Majeure Informatique - X04

Contrôle écrit : Réseaux, protocoles (INF 586)

Jeudi 29 mars 2007

Le contrôle dure 2h. Les documents sont autorisés. Notation sur 22.

Question 1 : Gestion de Buffer (2 points)

Considérons le tampon de sortie d'un nœud lié à une liaison ayant une capacité de C bits par seconde (bps). Le tampon peut contenir X bits. Soit $R(t)$ le nombre de total de bits générés par l'application dans l'intervalle de temps $[0, t]$ et envoyés vers le tampon pour transmission.

a) Calculer la taille minimale du tampon X_{min} en fonction des autres paramtères du système pour éviter tout débordement du tampon.

b) Calculer la capacité minimale C_{min} en fonction des autres paramètres de façon à ce que la quantité des données en attente dans le buffer ne dépasse pas X.

Question 2 : Efficacité des mécanismes de retransmission (5 points)

Considérons une liaison de capacité C bps entre deux stations A et B. Nous nous proposons de calculer l'efficacité (débit maximum normalisé) des protocoles de contrôle d'erreur Stop and Wait (SW), Selective Repeat (SR) et Go-Back-N (GBN). Soient L (bits) la longueur de la trame de données et d le délai de propagation entre A et B. Nous négligeons le temps de traitement dans les stations A et B ainsi que la taille des trames d'accusés de réception. On définit le paramètre $a = d/T$ (T étant le temps de transmission d'une trame sur la liaison). Soit p la probabilité qu'une trame de donnée soit perdue (les pertes sont considérées indépendentes et uniformément réparties). Les trames d'accusés de réception ne sont jamais perdues.

a) Calculer l'efficacité E_{SW} du protocole Stop and Wait en fonction de a en considérant un canal sans erreur (p=0).

b) Considérons le cas d'un lien avec erreurs ($p \neq 0$). Le protocole Stop and Wait est utilisé. La temporisation de retransmission est égale à $2d$ (le temporisateur est armé juste à la fin de la transmission de la trame). Soit N_{tx} le nombre moyen de transmission d'une trame. Calculer N_{tx} en fonction de p. Calculer l'expression de E_{SW} dans le cas d'un lien avec erreurs. (On rappelle que $\sum_{i=1}^{\infty}(i.p^{(i-1)}) = 1/(1-p)^2$ pour $-1 < p < 1$.)

c) Considérons un protocole de contrôle d'erreur utilisant une fenêtre coulissante de taille W paquets. Calculer l'efficacité d'un tel protocole dans le cas d'un lien sans perte en fonction de W et de a.

d) Considérons le cas d'un lien avec erreurs ($p \neq 0$). Le protocole utilisé est Selective Repeat (seul le paquet considéré perdu est retransmis). Calculer N_{tx} en fonction de p dans ce cas. En déduire l'expression de l'efficacité E_{SR} sur un lien avec erreurs.

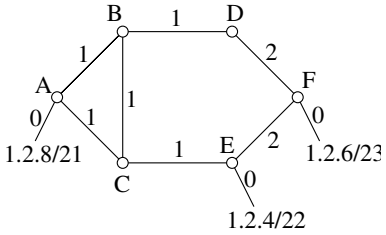
e) Considérons le cas d'un lien avec erreurs ($p \neq 0$). Le protocole utilisé est le Go-Back-N. Notons que dans ce cas, chaque erreur provoque la retransmission de K trames et non pas d'une seule trame (comme c'est la cas avec le Selective Repeat).

e1) Calculer K en fonction de a et de W.

e2) Calculer en fonction de p et de W le nombre moyen de trames transmises afin de transmettre une trame avec succès. On rappelle que $\sum_{i=1}^{\infty}(p^{(i-1)}) = 1/(1-p)$ pour $-1 < p < 1$.

e3) En déduire l'expression de l'efficacité E_{GBN} sur un lien avec erreurs.

Question 3: Routage par vecteur de distances (4 points)



On conisdère le réseau de la figure ci-dessus qui utilise un protocole de routage par vecteur de distances. Tous les liens inter-routeurs ont longueur 1 sauf DF et EF qui ont longueur 2. A annonce systématiquement une distance de 0 au sous-réseau IP 1.2.8/21 (indiquant ainsi que les adresses IP de même préfixe de 21 bits que les trois octets 1.2.8 sont accessibles par A). De même E et F annoncent systématiquement une distance de 0 à 1.2.4/22 et 1.2.6/23 respectivement (une adresse IP sera toujours trouvée dans le sous-réseau de plus long préfixe commun).

