer Networking*", Addison-Wesley, 1997
- G. Hebuterne "Gestion du trafic - modèles pour les réseaux de télécommunications", poly INT

Pré-requis

- Cours Réseaux
  - architectures en couche
  - TCP/IP
- Bases de la théorie de files d'attente

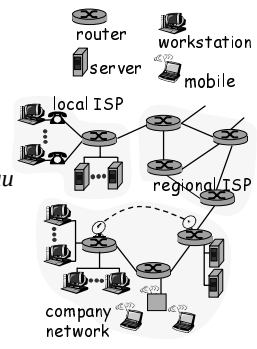
Partie 1
Introduction

Architectures en couche
Performances

Internet

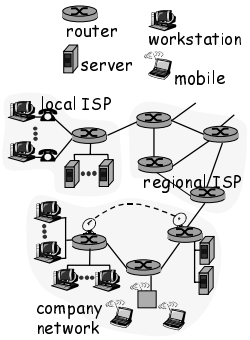
- millions de systèmes connectés: hôtes (*hosts, end-systems)*
  - stations PC, serveurs
  - PDA, toasters

  exécutent *applications réseau*
- *liens de communication*
  - fibre, cuivre, radio
- *routeurs:* acheminent de paquets de données à travers le réseau



Internet

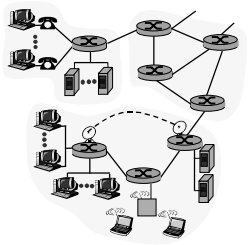
- *protocoles*: contrôle de la communication
  - TCP, IP, HTTP, FTP
- *Internet:* "network of networks"
  - structure hiérarchique
  - Internet public vs. intranet privé
- standards d'Internet
  - RFC: Request for comments
  - IETF: Internet Engineering Task Force



7

Internet : services

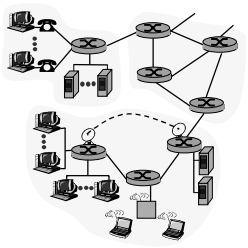
- *infrastructure de* communication qui supporte des applications très variées :
  - WWW, email, news, NFS, X, telnet, talk, jeux, téléphone
  - futures ?
- services de communication
  - sans connexion
  - en mode connecté



8

Structure de réseau

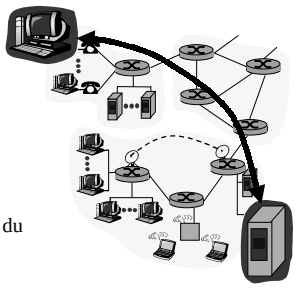
- bordure :
  - applications and hosts
- cœur :
  - routeurs
  - *"network of networks"*
- réseaux d'accès, canaux physique: liens de communication



9

Bordure

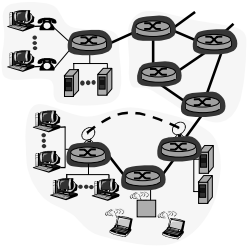
- Hôtes
  - exécutent des applications
    - » WWW, email
  - se connectent au cœur du réseau
- Modèle client/serveur
  - client envoie une requête
  - attend la réponse
  - reçoit un service de la part du serveur
  - e.g., client WWW (navigateur)/ serveur, e-mail FTP, telnet, X



10

Cœur du réseau

- Interconnexion de systèmes intermédiaires
- *La* question: comment les données sont transportées ?
  - commutation de paquets : données découpées en paquets acheminés par les routeurs
  - commutation de circuits : circuit dédié par appel et mis en place par les commutateurs



11

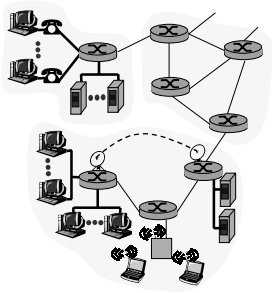
Réseaux d'accès et canaux physiques

Q: Comment se connecter au routeur de bordure

- réseaux d'accès résidentiels
- réseaux d'accès institutionnels
- réseaux d'accès mobiles

Caractéristiques ?

- débit nominal
- partagés ou dédiés



12

Modèle OSI de l'ISO

Application	● Fonctions communes
Présentation	● Format interchangeable
Session	● Organisation du dialogue
Transport	● Transmission fiable entre processus
Réseau	● Acheminement à travers le réseau
Liaison	● Transmission entre deux sites
Physique	● Transmission des signaux

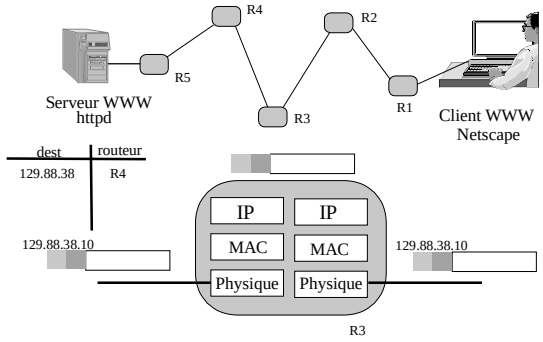
13

Modèle TCP/IP

Applications	● FTP, WWW, telnet, SNMP
Transport	● Transport entre processus - TCP, UDP
Réseau	● Routage - IP
Transmission	● Transmission entre deux sites – réseau local – physique

14

IP



15

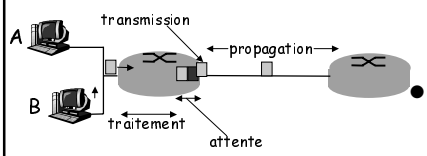
Performances

- Débit
  - nombre de bits par unité de temps (Kbit/s, Mbit/s, Gbit/s, Tbit/s, …)
  - *bandwidth, throughput, bit rate*
- Délai ou latence
  - le temps entre l'émission et la réception d'un bit
  - *delay or latency*
  - temps aller-retour *(RTT - Round-Trip Time)*

16

Latence dans les réseaux de commutation de paquets

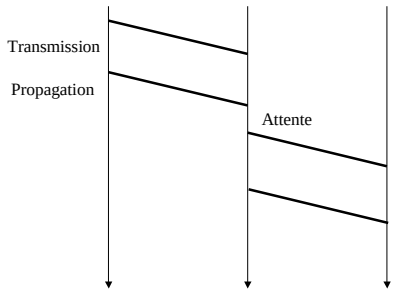
- Quatre sources de délai à chaque passage de routeur



- traitement :
  - vérifier le code d'erreurs
  - choix de liens de sortie
  - petit par rapport aux autres
- attente
  - dépend de la charge du routeur

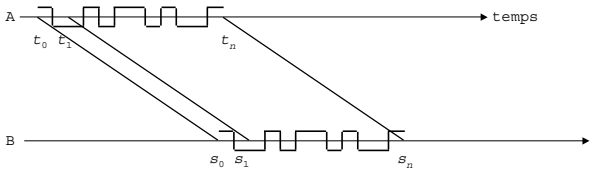
17

Latence



18

Temps de propagation



- Propagation entre A and B
  - le temps nécéssaire pour que le front du signal arrive de A à B : $s_i - t_i$

19

Performances

- Délai ou latence
  - Latence = Propagation + Transmission + Attente
  - Propagation = Distance / Vitesse
    - » cuivre : Vitesse = 2.3×10^8 m/s
    - » verre : Vitesse = 2×10^8 m/s
    - » Transmission = Taille / Débit
- 5 µs/km
- New York - Los Angeles en 24 ms
  - requête - 1 octet, réponse - 1 octet : 48 ms
  - fichier 25 Moctets sur 10 Mbit/s : 20 s
- Tour du monde en 0.2 s

20

Exemple

- à l'instant 0, A envoi un paquet de 1000 octets à B; quand il est reçu par B (vitesse = 2e+08 m/s) ?

distance	20 km	20000 km	2 km	20 m
débit	10 kb/s	1 Mb/s	10 Mb/s	1 Gb/s
propagation	0.1 ms	100 ms	0.01 ms	0.1 µs
transmission	800 ms	8 ms	0.8 ms	8 µs
latence	800.1 ms	108 ms	0.81 ms	8.1 µs
	modem	*satellite*	*LAN*	*Hippi*

21

A Simple Protocol: Stop and Go

- Packets may be lost during transmission: bit errors due to channel imperfections, various noises.
- Computer A sends packets to B; B returns an acknowledgement packet immediately to confirm that B has received the packet;

  A waits for acknowledgement before sending a new packet; if no acknowledgement comes after a delay *T1*, then A retransmits

22

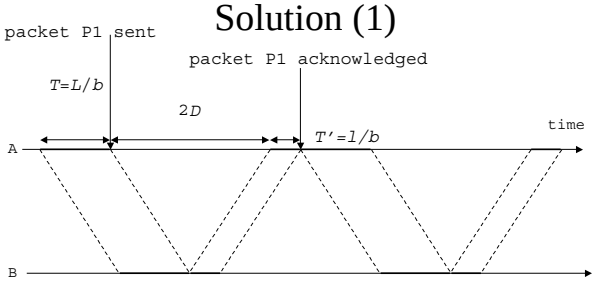
A Simple Protocol: Stop and Go

- **Question**: What is the maximum throughput assuming that there are no losses?

  notation:
  - packet length = *L*, constant (in bits);
  - acknowledgement length = *l*, constant
  - channel bit rate = *b*;
  - propagation = *D*
  - processing time = 0

23

Solution (1)



cycle time = $T + 2D + T'$

useful bits per cycle time = L

throughput = $Lb / (L + l + 2Db) = b / (\omega + \beta/L)$

with $\omega = (L+l)/L$=overhead and $\beta = 2Db$=**bandwidth-delay product**

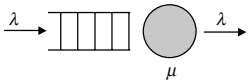
24

Solution (2)

distance	20 km	20000 km	2 km	20 m
bit rate	10kb/s	1 Mb/s	10 Mb/s	1 Gb/s
propagation	0.1ms	100 ms	0.01 ms	0.1µs
transmission	800 ms	8 ms	0.8 ms	8 µs
reception time	800.1 ms	108 ms	0.81 ms	8.1 µs
	modem	*satellite*	*LAN*	*Hippi*
$\beta = 2Db$	2 bits	200 000 bits	200 bits	200 bits
throughput = $b \times$	99.98%	3.8%	97.56%	97.56%

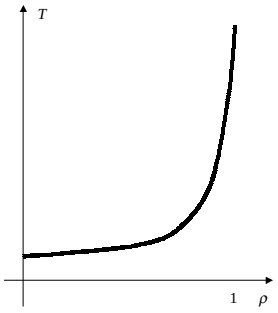
Temps d'attente

- File d'attente M/M/1
  - temps distribués exponentiellement
  - taux d'arrivée λ, taux de service μ,
  - nombre de clients N, délai T



$$N = \frac{\rho}{(1-\rho)} \qquad T = \frac{1}{\mu(1-\rho)} \qquad T = \frac{N}{\lambda}$$

Délai



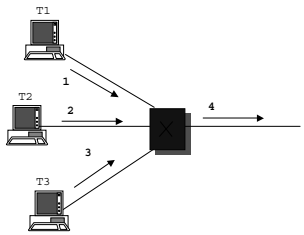
Performances

- Paquet de 1500 octets (en moyenne)
  - liaison de 1 Mbit/s
    - » temps de transmission 12 ms
    - » taux de service 83 paq/s

λ [p/s]	10	40	60	70
$1/\lambda$ [ms]	100	25	16	14
T [ms]	13	23	43	76

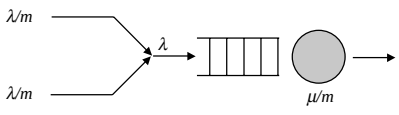
Statistical and Non-statistical Multiplexing

- Non-statistical multiplexing
  - several sources use the same link
- Statistical multiplexing
  - the bit rate is less than the sum of the incoming bit rates
  - may produce packet loss; requires congestion control



Non-statistical Multiplexing

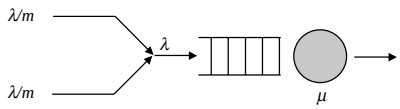
- m sources with rate λ/m
- simple model M/M/1
  - service rate μ/m



$$T = \frac{m}{\mu - \lambda}$$

Statistical Multiplexing

- m sources with rate λ/m
- simple model M/M/1
  - service rate μ



$$T = \frac{1}{\mu - \lambda}$$

31

Statistical and Non-statistical Multiplexing

- Non-statistical multiplexing examples
  - Frequency Division Multiplexing, Time Division Multiplexing
  - telecommunication networks, GSM
- Statistical multiplexing examples
  - Ethernet, IP, ATM