En cas de choix possible entre plusieurs chemins, on supposera qu'un routeur choisira toujours de faire passer à son premier voisin, dans l'ordre alphabétique des noms, offrant un plus court chemin (préférant B à C par exemple).

a. Quelles seront les routes suivies par des paquets IP de :

1. 1.2.8.1 vers 1.2.6.1,
2. 1.2.4.2 vers 1.2.12.2,
3. 1.2.15.15 vers 1.2.5.15,
4. 1.2.7.7 vers 1.2.8.8 ?

b. Quelle est la distance maximale qu'on pourrait observer dans le réseau (avec défaillances possibles) ? Quelle valeur préconisez vous pour l'infini ?

On suppose que les routeurs transmettent leurs vecteurs de distances avec la même période T dans l'ordre cyclique A, B, C, D, E, F : A transmet son vecteur de distances à ses voisins, puis B, puis C,... On suppose de plus que la table de routage n'est re-calculée qu'au moment d'annoncer le vecteur distance.

c. Tracer dans un tableau l'évolution des distances de chaque routeur vers 1.2.4/22 si le lien CE casse. Au bout de combien de périodes le réseau sera-t-il stabilisé ? Sur quels liens un trafic de 1.2.8.1 vers 1.2.4.1 avec TTL de 255 aurait-il le plus d'impact ?

d. Indiquer à nouveau dans un tableau l'évolution des distances vers 1.2.4/22 si le protocole utilise le mécanisme de "split horizon" (qui consiste à cacher certaines entrées de son vecteur de distances à chaque voisin) ? Quel avantage procure cette technique ici ?

Question 4 : Contrôle de congestion TCP (4 points)

La version TCP Tahoe utilise trois algorithmes pour assurer le contrôle de congestion : Slow-Start, Congestion Avoidance et Fast Retransmit.

a) Expliquer en détail le fonctionnement de deux premiers algorithmes et en particulier le rôle et l'évolution des variables cwnd et ssthresh.

b) Avec le Fast Retransmit, TCP infère qu'un paquet est perdu suite à la réception d'un "petit" nombre d'ACKs en double (duplicate ACKs) et retransmet ce paquet sans attendre l'expiration du temporisateur de retransmission (l'émetteur réduit cwnd de moitié). Afin d'éviter que le "tuyau se vide" après un Fast Retransmit (ce qui pourrait provoquer un passage en mode Slow Start) suite à la

perte d'un seul paquet, la fenêtre est incrémentée de 1 pour chaque ACK en double reçu. TCP sort du mode Fast Recovery quand l'émetteur reçoit un ACK acquittant au moins le paquet soupçonné perdu. Expliquer pourquoi cette modification (dite Fast Recovery et implémentée dans TCP Reno) permet d'améliorer les performances de TCP dans le cas d'erreurs aléatoires tout en respectant dans le fond la philosophie du contrôle de congestion TCP.

c) Quel serait le comportement de TCP Reno en présence de multiples pertes de paquets dans une même fenêtre de transmission ? (Dans le cas de réseaux ayant un produit bande passante délai élevé et un taux de corruption de paquet non nul, il se peut qu'il y ait en fait plus qu'un paquet perdu dans une même fenêtre). Proposer une modification du Fast Recovery permettant d'améliorer les performances de TCP en cas de pertes multiples. Tuyau: Que faudrait il faire pour rester en mode Fast Recovery jusqu'à ce que *tous* les paquets transmis soit acquittés ?

Question 5 : Réseaux avec borne de délai (4 points)

Nous avons vu dans le cours qu'il est possible de calculer une borne maximale des délais dans les réseaux par paquets.

a) Quelles composantes sont nécessaires et suffisantes pour construire un réseau avec une borne sur les délais ? Décrire brièvement chacune de ces composantes.

Le délai de transmission et d'attente (transmission and queueing delay) $D(i)$ pour les paquets d'une connexion i est donnée par la formule :

$$D(i) \leq \sigma(i)/g(i) + \sum_{k=1}^{k=K-1} Pmax(i)/g(i,k) + \sum_{k=1}^{k=K} Pmax/r(k)$$

où :
$\sigma(i)$ est la taille maximum de burst pouvant être généré par la connexion i,
$g(i,k)$ est la bande passante allouée à la connexion i, au niveau du lien de sortie du $k^{ième}$ ordonnanceur,
K est le nombre de routeurs-ordonnanceurs traversés par la connexion i,
$g(i)$ est le minimum de tous les $g(i,k)$,
$r(k)$ est la capacité des liens de sortie du $k^{ième}$ routeur-ordonnanceur,
$Pmax(i)$ est la taille maximum des paquets de la connexion i,
$Pmax$ est la taille maximum des paquets dans le réseau.

b) Expliquer à partir de cette formule de quoi dépend la borne du délai. Considérer en particulier le cas où $Pmax$ tend vers zéro. Considérer ensuite l'impact d'une taille de paquet non nulle sur les délais.

c) Soit un réseau avec une source, une destination et trois routeurs intermédiaires. Toutes les liaisons ont une capacité de 155 Mbps. Considérons une connexion lissée avec un "leaky bucket" dont les paramètres sont (σ = 4096 octets, ρ = 100 Kbps) (ceci signifie que la source peut émettre $\sigma + \rho t$ bits dans n'importe quel intervalle de temps de durée t). La taille de tous les paquets est de 1 Koctet. Le délai de propagation est de 30 ms. Quelle est la valeur de la bande passante qui devrait être allouée à cette connexion dans le réseau (au niveau des liens de sortie des routeurs-ordonnanceurs, toutes les $g(i,k)$ étant égales à $g(i)$) afin de garantir une borne de délai de bout en bout de 150 ms. En déduire le facteur de sur-réservation de la connexion i et le poids de cette connexion au niveau des ordonnanceurs WFQ. Que deviennent les résultats (bande passante à allouer, facteur de sur-réservation et poids de la connexion) pour une taille de paquets de 100 octets ? de 10.000 octets ? Discussion.

Question 6 : Multiplexage statistique (3 points)

Considérons un multiplexeur avec quatre liens (A, B, C et D) en entrée et un lien (E) en sortie. Le trafic arrivant sur le lien A occupe le lien à 50%, le trafic arrivant sur le lien B occupe le lien à 25%, le traffic arrivant sur le lien C occupe le lien à 12.5% et le trafic arrivant sur le lien D occupe le lien à 12.5 %. Les paquets (ou plutôt les cellules) ont tous la même taille. Le temps de transmission d'une cellule est utilisé comme unité de temps.

a) Calculer le débit normalisé minimal et maximal sur le lien de sortie pour un fonctionnement régulier du multiplexeur quelque soit le profil exact des flux en entrée.

b) Considérons que nous avons un multiplexage synchrone et que le trafic sur A consiste en la transmission de deux cellules (à partir de l'instant 0) suivies par deux intervalles vides (i.e. des transmissions dans les intervalles de temps 4n+1 et 4n+2 pour $n \geq 0$), que le trafic sur B consiste en une transmission d'une cellule dans les intervalles 4n+3 ($n \geq 0$), que le trafic sur C consiste en une transmission d'une cellule dans les intervalles 8n+3 ($n \geq 0$) et que le trafic sur D consiste en une transmission d'une cellule dans les intervalles 8n+5 ($n \geq 0$). La taille de la trame de sortie est de 4 cellules. Indiquer le contenu des douze premières trames en sortie. Discuter l'efficacité du multiplexage synchrone.

c) On considère maintenant que nous avons un multiplexage asynchrone. Les flux d'entrée arrivent avec le profil décrit en b), mais il est possible de réorganiser les cellules. Comment devrait on "organiser" les données pour minimiser la capacité du lien de sortie ? Tracer le contenu des huit premières trames avec cette nouvelle organisation des données.

Question 1: Gestion de Buffer

$R(t) - R(s)$ est la quantité de données reçue par l'application pour transmission pendant un intervalle de temps [s,t] dont $C(t-s)$ seront transmis sur le lien et $Q = R(t) - R(s) - C(t-s)$ stockées dans le tampon. La taille minimale X_{min} de tampon requis au nœud pour éviter tout débordement devrait être supérieure ou égale à $\sup_{s \leq t}(Q)$.