32

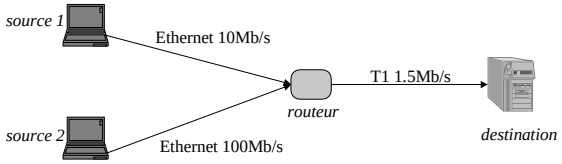
Partie 2
Contrôle de trafic

Taxonomie de mécanismes

Critères : efficacité et équité

33

Contrôle de trafic



- Comment sont allouées les ressources du réseau
  - débits des liaisons
  - tampons des routeurs ou des commutateurs

34

Contrôle de trafic

- Problème à résoudre au niveau réseau et transport
- Ressources partagées et réparties
  - peu d'utilisateurs → sous-utilisation
  - trop d'utilisateurs → congestion
- Allocation de ressources
  - réservation
    - » ressources suffisantes
    - » difficile à faire : ressources sont réparties
  - pas de réservation
    - » congestion et récupération

35

Contrôle de trafic

- Mécanismes internes (dans le réseau)
  - politique d'ordonnancement dans les routeurs
  - gestion de la file d'attente
- Mécanismes externes (dans les hôtes)
  - limitation du débit des sources
  - lissage du trafic à l'entrée
- Contrôle de flux vs. contrôle de congestion
  - contrôle de flux : adaptation de la vitesse de réception et d'émission
  - contrôle de congestion : éviter l'encombrement du réseau

36

Notion de flot

- Flot
  - séquence de paquets envoyée par une source à une destination empruntant une route
- Circuit virtuel
  - établissement explicite
  - réservation des ressources
- Datagrammes
  - établissement implicite
  - pas de réservation des ressources

37

Taxonomie des mécanismes

- Dans le réseau (ATM)
  - routeur/commutatuer décide quel paquet servir ou jeter
  - routeur signale à la source à quel débit elle peut émettre
- Boucle ouverte (ATM)
  - réservation des ressources
  - contrôle d'admission
- Dans les hôtes (TCP)
  - hôte observe le réseau et ajuste le débit d'émission
- Boucle de rétroaction
  - information sur l'état de congestion
    - » explicite (RTCP)
    - » implicite - pertes (TCP)

38

Taxonomie des mécanismes

- Débit d'émission *(rate)*
  - négocié avec le réseau
  - peut être ajusté si nécessaire
  - ATM, RTP
- Fenêtre ou crédit
  - volume de données à émettre
  - TCP
- Réservation (boucle ouverte) implique
  - un mécanisme implanté dans les routeurs
  - un contrôle du débit d'émission

39

Critères d'évaluation

- Efficacité
  - meilleure allocation des ressources
- Équité
  - partage équitable

40

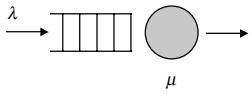
Efficacité

- Deux métriques essentielles
  - débit → utilisation 100%
  - délai → utilisation 0%
- Optimiser les deux ?
  - métrique puissance *(power)*
    - » puissance = débit/délai

41

Puissance

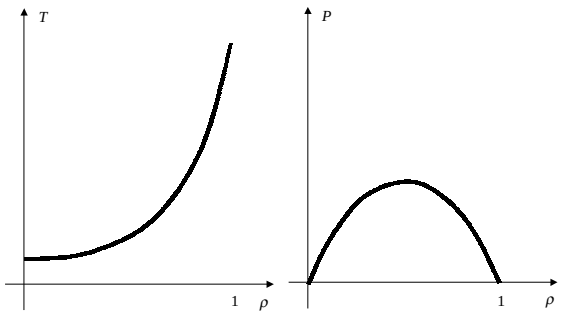
- File d'attente M/M/1
  - temps distribués exponentiellement
  - débit : λ
  - délai : T



$$T = \frac{1}{\mu(1-\rho)} \qquad P = \mu^2 \rho(1-\rho)$$

42

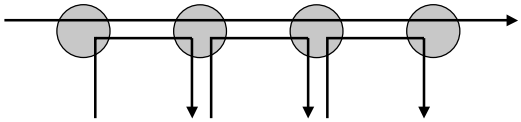
Puissance



43

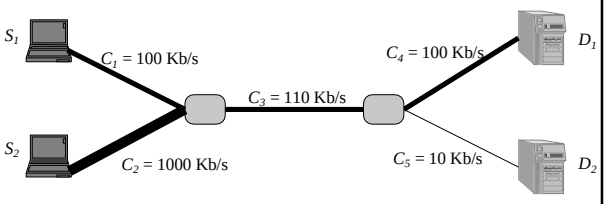
Équité

- Comment se fait le partage ?
  - parts égales
  - parts proportionnelles aux débits des sources
- Scénario de parking *(parking lot)*
  - débit est maximal, si le flot horizontal = 0



44

Contrôle de trafic - exemple



- Allocation des débits
  - si la somme de débits dépasse la capacité de la liaison, les sources doivent réduire leurs trafics proportionnellement à leurs débits
  - hypothèse vrai, si FIFO dans les routeurs

Allocation des débits

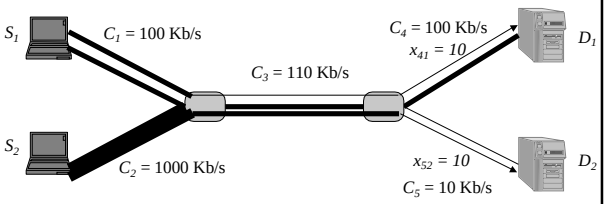
- Débit x_{ls} : source s sur liaison l
- Trafic λ_s : généré par source s
- Allocation

$x_{11} = \min(\lambda_1, C_1)$
$x_{22} = \min(\lambda_2, C_2)$
$x_{3i} = \min(x_{ii}, C_3 x_{ii}/(x_{11} + x_{22}))$
$x_{41} = \min(x_{31}, C_4)$
$x_{52} = \min(x_{32}, C_5)$
débit $\theta = x_{41} + x_{52}$

Application numérique :

$x_{11} = 100$
$x_{22} = 1000$
$x_{31} = 110 \times 100/1100 = 10$
$x_{32} = 110 \times 1000/1100 = 100$
$x_{41} = 10$
$x_{52} = 10$
débit $\theta = 20$ Kb/s

Contrôle de trafic - exemple

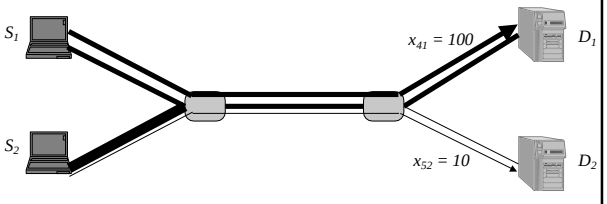


- S_1 envoie 10 Kb/s à cause de la compétition avec S_2 sur la liaison 3, S_2 est limitée à cause de la liaison 5

Contrôle de trafic - exemple

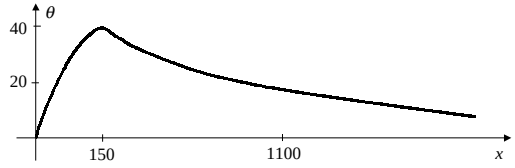
- Augmenter le débit ?
  - S_2 voit la situation et veut coopérer (optimisation globale)
  - S_2 réduit x_{22} à 10 Kb/s, parce que de toute façon, elle ne peut pas dépasser 10 Kb/s sur liaison 5
  - x_{31} = 100 Kb/s et x_{41} = 100 Kb/s sans pénaliser S_2
  - θ = 110 Kb/s

Contrôle de trafic - exemple



- Utilisation optimale des liaisons

Débit en fonction de la charge



- $\lambda_1 = \lambda$, $\lambda_2 = \lambda^2/10$, λ - un paramètre
- $\lambda_1(1) = 1$, $\lambda_2(1) = 1/10$
- $\lambda_1(10) = 10$, $\lambda_2(10) = 10$
- $\lambda_1(100) = 100$, $\lambda_2(100) = 1000$
- charge offerte $x = \lambda_1 + \lambda_2$

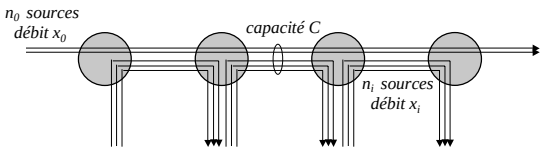
Critère d'efficacité

- Dans un réseau de commutation de paquets des sources doivent limiter leurs débits d'émission en prenant en compte l'état d'encombrement du réseau, sinon l'efficacité n'est pas optimale
- Objectif des mécanismes du contrôle de trafic
  - éviter cette inefficacité

51

Équité vs. efficacité

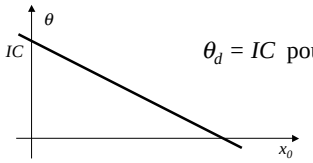
- Scénario de parking *(parking lot)*
  - capacité des liaisons : C
  - n_i sources, débit x_i, $i = 1, ..., I$
  - trafic sur liaison i : $n_0 x_0 + n_i x_i$



52

Débit maximal

- Pour n_0 et x_0 donnés, il faut que
  - $n_i x_i = C - n_0 x_0$
- Débit total mesuré à la sortie du réseau
  - $\theta = n_0 x_0 + \Sigma\, n_i x_i = n_0 x_0 + \Sigma(C - n_0 x_0) =$
    $= n_0 x_0 + I(C - n_0 x_0) = IC - (I - 1)\, n_0 x_0$



$\theta_d = IC$ pour $x_0 = 0$!

53

Équité

- Optimisation du débit
  - partage de la capacité non-équitable
    - » certaines sources (type 0) doivent réduire leur débit à 0 !
- Critère d'équité
  - allouons la même partie de la capacité à toutes les sources, par exemple pour $n_i = 1$
    - » $x_i = C/2$
    - » $\theta_{éq} = (I+1)C/2$

54

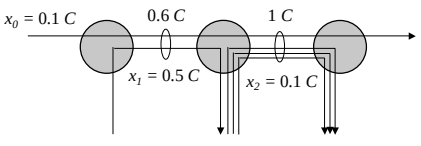
Optimisation de l'équité

- On reprend pour n_i quelconque
  - allocation équitable sur liaison i
    - » $x_i = C/(n_0 + n_i)$, $i = 1, ..., I$
  - on diminue x_0 pour augmenter θ
    - » $x_0 = \min C/(n_0 + n_i)$,
  - exemple
    - » I = 2, $n_0 = n_1 = 1$, $n_2 = 9$
    - » liaison 2 : $x_2 = C/(1 + 9) = 0.1\, C$
    - » liaison 1 : $x_1 = C/(1 + 1) = 0.5\, C$
    - » $x_0 = \min(0.5\, C, 0.1\, C) = 0.1\, C$

55

Exemple

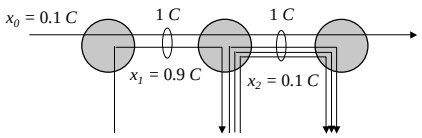
- Problème
  - liaison 1 : 0.6 C
    - » sous-utilisée
  - liaison 2 : 1 C



56

Équité Max-Min

- On peut augmenter x_1 sans diminuer les parts des autres flots
  - » $x_0 = 0.1\ C$, $x_1 = 0.9\ C$, $x_2 = 0.1\ C$



57

Équité Max-Min

- Allouer des ressources en partageant équitablement n'est pas une solution optimale
  - – certaines sources peuvent obtenir plus de ressources sans diminuer les parts des autres sources
  - – allocation Max-Min
    - » Min : pour l'équité sur des liaisons critiques
    - » Max : pour augmenter certaines parts, si possible

58

Allocation "remplissage progressif"

- Liaison critique *(bottleneck link) l* pour source *s*
  - – liaison *l* est saturée : $\Sigma\ x_i = C$
  - – le débit de source *s* sur liaison *l* est maximal
- Allocation "remplissage progressif"
  - – $x_i = 0$
  - – augmenter x_i équitablement jusqu'à $\Sigma\ x_i = C$
  - – on n'augmente plus les sources qui utilisent cette liaison (pour elles, c'est une liaison critique)
  - – on continue à augmenter le débit des autres sources sur les autres liasons

59

Exemple d'allocation

- Scénario de parking
  - – $x_i = 0$
  - – $x_i = d$ jusqu'à $n_0 x_0 + n_i x_i \leq C$
  - – liaison critique pour $d_1 = \min (C/(n_0 + n_i))$, source 0 ou *i*
    - » $x_0 = \min (C/(n_0 + n_i))$ $\quad x_0 = 0.1\ C$, $x_2 = 0.1\ C$
  - – augmenter les autres sources
    - » $x_i = (C - n_0 x_0)/ n_i$ $\quad x_1 = 0.9\ C$

60

Proportional Fairness

- An allocation of rates x_s is ``proportionally fair" if and only if, for any other feasible allocation y_s we have

$$\sum_{s=1}^{S} \frac{y_s - x_s}{x_s} \leq 0$$

- ❏ Any change in the allocation must have a negative average change
- ❏ Parking lot example with $n_s = 1$
  - max-min fair allocation $x_s = C/2$ for all s
  - let decrease x_0 by δ : $y_0 = C/2 - \delta$, $y_s = C/2 + \delta$, $s = 1, \ldots, I$
  - average rate of change is

$$\left(\sum_{s=1}^{I} \frac{2\delta}{C} \right) - \frac{2\delta}{C} = \frac{2(I-1)\delta}{C}$$

  - not proportionally fair for $I \geq 2$!

61

Proportional Fairness

- There exists one unique proportionally fair allocation. It is obtained by maximizing

$$J(\vec{x}) = \sum_s \ln(x_s)$$

over the set of feasible allocations

62

Parking lot example

- For any choice of x_0 we should set x_i such that
  $n_0 x_0 + n_i x_i = C$, $i = 1, \ldots, I$
- Maximize

$$f(x_0) = n_0 \ln(x_0) + \sum_{i=1}^{I} n_i (\ln(C - n_0 x_0) - \ln(n_i))$$

  over the set $0 \le x_0 \le C/n_0$.
- The maximum is for

$$x_0 = \frac{C}{\sum_{i=0}^{I} n_i} \qquad x_i = \frac{C - n_0 x_0}{n_i}$$

- If $n_i = 1$, $x_0 = C/(I+1)$, $x_i = CI/(I+1)$
- Max-min allocation is $C/2$ for all rates - sources of type 0 get a smaller rate, since they use more network resources

63

Additive increase, Multiplicative decrease

- End-to-end congestion control
  - binary feedback
  - adaptation mechanism of additive increase, multiplicative decrease
- Modeling
  - I sources, rate $x_i(t)$, $i = 1, \ldots, I$
  - link capacity : c
  - discrete time, feedback cycle = one time unit
  - during one time cycle, the source rates are constant, and the network generates a binary feedback signal $y(t) \in \{0, 1\}$
  - sources: increase the rate if $y(t) = 0$ and decrease if $y(t) = 1$
  - feedback $y(t) = [if\ (\sum_{i=1}^{I} x_i(t) \le c)\ then\ 0\ else\ 1]$

64

Linear adaptation algorithm

- Find constants u_0, u_1, v_0, v_1, such that

$$x_i(t+1) = u_{y(t)}\, x_i(t) + v_{y(t)}$$

  - converge towards a fair allocation
  - one single bottleneck, so all fairness criteria are equivalent
  - we should have $x_i = c/I$
  - the total rate

$$f(t) = \sum_{i=1}^{I} x_i(t)$$

    should oscillate around c: it should remain below c until it exceeds it once, then return below c

65

Necessary conditions

$$f(t+1) = u_{y(t)}\, f(t) + v_{y(t)}$$

- we must have

  $u_0 f + v_0 > f$, increase

  $u_1 f + v_1 < f$, decrease
- this gives the following conditions

  $u_1 < 1$ and $v_1 \le 0$

  or

  $u_1 = 1$ and $v_1 < 0$

  and

  $u_0 > 1$ and $v_0 \ge 0$

  or

  $u_0 = 1$ and $v_0 > 0$

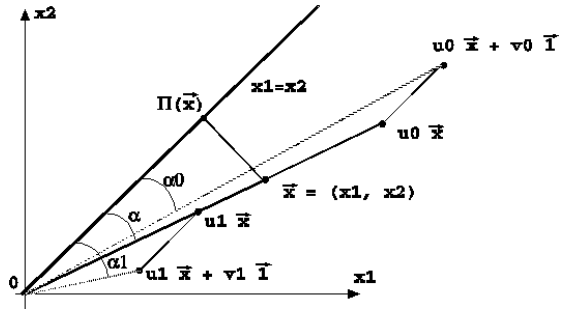
66

Ensure fairness

- we need to measure how much a given rate allocation deviates from fairness
- measure of unfairness:
  - » the distance between the rate allocation $\vec{x}$ and its nearest fair allocation $\Pi(\vec{x})$, where $\Pi(\vec{x})$ is the orthogonal projection on the set of fair allocations, normalized by the length of the fair allocation

67

Effect on fairness of an increase or a decrease



68

Ensure fairness

- when we apply a multiplicative increase or decrease, the unfairness is unchanged
- an additive increase decreases the unfairness, whereas an additive decrease increases the unfairness
- to obtain that unfairness decreases or remains the same, and such that in the long term it decreases

  $v_1 = 0$ decrease must be multiplicative

  $u_0 = 1$ increase must be additive

69

Additive increase, Multiplicative decrease

- Fact
  - In order to satisfy efficiency and convergence to fairness, we must have a multiplicative decrease (namely, $u_0 = 1$ and $v_1 = 0$ and a non-zero additive component in the increase (namely, $u_0 \geq 1$ and $v_0 > 0$).
  - If we want to favour a rapid convergence towards fairness, then the increase should be additive only (namely, $u_0 = 1$ and $v_0 > 0$).

70

Facts to remember

- In a packet network, sources should limit their sending rate by taking into consideration the state of the network
- Maximizing network throughput as a primary objective may lead to gross unfairness
- Objective of congestion control is to provide both efficiency and some form of fairness
- Fairness can be defined in various ways: max-min, proportional
- End-to-end congestion control in packet networks is based on binary feedback and the adaptation mechanism of additive increase, multiplicative decrease.

71

72

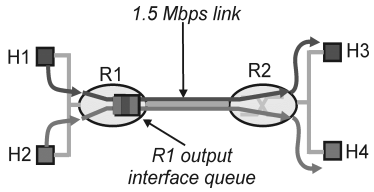
Parite 3 : Quality of Service in IP Networks

Principles of QoS
Traffic shaping
Scheduling mechanisms
IntServ
DiffServ

73

Improving QOS in IP Networks

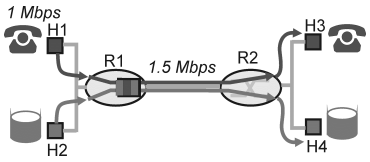
- IETF groups are working on proposals to provide better QOS control in IP networks, i.e., going beyond best effort to provide some assurance for QOS
- Work in Progress includes RSVP, Differentiated Services, and Integrated Services
- Simple model for sharing and congestion studies:



74

Principles for QOS Guarantees

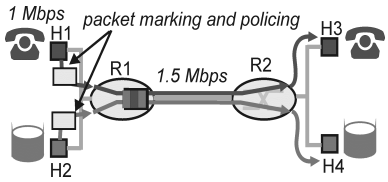
- Consider a phone application at 1Mbps and an FTP application sharing a 1.5 Mbps link.
  - bursts of FTP can congest the router and cause audio packets to be dropped.
  - want to give priority to audio over FTP
- **PRINCIPLE 1: Marking of packets is needed for router to distinguish between different classes; and new router policy to treat packets accordingly**



75

Principles for QOS Guarantees (more)

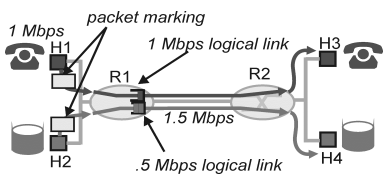
- Applications misbehave (audio sends packets at a rate higher than 1Mbps assumed above);
- **PRINCIPLE 2: provide protection (isolation) for one class from other classes**
- Require Policing Mechanisms to ensure sources adhere to bandwidth requirements; Marking and Policing need to be done at the edges:



76

Principles for QOS Guarantees (more)

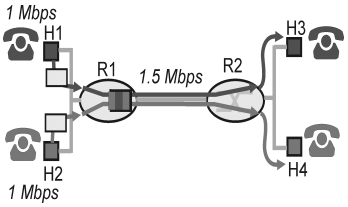
- Alternative to Marking and Policing: allocate a set portion of bandwidth to each application flow; can lead to inefficient use of bandwidth if one of the flows does not use its allocation
- **PRINCIPLE 3: While providing isolation, it is desirable to use resources as efficiently as possible**



77

Principles for QOS Guarantees (more)

- Cannot support traffic beyond link capacity
- **PRINCIPLE 4: Need a Call Admission Process; application flow declares its needs, network may block call if it cannot satisfy the needs**



78

Lissage des sources *(traffic shaping)*

- Comment prévenir la congestion ?
  - congestion peut être provoquée par l'irrégularité du trafic
  - arrivées plus déterministes, meilleures performances
    - exemple : nb. de clients D/D/1 vs. G/D/1
  - contrôler le débit et la taille de rafales *(bursts)*
    - description du trafic
- Contrat de service
  - si le réseau cannait le type de trafic, il peut gérer mieux les ressources
  - contrat entre la source et le réseau
    - source : description du trafic
    - réseau : garantie de la QoS si le trafic se conforme à la description
    - si le trafic n'est pas conforme, punition : rejet du paquet ou pas de garanties *(traffic policing)*

79

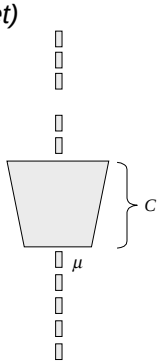
Policing Mechanisms

- Three criteria:
  - (Long term) **Average Rate** (100 packets per sec or 6000 packets per min?), crucial aspect is the interval length
  - **Peak Rate**: e.g., 6000 p/minute Avg and 1500 p/sec Peak
  - (Max.) **Burst Size**: Max. number of packets sent consecutively, ie. over a short period of time

80

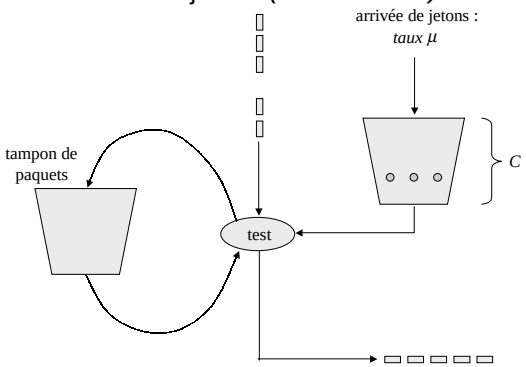
Seau troué *(leaky bucket)*

- Tampon de taille limitée avec un taux de sortie constant
  - μ si le tampon non-vide
  - 0 si le tampon vide
- Équivalent à une file G/D/1/N
- Paquets de taille fixe
  - un paquet par top d'horloge
- Paquets de taille variable
  - no. d'octets par top d'horloge
- Pertes si débordement



81

Seau à jetons *(token bucket)*



82

Seau à jetons *(token bucket)*

- Jetons générés avec le taux μ
  - 1 jeton : 1 paquet ou k octets
- Paquet doit attendre un jeton avant l'émission
  - pas de pertes
  - permet des rafales limitées (un peu plus que C)
- Quand paquets ne sont pas générés, accumulation des jetons
  - n jetons - rafale de n paquets
  - si le seau déborde, les jetons sont perdus
- Débit moyen de sortie : μ
- Délai limité par C/μ (formule de Little)

83

Exemple

- Source génère de données à 25 Mo/s
- Réseau peut supporter 25 Mo/s, mais il est préférable d'utiliser 2 Mo/s en continu
- Données
  - 1 Mo toutes les 40 ms pendant 1 s
- Exemple
  1. seau troué avec C = 1 Mo, p = 25 Mo/s, μ = 2 Mo/s
  2. seau à jeton avec C = 250 Ko, p = 25 Mo/s, μ = 2 Mo/s
  3. seau à jeton avec C = 500 Ko, p = 25 Mo/s, μ = 2 Mo/s
  4. seau à jeton avec C = 750 Ko, p = 25 Mo/s, μ = 2 Mo/s
  5. seau à jeton avec C = 500 Ko, p = 25 Mo/s, μ = 2 Mo/s et seau troué avec C = 1 Mo, p = 25 Mo/s, μ = 2 Mo/s

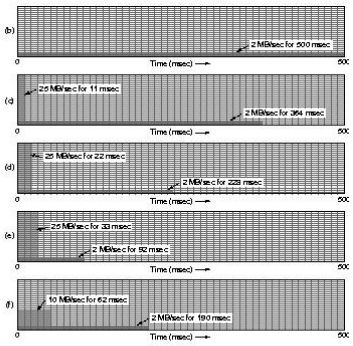
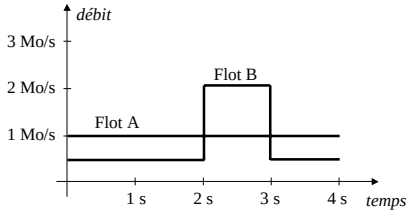


Fig. 5-25. (a) Input to a leaky bucket. (b) Output from a leak bucket. (c) - (e) Output from a token bucket with capacities 250KB, 500KB, and 750KB. (f) Output from a 500KB toke bucket feeding a 10 MB/sec leaky bucket.

Durée de rafale

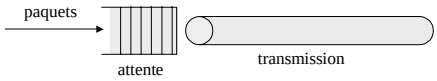
- Durée de rafale - S sec
- Capacité du seau - C octets
- Taux maximal de sortie - p octets/s
- Taux d'arrivée de jetons - μ octets /s
  - rafale de $C + \mu S$ octets
  - rafale de pS
  - $C + \mu S = pS \rightarrow S = C/(\mu - p)$
- Exemple
  - C = 250 Ko, p = 25 Mo/s, μ = 2 Mo/s
  - S = 11 ms

Description du trafic



- Flot A : μ = 1 Mo/s, C = 1 octet
- Flot B : μ = 1 Mo/s, C = 1 Moctet
  - pendant 2 s, il économise 2 s $\times$ 0.5 Mo/s = 1 Moctet

Politiques d'ordonnancement



- Rôle de l'ordonnanceur
  - définir l'ordre de transmission des paquets
- Algorithmes d'allocation
  - du débit
    - quel paquet est choisi pour être transmis
  - des tampons
    - quel paquet est rejeté

PAPS - FIFO

- Premier paquet arrivé, premier transmis
  - derniers paquets sont rejetés
- FIFO transfère la responsabilité de la gestion de congestion aux stations
  - état actuel de l'Internet
  - TCP ajuste le débit en fonction des pertes
- Découpler l'ordre de transmission et le rejet
  - techniques RED *(Random Early Discard)*
    - choisir au hasard un paquet dans la file et le rejeter

Caractéristiques du FIFO

- Permet de partager le débit
  - proportionnellement au débit d'arrivée
- Pas d'isolation
  - flots élastiques (débit contrôlé par la source, p.ex. TCP) peuvent subir l'influence des autres flots
    - flot UDP qui maintient un débit source malgré les pertes, peut obtenir une part importante de débit
    - flots temps réel (transfert multimédia) peuvent subir des retards à cause des files d'attente

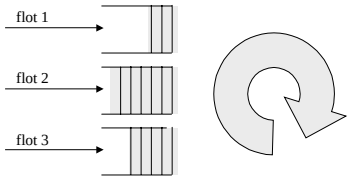
90

File avec priorités

- Plusieurs files de priorité différente
  - sources marquent leurs paquets avec la priorité
    - exemple : champ TOC de IP (3 bits de priorité)
  - paquets de même priorité servis en FIFO
  - après tous les paquets prioritaires, on sert les paquets moins prioritaires
- Problème
  - comment empêcher que toutes les sources envoient des paquets de priorité maximale ?

91

Tourniquet *(Round Robin)*



- File qui ressemble à *Processor Sharing* ou au Temps Partagé
  - une file par flot
  - servies de manière cyclique, un paquet par file

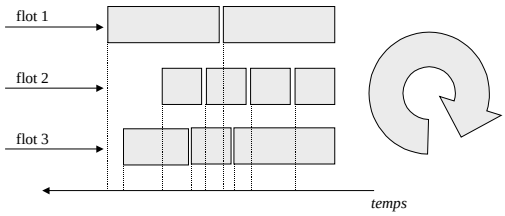
92

Caractéristiques du tourniquet

- Changer la stratégie optimale
  - FIFO : transmettre le plus possible
  - Tourniquet : utiliser au mieux sa part du débit
    - si la source envoie plus, sa file croît : le délai plus grand, la probabilité de perte plus grande
- Isolation
  - "bonnes" sources sont protégées contre les "mauvaises"
- Problème
  - flots de paquets larges obtiennent plus !
  - coût de la classification des flots

93

File équitable *(Fair Queueing)*



- Tourniquet "bit par bit"
  - chaque paquet est marqué par la date de transmission de son dernier bit
  - ordre des dates

94

File équitable avec poids *(Weighted Fair Queueing)*

- File équitable
  - parts égales du débit : 1/*n*
- File équitable avec poids
  - chaque flot a droit d'envoyer un nombre de bits à chaque tour
- Exemple - file équitable avec poids w_i
  - flot 1 poids 2 — flot 2 poids 1 — flot 3 poids 3
  - 1/3 — 1/6 — 1/2

$$x_i = D\frac{w_i}{\sum w_i}$$

D : le débit de la liaison de sortie

95

Garantie de débit

- Poids exprimés comme proportions (w_i - le poids garanti)

$$x_i = D\frac{w_i}{\sum w_i}, \sum w_i \leq 1$$

$$x_i \geq D \times w_i$$

- Poids pour garantir un débit

$$w_i = x_i / D$$

96

Garantie de délai

- Flot caractérisé par un seau à jeton *(token bucket)*
  - débit μ, capacité C
  - délai limité par C/μ
- Si $x_i > \mu$ (part du débit est suffisante pour le flot)
  - délai limité par C/μ
  - délai total limité par la somme de tous les délais

97

Gestion des files d'attente *(Random Early Detection)*

- Famille de techniques pour détecter la congestion et la signaler à la source
  - quand la file déborde, rejet des paquets entrants
  - pertes des paquets interprétées comme le signalement de la congestion → limitation du débit
- Idée
  - agir avant la congestion pour demander la limitation du débit par les sources
  - un seuil au-delà duquel on commence à rejeter des paquets

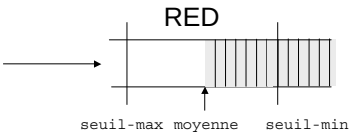
98

Motivation

- Pertes sont inefficaces
  - provoquent des retransmissions - les paquets rejetés doivent être retransmis
  - entrée en phase Démarrage Lent *(Slow Start)*
- Synchronisation des sources TCP
  - rejet de plusieurs paquets, quand la file est pleine
  - plusieurs sources détectent la congestion et entrent en phase Démarrage Lent au même moment

99

RED



- Estimation de la moyenne
  - `moyenne` ← `q` × `mesure + (1 - q)` × `moyenne`
- Si `moyenne` ≤ `seuil-min`
  - accepte le paquet
- Si `seuil-max` < `moyenne` < `seuil-min`
  - rejette le paquet avec probabilité `p`
- Si `seuil-max` ≤ `moyenne`
  - rejette le paquet

100

Caractéristiques du RED

- Il maintient la longueur de file raisonnable
  - délai peut être faible
- Convient bien à TCP
  - une seule perte est récupérée facilement par *Fast Retransmit*
- Probabilité `p` de choisir un flot donné est proportionnelle à la part du débit que reçoit ce flot
  - plus de paquets envoyés par un flot, plus de chances de choisir un de ses paquets pour le rejet

101

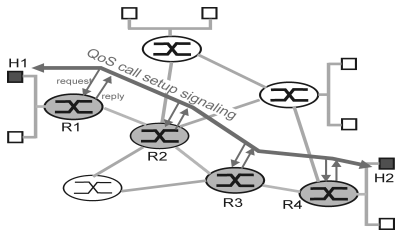
Caractéristiques du RED

- Probabilité `p` dynamique
  - `p-tmp = max-p × (moyenne - seuil-min)/ (seuil-max - seuil-min)`
  - `p = p-tmp/(1 - nb-paquets × p-tmp)`
  - `nb-paquets` : combien de paquets ont été acceptés quand `moyenne` était entre deux seuils
  - `p` croît lentement avec `nb-paquets`
- Exemple
  - `max-p` = 0.02
  - `moyenne` au milieu de deux seuils, 1 rejet sur 50

Integrated Services

102

- An architecture for providing QOS guarantees in IP networks for individual application sessions
- relies on resource reservation, and routers need to maintain state info (Virtual Circuit??), maintaining records of allocated resources and responding to new Call setup requests on that basis



Call Admission

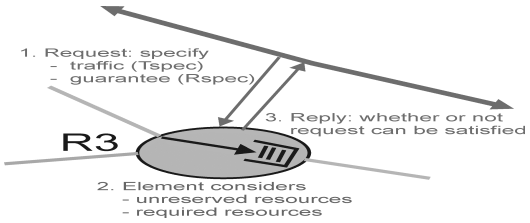
103

- Session must first declare its QOS requirement and characterize the traffic it will send through the network
- **R-spec**: defines the QOS being requested
- **T-spec**: defines the traffic characteristics
- A signaling protocol is needed to carry the R-spec and T-spec to the routers where reservation is required; RSVP is a leading candidate for such signaling protocol

Call Admission

104

- Call Admission: routers will admit calls based on their R-spec and T-spec and base on the current resource allocated at the routers to other calls.



Integrated Services: Classes

105

- **Guaranteed QOS**: this class is provided with firm bounds on queuing delay at a router; envisioned for hard real-time applications that are highly sensitive to end-to-end delay expectation and variance
- **Controlled Load**: this class is provided a QOS closely approximating that provided by an unloaded router; envisioned for today's IP network real-time applications which perform well in an unloaded network

Differentiated Services

106

- Intended to address the following difficulties with Intserv and RSVP;
- **Scalability**: maintaining states by routers in high speed networks is difficult sue to the very large number of flows
- **Flexible Service Models**: Intserv has only two classes, want to provide more qualitative service classes; want to provide 'relative' service distinction (Platinum, Gold, Silver, …)
- **Simpler signaling**: (than RSVP) many applications and users may only w ant to specify a more qualitative notion of service

Differentiated Services

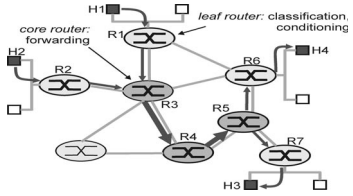
107

- Approach:
  - Only simple functions in the core, and relatively complex functions at edge routers (or hosts)
  - Do not define service classes, instead provides functional components with which service classes can be built

Edge Functions

108

- At DS-capable host or first DS-capable router
- **Classification**: edge node marks packets according to classification rules to be specified (manually by admin, or by some TBD protocol)
- **Traffic Conditioning**: edge node may delay and then forward or may discard



Core Functions

109

- **Forwarding**: according to "Per-Hop-Behavior" or PHB specified for the particular packet class; such PHB is strictly based on class marking (no other header fields can be used to influence PHB)
- BIG ADVANTAGE:

  No state info to be maintained by routers!

Classification and Conditioning

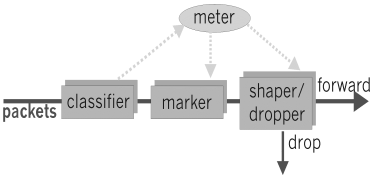
110

- Packet is marked in the Type of Service (TOS) in IPv4, and Traffic Class in IPv6
- 6 bits used for Differentiated Service Code Point (DSCP) and determine PHB that the packet will receive
- 2 bits are currently unused

Classification and Conditioning

111

- It may be desirable to limit traffic injection rate of some class; user declares traffic profile (eg, rate and burst size); traffic is metered and shaped if non-conforming



Forwarding (PHB)

112

- PHB result in a different observable (measurable) forwarding performance behavior
- PHB does not specify what mechanisms to use to ensure required PHB performance behavior
- Examples:
  - Class A gets x% of outgoing link bandwidth over time intervals of a specified length
  - Class A packets leave first before packets from class B

Forwarding (PHB)

113

- PHBs under consideration:
  - **Expedited Forwarding**: departure rate of packets from a class equals or exceeds a specified rate (logical link with a minimum guaranteed rate)
  - **Assured Forwarding**: 4 classes, each guaranteed a minimum amount of bandwidth and buffering; each with three drop preference partitions

114

Differentiated Services Issues

- AF and EF are not even in a standard track yet… research ongoing
- "Virtual Leased lines" and "Olympic" services are being discussed
- Impact of crossing multiple ASs and routers that are not DS-capable

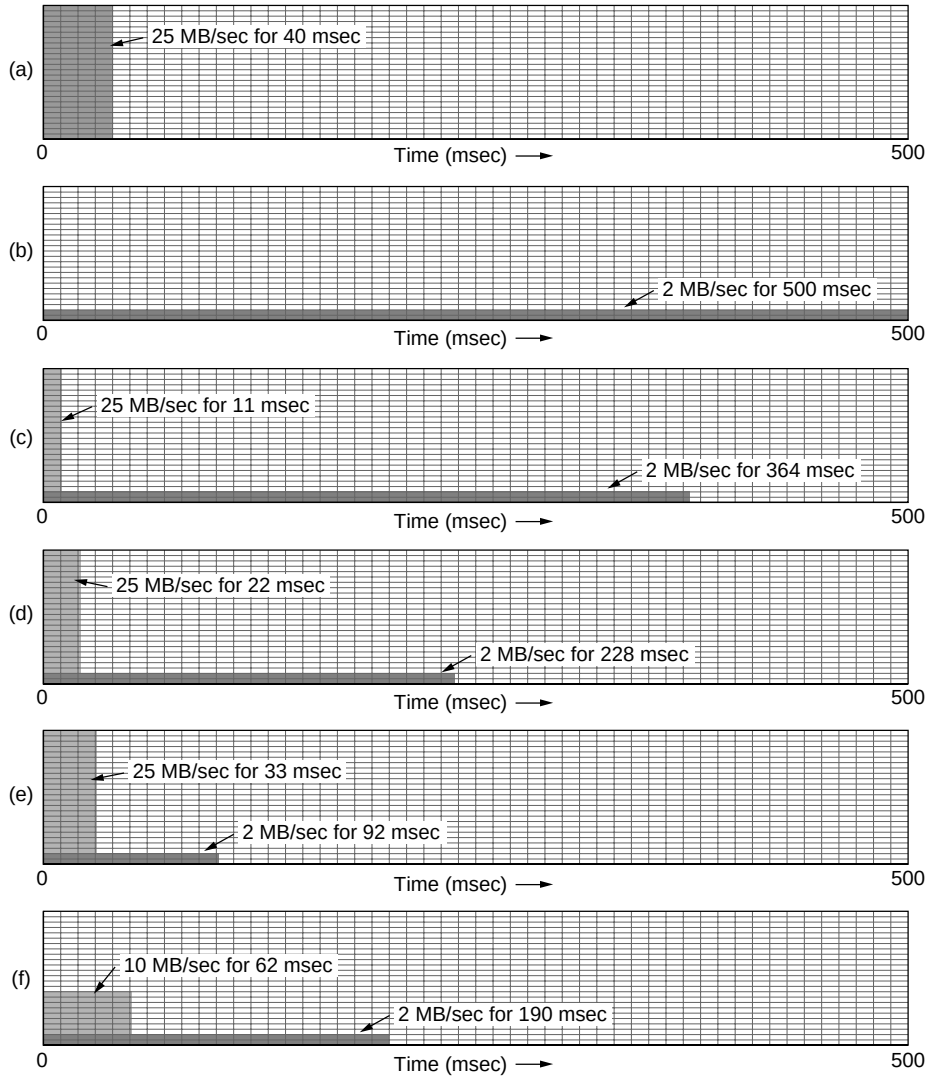


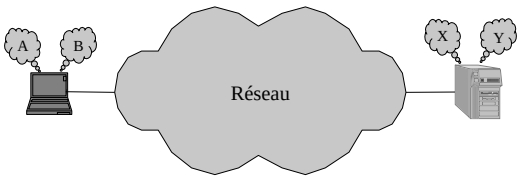
Fig. 5-25. (a) Input to a leaky bucket. (b) Output from a leaky bucket. (c) - (e) Output from a token bucket with capacities of 250KB, 500KB, and 750KB. (f) Output from a 500KB token bucket feeding a 10 MB/sec leaky bucket.

Partie 4
Algorithmes avancés du TCP

Couche transport
Fonctions de TCP
Contrôle d'erreurs
Contrôle de flux
Contrôle de congestion

115

Couche transport



- Deux entités connectés directement par la couche réseau
  - ressemblance avec la couche liaison
- Transfert de données entre deux processus selon une qualité de service demandée

116

Couche de transport

- Fonctions
  - pallier les imperfections des couches plus basses
    - » pertes, erreurs, paquets en désordre
  - ajuster les vitesses d'envoi et de réception
    - » contrôle de flux
  - traiter des cas de congestion dans le réseau
  - optimiser les performances du transfert

117

Couche de transport

- Adressage
  - station, port
- Multiplexage
  - plusieurs connexions transport utilisant la même adresse réseau
- Garanties
  - sans connexion (UDP)
  - avec connexion (TCP)
  - qualité de service
    - » délai, gigue, débit

118

Problèmes

- Connexion
  - problème d'une entente répartie en présence de pertes, duplication ou rétention temporaire de paquets
- Transfert - les techniques de la couche de liaison
  - fenêtre d'anticipation
  - retransmission
  - contrôle de flux par des crédits
    - » la fenêtre d'émission peut être modifiée par le récepteur

119

TCP *(Transmission Control Protocol)*

- Fonction
  - transfert d'une séquence d'octets
    - » pas de marquage de messages
- Unité de protocole
  - segment
- Phases
  - connexion
  - transfert
  - fermeture

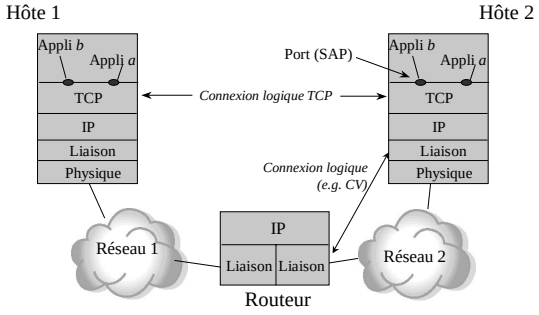
120

TCP *(Transmission Control Protocol)*

- Fiabilité
  - fenêtre d'anticipation
  - détection d'erreurs par le récepteur
  - retransmission continue *(Go-back-N)*, et retransmission sélective (stockage en désordre)
    - » heuristiques *"retransmission rapide"*
- Contrôle de flux
  - fenêtre modulée par récepteur (crédit)
- Contrôle de congestion
  - adaptation à l'état d'occupation du réseau

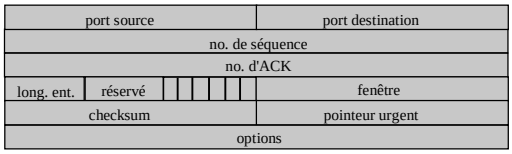
121

Schéma général



122

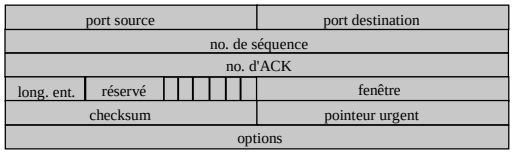
En-tête TCP



- No. de séquence
  - no. du premier octet de données
- No. d'ACK
  - no. de l'octet attendu

123

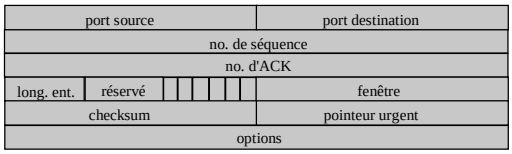
En-tête TCP



- Bits - *flags*
  - SYN - segment de connexion
  - ACK - no. d'ACK actif
  - FIN - fermeture de connexion

124

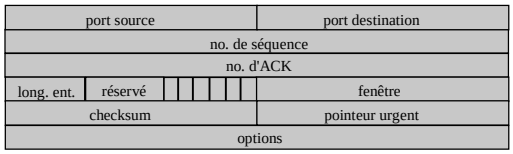
En-tête TCP



- Bits - *flags*
  - URG - pointeur urgent actif
  - RST - *reset*
  - PSH - *push* : force la création d'un segment et et son restitution à l'application

125

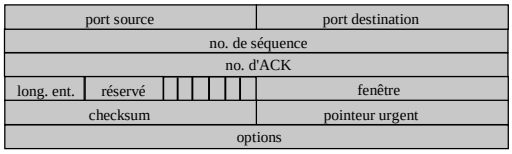
En-tête TCP



- Fenêtre annoncée
  - récepteur contrôle la fenêtre d'émission (par défaut 4 Koctets, max. 64Koctets)
- Checksum
  - sur le pseudo-en-tête (adresses IP), en-tête et les données

126

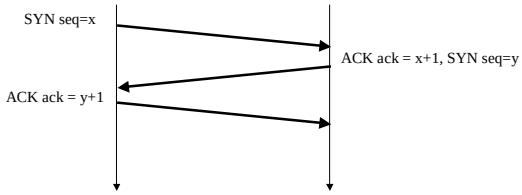
En-tête TCP



- Pointeur urgent
  - indique la fin des données urgentes
- Options
  - MSS *(Maximal Segment Size)* (sans en-tête)
    - défaut 536 octets ; 1460

127

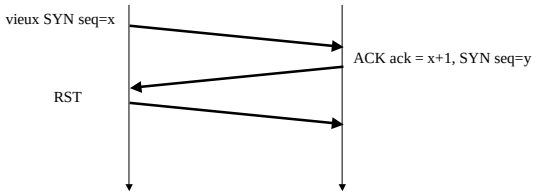
Connexion



- Trois échanges *(three-way handshake)*
  - entente sur les numéros de séquences différents d'une connexion précédente
  - *x, y* : choisis en fonction de l'horloge

128

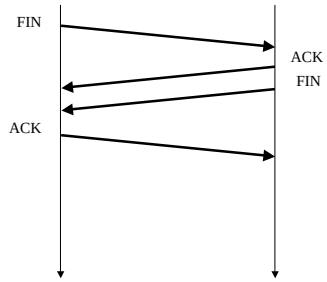
Connexion



- Cas d'un segment retardé

129

Fermeture



- Deux échanges *(two-way handshake)*

130

Segments et octets

- Connexion TCP
  - une séquence ordonnée d'octets
- Segments
  - les octets de données sont accumulés jusqu'au moment où TCP décide d'envoyer un segment
  - découpage en segment indépendant du découpage au niveau application
  - MSS - la longueur maximale de segment

131

Transfert de données

- Contrôle de flux
  - envois des segments par anticipation
  - le récepteur régule la taille de fenêtre - crédit
- Contrôle des erreurs
  - ACK des segments
  - retransmission
- Contrôle de congestion
  - si le réseau est trop encombré, réduire le taux d'envoi

132

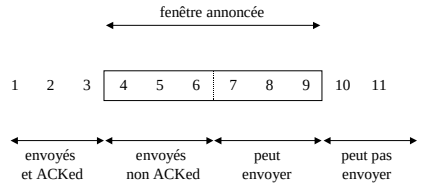
Contrôle de flux

- Contrôle de flux et de congestion par le mécanisme de *fenêtre glissante*
- Le segment ACK=*j* , W=*k* autorise théoriquement l'émetteur à envoyer des octets jusqu'au rang *j* + *k* - 1
- Fenêtre d'émission dépend de
  - la fenêtre annoncée par la destination W
  - la fenêtre de congestion (`cwnd`) discutée plus loin

F = min (`cwnd`, W)

133

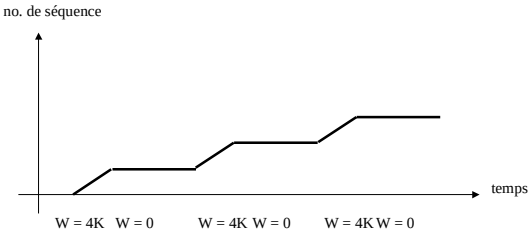
Transfert



- Fenêtre glissante
  - se referme à chaque envoie
  - avance à la réception de ACK

134

Contrôle de flux



- Emetteur rapide, récepteur lent
  - envoie de 4K
  - fenêtre de 0, attente

135

Contrôle de flux

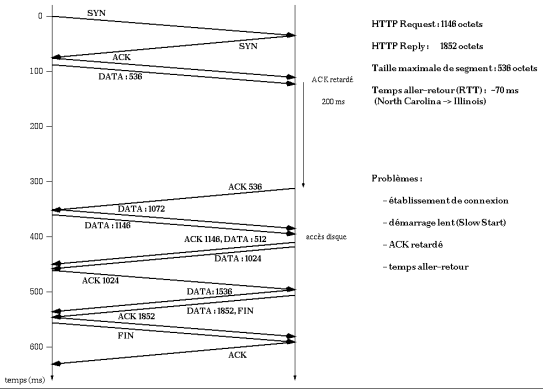
- Taille maximale de la fenêtre d'émission de 64Koctets
  - suffisant pour remplir le tuyaux ?
  - exemple pour RTT de 100ms

débit	*débit-latence*
T1 (1.5Mb/s)	18 Koctets
Ethernet (10Mb/s)	122 Koctets
T3 (45Mb/s)	549 Koctets
FDDI (100Mb/s)	1.2 Moctets

- Option "facteur d'échelle"

136

Exemple de transfert TCP

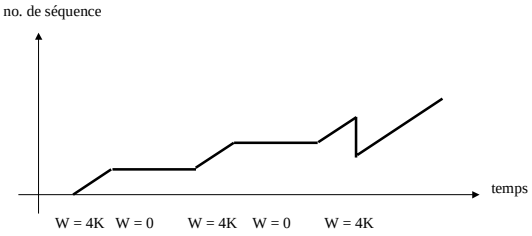


Contrôle des erreurs

- Les segments portent le n° du 1er octet transporté
- À l'envoi, une temporisation est armée
- Le récepteur acquitte les segments reçus
  - soit immédiatement (segment en désordre)
  - soit après un retard
    - » il a un autre segment à envoyer *(piggybacking)*
    - » reçoit un autre segment (ACK cumulatifs)
    - » règle générale : *" ACK every other segment "*
- Sur épuisement de la tempo, la retransmission à partir du 1er segment non acquitté (TCP Reno : remise à zéro des temporisations des autres segments)

138

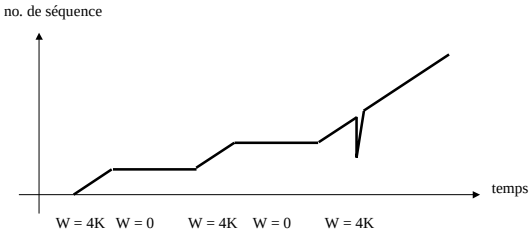
Retransmissions



- Retransmission à partir du premier octet non acquitté

139

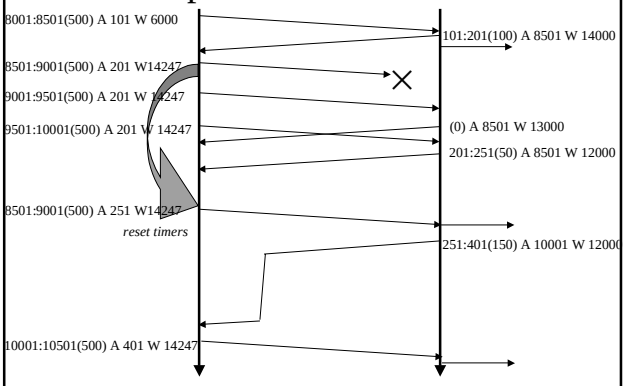
Retransmissions



- Réception des segments en désordre
  - heuristiques *"retransmission rapide"*

140

Exemple de transfert - Reno



141

Exemple de transfert

- Retransmission à partir 8501
  - retransmission continue *(Go-back-N)*, mais seulement du premier segment
  - récepteur stocke les segments en désordre (9001 et 9501)
  - dès la réception de 8501, on peut passer 8501:10001 à l'application
  - après la réception de 10001, la transmission continue

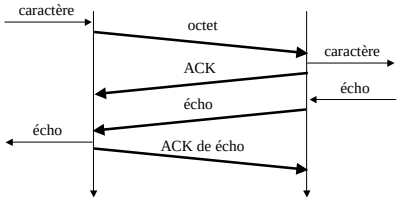
142

Exemple de transfert

- Reno - optimisation
  - éviter la retransmission des segments après une temporisation
  - les temporisations de tous les autres segments sont arrêtées
  - ressemble plus à la retransmission sélective

143

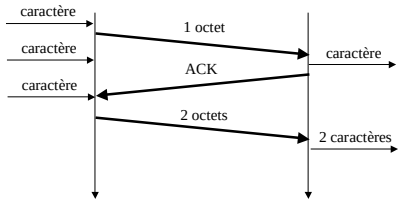
Trafic interactif



- ACK retardé
  - ACK et écho combinés dans le même segment
  - délai 200 ms

144

Algorithme de Nagle



- Émetteur accepte l'envoi d'un seul petit segment (petit = moins que MSS) non acquitté
  - éviter l'envoi de petit segments sur un WAN
  - peut être désactivé par l'application (ex. X)

145

Silly Window syndrome

- Petite taille de la fenêtre annoncée

```
            ←    Ack 0 W 2000
0:1000      →    buf = 2000, freebuf = 1000
1000:2000   →    freebuf = 0
            ←    Ack 2000 W 0
                 appl lit 1 octet : freebuf = 1
            ←    Ack 2000 W 1
2000:2001   →    freebuf = 0
                 appl lit 1 octet : freebuf = 1
            ←    Ack 2001 W 1
2001:2002   →    freebuf = 0
```

146

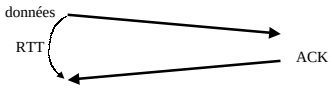
Silly Window syndrome

- Émetteur a beaucoup de données
- Petite fenêtre annoncée l'oblige à envoyer des segments petits
- Solution par récepteur
  - annoncer la fenêtre par tranches larges (MSS ou 1/2 du tampon du récepteur)
- Solution par émetteur
  - retarder l'envoi de petits segments
  - soit PUSH positionné
  - soit on a au moins 1/2 de la max fenêtre à envoyer

147

Traitement d'erreurs

- Temporisation associée à chaque segment non acquitté RTO *(Retransmission TimeOut)*
- Estimer le temps aller-retour RTT - *(Round Trip Time)*
- Réglage des temporisations
  - réglage de la fenêtre optimale
- Mesure du temps de retour d'un ACK
- Moyenne pondérée, filtre passe-bas



148

Estimation du RTT

- Initiale
  - RTO = R × β, β = 2
  - R - estimation lissée du temps aller-retour
    - » R ← qR + (1 - q) RTT, q = 0.9
  - RTT - mesure du temps aller-retour
  - RTO double à chaque retransmission
- Actuelle (Van Jacobson)
  - estimation plus fine, écart type
  - 0.25 et 0.125 - gains du filtre passe bas (puissance de 2)

149

Estimateur du RTT

```
sampleRTT = la derniere mesure de RTT
estimatedRTT = la derniere estimation de la moyenne
deviation = la derniere estimation de la variance

initialisation : estimatedRTT = sampleRTT + 0.5 s;
deviation = estimatedRTT/2

err = sampleRTT - estimatedRTT
estimatedRTT = estimatedRTT + 0.125 × err
deviation = deviation + 0.250 × (|err| - deviation)
RTO = estimatedRTT + 4 × deviation
```

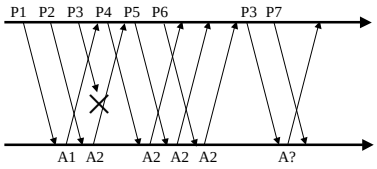
150

Règles de Karn et Partridge

- On ne mesure pas de RTT s'il a des retransmissions
  - on ne sait pas si c'est un segment perdu ou un ACK perdu
- *Timer exponential backoff*
  - on double la valeur de RTO à chaque retransmission
- Si au début il n'y a pas de mesures
  - RTO = 6s
  - après RTO = 12s et on applique la règle de Karn
    - » $2 \times 12s = 24s$

151

Retransmission rapide



- *Fast retransmit*
  - intervalle de retransmission peut être grand
  - ajouter un comportement du type *Retransmission Sélective*
  - si on reçoit 3 ACK dupliqués pour le même segment avant la temporisation, on retransmet

152

153

Partie 5 : Congestion Control in TCP

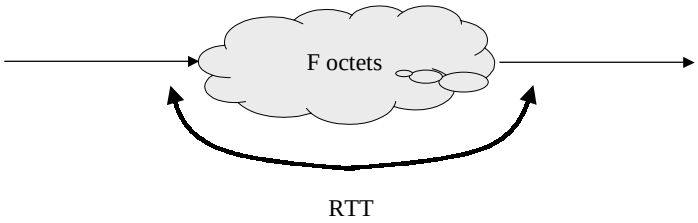
154

Contrôle de congestion

- TCP s’appuie sur IP (sans connexion)
- TCP opère de bout-en-bout (détection de la congestion peu fiable et indirecte)
- Chaque source TCP agit «seule» (pas de coopération entre sources)
- Pertes de paquets interprétées comme un signalement de la congestion

155

Fenêtre d'émission



- F - le nombre d'octets non acquittés
  - débit = F/RTT (formule de Little)
- Si congestion
  - RTT augmente, réduction automatique du débit de la source
  - mécanisme de contrôle : diminuer F

156

Contrôle de congestion

- Fenêtre d'émission - nombre d'octets non-acquittés
  - F = min (`cwnd`, W)
    - `cwnd` - maintenu par la source
    - W - fixé par destination, champs W
- Connexion TCP peut être dans une de trois phases (point de vue du contrôle de congestion)
  - **démarrage** *(slow start)*, après une perte détectée par une tempo
  - **récupération** *(fast recovery)*, après une perte détectée par retransmission rapide *(fast retransmit)*
  - **évitement** *(congestion avoidance)*, tous les autres cas

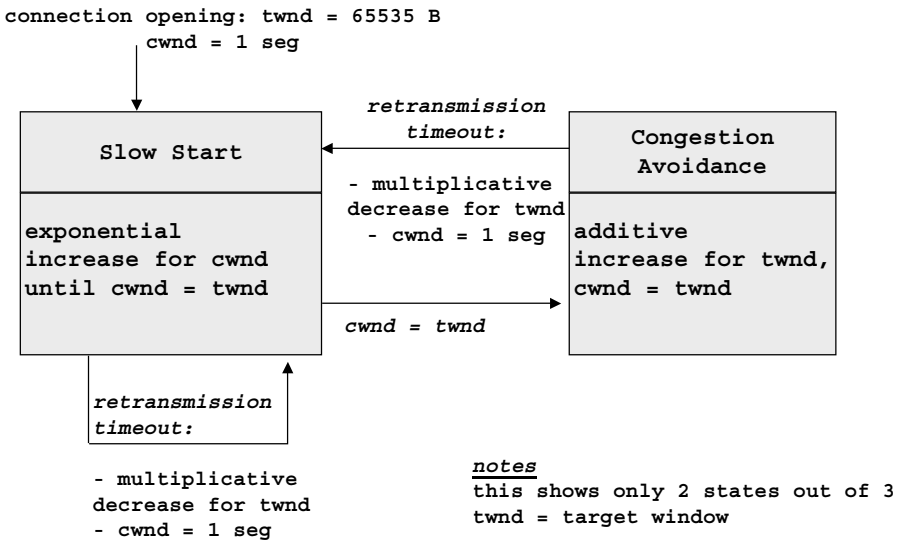
157

TCP and Congestion Control

- TCP is used to avoid congestion in the Internet
  - a TCP source adjusts its window to the congestion status of the Internet (slow start, congestion avoidance)
  - this avoids congestion collapse and ensures some fairness
- TCP sources interprets losses as a negative feedback
  - use to reduce the sending rate
- UDP sources are a problem for the Internet
  - use for long lived sessions (ex: RealAudio) is a threat: congestion collapse
  - UDP sources should imitate TCP : “TCP friendly”

158

Slow Start and Congestion Avoidance



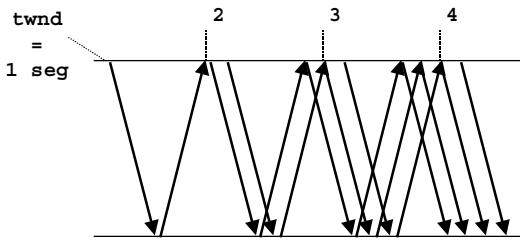
159

Increase/decrease

- Multiplicative decrease
  - **twnd = 0.5 × current_window**
  - **twnd = max (twnd, 2 × MSS)**
- Additive increase
  - for each ACK
    - **twnd = twnd + MSS × MSS / twnd** ***(w←w+1)***
    - **twnd = min (twnd, max-size) (64KB)**
- Exponential increase
  - for each ACK
    - **cwnd = cwnd + MSS**
    - **if (cwnd == twnd)** go to avoidance phase

160

twnd Additive Increase



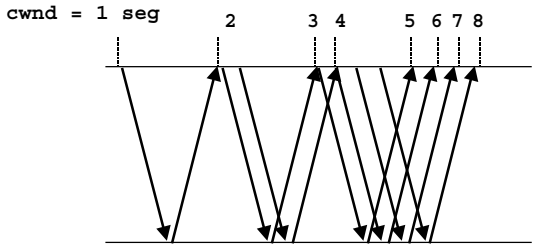
- during one round trip + interval between packets: increase by 1 packet (linear increase)
- (equivalent to **twnd = twnd + 1/twnd** if TCP would have all segments of length **MSS**)

161

Slow Start

- purpose of this phase: avoid burst of data after a retransmission timeout
- / * exponential increase for **cwnd** */

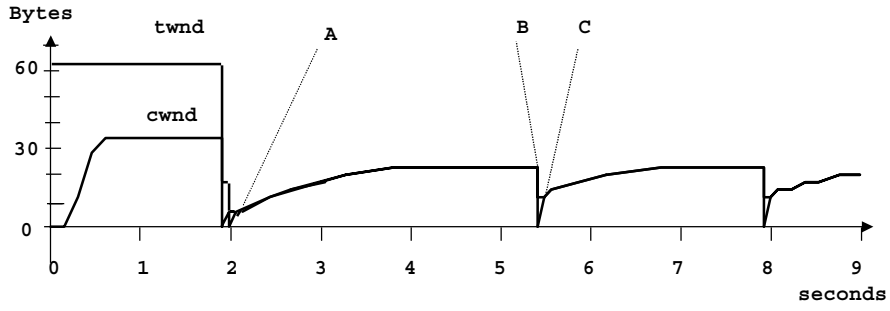
```
non dupl. ack received during slow start ->
            cwnd = cwnd + seg (in bytes)
            if cwnd = twnd then transition to congestion avoidance
```



- window increases rapidly up to the stored value of **twnd** (this stored value is called **ssthresh** in the literature)

162

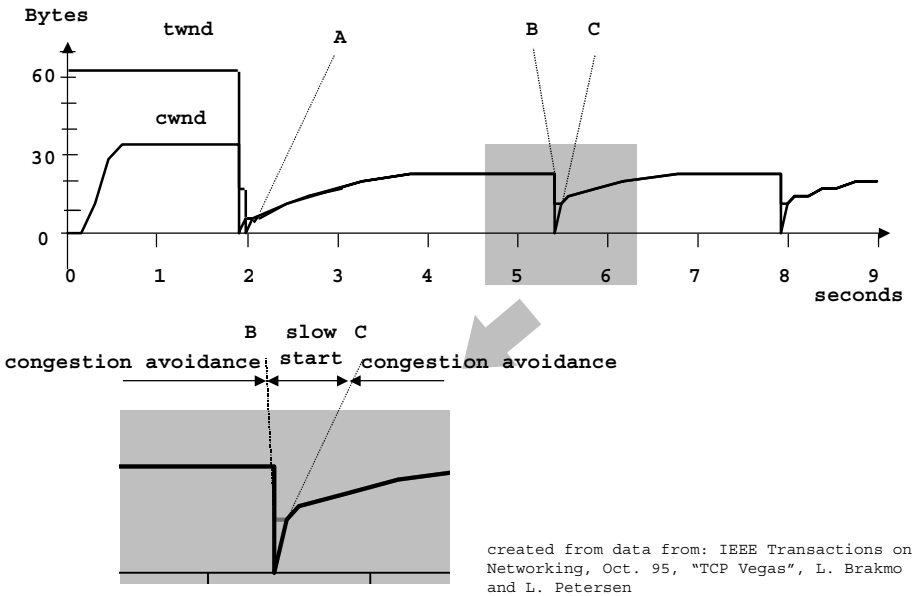
Slow Start and Congestion Avoidance Example



created from data from: IEEE Transactions on Networking, Oct. 95, "TCP Vegas", L. Brakmo and L. Petersen

163

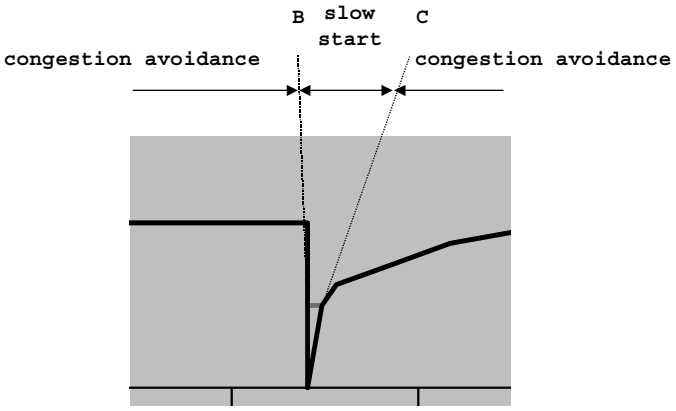
Slow Start and Congestion Avoidance Example



created from data from: IEEE Transactions on Networking, Oct. 95, "TCP Vegas", L. Brakmo and L. Petersen

164

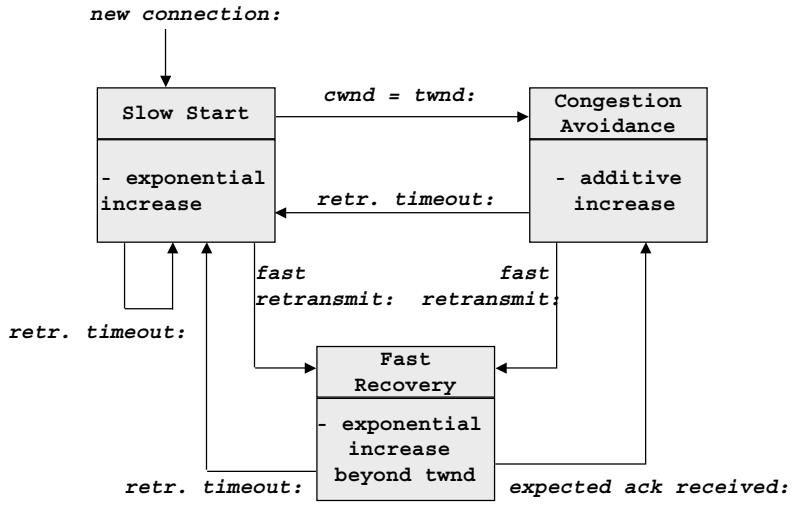
Slow Start and Congestion Avoidance Example



created from data from: IEEE Transactions on Networking, Oct. 95, "TCP Vegas", L. Brakmo and L. Petersen

165

Congestion States



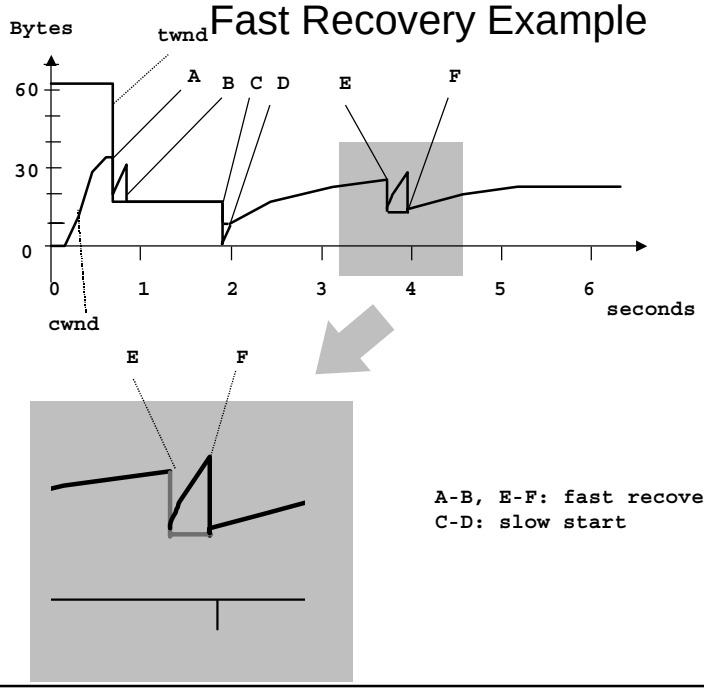
166

Fast Recovery

- Multiplicative decrease
  - `twnd = 0.5 × current-size`
  - `twnd = max (twnd, 2 × MSS)`
- Fast Recovery
  - `cwnd = twnd + 3 × MSS (exp. increase)`
  - `cwnd = min (cwnd, 64K)`
  - `retransmission of the missing segment (n)`
- For each duplicated ACK
  - `cwnd = cwnd + MSS (exp. increase)`
  - `cwnd = min (cwnd, 64K)`
  - `send following segments`

167

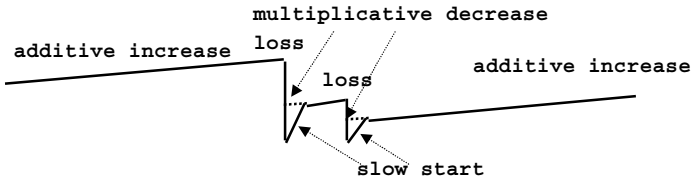
Fast Recovery Example



168

Congestion Control: Summary

- Congestion control aims at avoiding congestion collapse in the network
- With TCP/IP, performed in end-systems, mainly with TCP
  TCP Congestion control summary
- Principle: sender increases its sending window until losses occur, then decrease



- target window: additive increase (no loss), multiplicative decrease (loss)
- 3 phases:
  slow start: starts with 1, exponential increase up to `twnd`
  congestion avoidance: additive increase until loss or no increase
  fast recovery: fast retransmission of one segment
- slow start entered at setup or after retransmission timeout
- fast recovery entered at fast retransmit

Partie 6
Contrôle de trafic dans ATM

Introduction à ATM
Catégories de services
Contrôle de trafic

169

ATM *(Asynchronous Transfer Mode)*

- Couche réseau orientée connexion
  - circuits virtuels
    - » commutation de circuits - performance
    - » phase d'établissement - négociation du contrat de service
  - paquets courts de longueur fixe
    - » cellule *(cell)*
  - acheminement non garanti - pertes possibles
  - ordre garanti
  - s'appuie sur un support physique optique
    - » faible taux d'erreurs

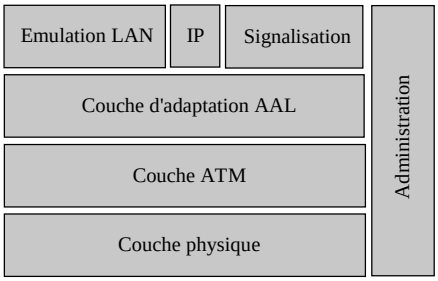
170

ATM

- Au-dessus d'un support large bande
  - couche physique sur fibre optique
    - » hiérarchies synchrones SDH et SONET
  - débits
    - » OC-3, STM-1 : 155 Mbit/s
    - » OC-12, STM-4 : 622 Mbit/s
    - » OC-48 : 2.4 Gbit/s
  - multiplexage statistique
    - » meilleure utilisation du débit

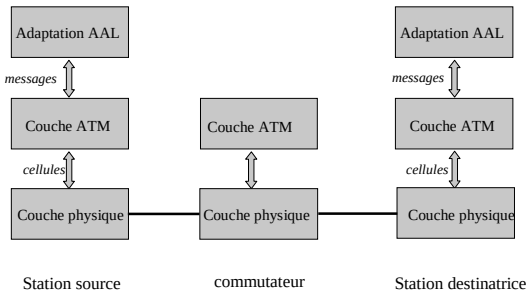
171

Modèle de référence



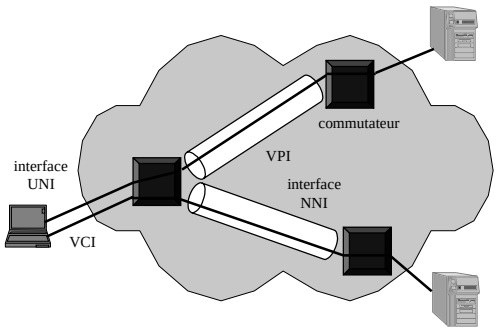
172

Couches ATM



173

Commutation ATM



174

Interfaces

- UNI *(User -Network Interface)*
  - hôte au réseau
- NNI *(Network -Network Interface)*
  - commutateur au commutateur

175

Cellules ATM

- GFC *(Generic Flow Control)*
  - pour transmettre de l'information sur le contrôle de flux (seulement sur l'interface UNI)
- VPI *(Virtual Path Id)*
  - no. du chemin virtuel
- VCI *(Virtual Circuit Id)*
  - no. du circuit virtuel

176

Cellules ATM

- Type
  - cellule de données ou de gestion (ex. RM)
  - indication de congestion
  - bit SDU *(Service Data Unit)* - marque de fin
- CLP *(Cell Loss Priority)*
  - 1 : la cellule peut être détruite s'il faut
- HEC *(Header Error Check)*
  - code correcteur d'une erreur sur l'entête

177

Cellules ATM

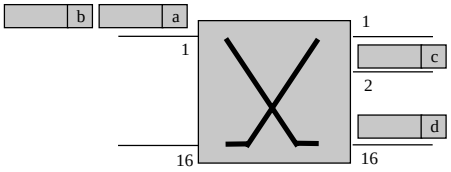
- Sur l'interface UNI

- Sur l'interface NNI

178

Commutation VPI/VCI *(Label Swapping)*

entrée	VPI/VCI	sortie	VIP/VCI
1	a	2	c
1	b	16	d



179

Commutation

- VPI/VCI
  - commutateur
- VPI
  - brasseur
  - circuits virtuels commutés à l'intérieur des VP

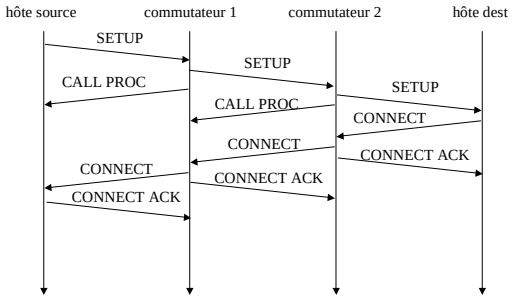
180

Signalisation

- Comment établir et fermer des circuits virtuels ?
  - un circuit spécial toujours ouvert (VPI=0, VCI=5)
  - protocole fiable SSCOP au dessus de AAL5
- Messages
  - SETUP
  - CALL PROCEEDING
  - CONNECT
  - CONNECT ACK
  - RELEASE
  - RELEASE COMPLETE

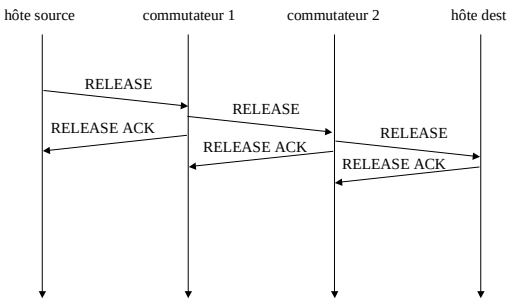
181

Établissement de connexion



182

Fermeture de connexion



183

Adressage

- Trois formats
  - E.164 : numéro de téléphone - 15 chiffres décimaux
  - adresse OSI - 17 octets
    - » pays - 2 octets
    - » autorité - 3 octets
    - » domaine - 2 octets
    - » zone - 2 octets
    - » adresse locale - 6 octets
    - » autres infos

184

Services de la couche ATM

- Circuit virtuel
  - ordre garanti
- Pas de garanties de fiabilité
  - pertes de cellules possibles
- Qualité de service d'un circuit virtuel
  - si un CV est alloué, le réseau garantit une qualité de service selon un contrat
- Description du trafic à l'entrée

185

Catégories de service

- CBR *(Constant Bit Rate)*
  - exemple : le canal T1
- VBR-RT *(Variable Bit Rate, Real Time)*
  - exemple : une vidéoconférence
- VBR-NRT *(Variable Bit Rate, Non-Real Time)*
  - exemple : e-mail multimédia
- ABR *(Available Bit Rate)*
  - exemple : Web surfing
- UBR *(Unspecified Bit Rate)*
  - exemple : FTP en tâche de fond

186

Service CBR *(Constant Bit Rate)*

- Débit garanti
  - émule un circuit physique (audio, vidéo)
- Contrat entre le réseau et la station
- Contrôle de conformité du trafic
  - ne pas dépasser l'allocation
- Paramètres contrôlés
  - débit crête - PCR *(Peak Cell Rate)*
  - irrégularité - CDVT *(Cell Delay Variation Tolerance)*
  - cellules non conformes marquées ou rejetées

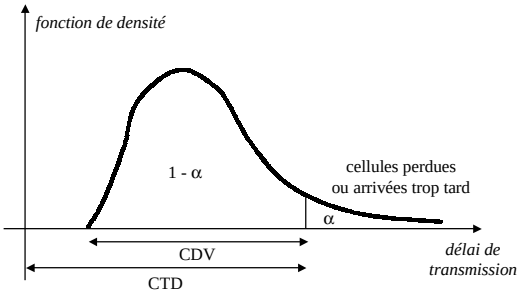
187

Attributs du CBR

- Taux de perte (ex. 10^{-10}) - CLR *(Cell Loss Rate)*
- Délai (ex. 100 ms) - CTD *(Cell Transfer Delay)*
- Gigue (ex. 10 ms) - CDV *(Cell Delay Variation)*
- Débit crête (ex. 170.2 cell/s pour 64 Kb/s) - PCR *(Peak Cell Rate)*
- Irrégularité (ex. 2 temps inter-cellule) - CDVT *(Cell Delay Variation Tolerance)*
  - correspond à un sceau troué *(Leaky Bucket)*

188

Attributs du CBR



189

Contrat de trafic

- Connexion CBR définie par PCR, CDVT
- Connexion doit se conformer à GCRA *(Generic Cell Rate Algorithm)*
- GCRA *(T, τ)*
  - *T* - délai inter-cellule idéal
    - » débit physique du lien / PCR [b/s]
  - τ - tolérance (CDVT)
  - exemple : PCR = 35 000 cell/s (15 Mb/s), débit physique = 155 Mb/s, CDVT = 2
    - » GCRA (10, 2)

190

GCRA

- Algorithme qui décide de la conformité
  - cellule arrive à *t*
  - `X` - niveau, `LCT` - Last Conformance Time

```
GCRA (t)
  static X, LCT = 0
  if (X - t + LCT > τ)
    NON-CONFORME
  else
    X = max (X - t + LCT, 0) + T; LCT = t
    CONFORME
```

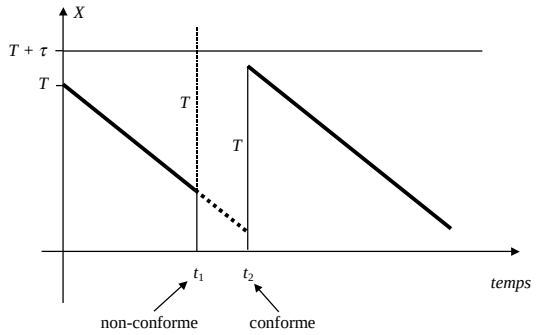
191

Comparaison avec le seau troué

- Sceau de capacité $T + \tau$
- Taux de sortie μ = 1/sec
- Arrivée d'une cellule
  - il reste $X - t$ du liquide
  - si le niveau plus haut que τ, cellule non-conforme
  - si le niveau plus bas que τ, cellule conforme, et le niveau augmente de *T*

192

GCRA



193

Exemple GCRA

- GCRA (10, 2), cellules non-conformes ?
  - 0, 10, 18, 28, 38
  - 0, 10, 15, 25, 35
  - 0, 10, 18, 26, 36
  - 0, 10, 11, 18, 28

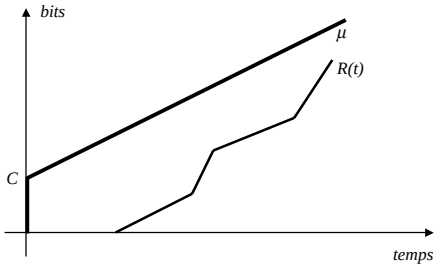
194

Comparaison avec le seau troué

- Multiplication par une constante
- Capacité $(T + \tau)\ l / T$, l - taille de cellule
- Taux de sortie $\mu = l / T$
- GCRA (T, τ) est équivalent à un seau troué
  - taux de sortie $\mu = l / T$ [bits]
  - capacité $l + \tau \mu$ [b/s]
- GCRA $(T, CDVT)$
  - PCR tel que $\mu = l / T$
  - capacité $C = l + CDVT\ \mu$

195

Flot contraint



- $R(t)$ - nb. de bits à l'entrée
  - $R(t + a) - R(a) \leq \mu\ t + C$

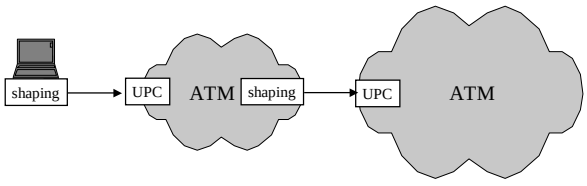
196

Allocation des circuits CBR

- Débits
  - somme des $\mu \leq$ débit physique
- Tampons
  - chaque circuit $l + CDVT \times \mu$

197

Polissage de sources



- Polissage limite les pertes
- Opération normale
  - pas de pertes à UPC

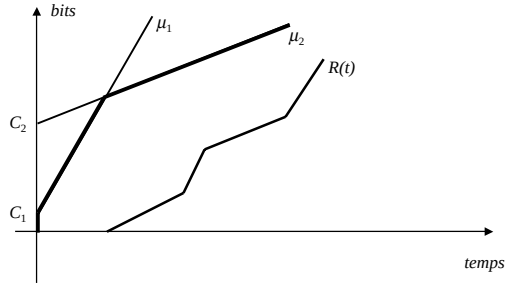
198

Service VBR *(Variable Bit Rate)*

- VBR-RT - le même que CBR, mais en plus
  - – débit moyen (ex. SCR = 3500 cell/s, PCR = 35000 cell/s) - SCR *(Sustainable Cell Rate)*
  - – tolérance de rafale (ex. BT = 500 temps inter-cellule) - BT *(Burst Tolerance)*
- Connexion doit se conformer à deux GCRA
  - – GCRA (débit physique/PCR, CDVT)
  - – GCRA (débit physique/SCR, BT + CDVT)
- VBR-NRT - le même que VBR-RT, mais
  - – pas de définition de CDV

199

VBR

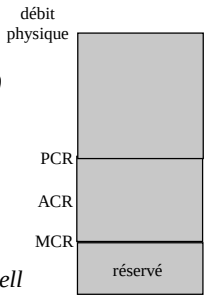


- Flot contraint par deux seaux troués

200

Service ABR *(Available Bit Rate)*

- Débit minimal garanti - MCR *(Minimum Cell Rate)*
- Débit crête - PCR *(Peak Cell Rate)*
- Débit permis - ACR *(Allowed Cell Rate)*
- Autres
  - – taux de perte - CLR *(Cell Loss Rate)*
  - – irrégularité de PCR - CDVT *(Cell Delay Variation Tolerance)*



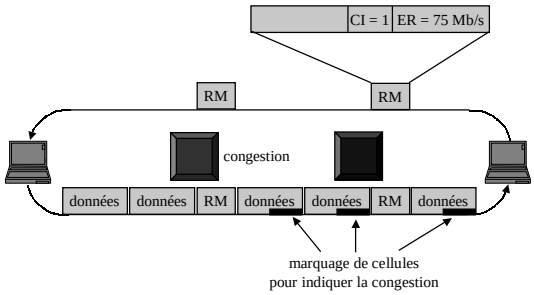
201

Service ABR

- Débit ajusté dynamiquement
- Réseau envoie une indication de congestion ou une notification de débit
  - – source réduit le débit en cas de congestion (diminution multiplicative)
  - – source augmente le débit, si pas de congestion (augmentation additive)
  - – source ne dépasse pas le débit notifié
  - – UPC dans le réseau

202

Indication de congestion



203

Contrôle de trafic

- Congestion détectée dans le réseau
  - – cellules marquées (bit 2 du champ Type = 1)
  - – commutateur ou destination
    - » positionne des champs dans les cellules de type RM *(Resource Management)* (champ Type = 110)
- Cellules RM
  - – CI *(Congestion Indication)*
    - » diminution multiplicative
  - – NI *(No Increase)*
  - – ER *(Explicit Rate)*
    - » ACR = min (ACR, ER)

204

Service UBR *(Unspecified Bit Rate)*

- Service au mieux *(best effort)*
- Perte d'une cellule provoque la perte d'un paquet de la couche adaptation (p. ex. AAL5)
  - EPD *(Early Packet Discard)*
    - » si un seuil dépassé, toutes les cellules d'un paquet rejetées

205

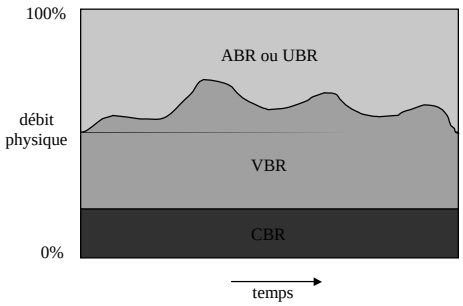
Catégories de service

Caractéristiques	CBR	VBR-RT	VBR-NRT	ABR	UBR
débit garanti	Oui	Oui	Oui	Min	No
pour le trafic RT	Oui	Oui	No	No	No
pour le trafic *bursty*	No	No	Oui	Oui	Oui
retour sur congestion	No	No	No	Oui	No

- RT - temps réél

206

Catégories de service



207

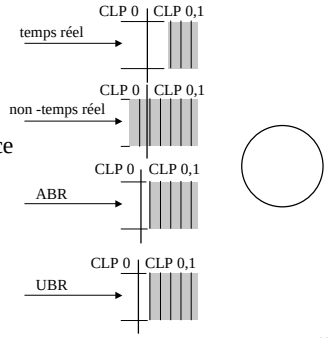
Éléments du contrôle de trafic

- Circuit virtuel
  - contrôle d'admission (CAC - *Connection Admission Control*)
- Cellule
  - conformité - UPC
  - ordonnancement et gestion des files
  - marquage de cellules
  - priorités de cellules

208

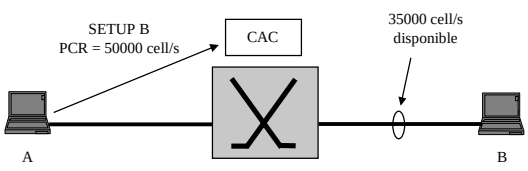
Ordonnancement et gestion des files

- Files de priorité
- Priorité
  - selon VPI/VCI
  - contrat de service à l'ouverture
- Rejet
  - bit CLP



209

Contrôle d'admission



- Décider si on accepte une connexion

210