Si l'on veut qu'il n'y ait pas du tout de mise en tampon des paquets, la capacité du lien de sortie devrait être égale au moins à $\sup_{s \leq t}((R(t)-R(s))/(t-s))$. Si par contre nous nous autorisons une mise en attente de X bits alors la capacité de sortie C_{min} devrait être au moins égale à $\sup_{s \leq t} \frac{((R(t)-R(s)-X)}{(t-s)}$ de façon à ce qu'on puisse "évacuer" tous les $(R(t) - R(s) - X)$ bits pendant un temps $(t-s)$.

Question 2: Efficacité des mécanismes de retransmission

a) Un long message sera envoyé en tant qu'une succession de trames T1, T2,..., Tn de la façon suivante : A envoie T1, B renvoie un ACK, A envoie T2, B renvoie un ACK,..., A envoie Tn, B renvoie un ACK. Le temps total pour envoyer le message sera donc de n.D, D étant le temps pour envoyer une trame et recevoir l'accusé de réception.

D = T + 2d (en négligeant le temps de traitement et la taille des ACKs). Comme on peut émettre au mieux (i.e. en l'absence d'erreurs) une trame toutes les D secondes, le débit maximal sera donc de 1/D trames par seconde. Le débit maximum normalisé sera de T/D ((1/D)/(1/T), la capacité de la liaison en trames par seconde étant de 1/T = C/L).

Le débit normalisé du Stop and Wait sans erreurs sera donc $E_{SW} = 1/(1+2a)$.

b) Si la trame est transmise deux fois, le temps requis est $D = 2(T+2d)$. Si la trame est transmise x fois alors le temps requis est de $x(T+2d)$. Le débit maximal normalisé est donc $E_{SW} = \frac{1}{N_{tx}(1+2a)}$. Il suffit donc de calculer N_{tx}. $N_{tx} = \text{E[transmission]} = \sum_{i=1}^{\infty}(i.\Pr[i \text{ transmissions}]) = \sum_{i=1}^{\infty}(i.p^{(i-1)}.(1-p)) = 1/(1-p)$. Donc $E_{SW} = (1-p)/(1+2a)$.

c) Si $W \geq 2a+1$, nous sommes en mode transmission continue, l'efficacité est de 1. Si $W < 2a+1$, nous pourrons émettre au maximum W pendant $1+2a$ et l'efficacité sera de $W/(2a+1)$.

d) Comme dans ce cas nous retransmettons uniquement UNE trame (celle pour laquelle nous avons reçu l'information explicite de demande de retransmission, le nombre moyen de paquets transmis N_{tx} est donné par la même formule que b) ci-dessus et sera donc égale à $1/(1-p)$. L'efficacité du Selective Repeat sera donc $E_{SR} = (1-p)$ pour $W \geq 2a+1$ et $W(1-p)/(2a+1)$ pour $W < 2a+1$.

e1) $K = 2a+1$ pour $W \geq (2a+1)$ et $K = W$ pour $W < (2a+1)$.

e2) $N_{tx} = \sum_{i=1}^{\infty}(f(i)p^{(i-1)}(1-p))$. $f(i)$ étant le nombre total de trames à transmettre si la trame d'origine devrait être émise i fois. $f(i) = 1 + (i-1)K$. Donc $N_{tx} = (1-K)\sum_{i=1}^{\infty} p^{(i-1)}.(1-p) + K\sum_{i=1}^{\infty} ip^{(i-1)}.(1-p) = 1 - K + K/(1-p) = \frac{(1-p+Kp)}{(1-p)}$.

L'efficacité E_{GBN} est donc égale à $\frac{(1-p)}{(1+2a)}$ si $W \geq (2a+1)$ et $\frac{W(1-p)}{(2a+1)(1-p+Wp)}$ si $W < 2a+1$.

Question 3 : Routage par vecteur de distances

a.

1. A, B, D, F,
2. E, C, A,
3. A, C, E,

4. F, D, B, A.

b. 7 de A à B si AB et BC tombent en panne. $\infty = 8$.

c. Quand C détecte la cassure de lien, il va décider de passer par A et va annoncer une distance de 3.

A	B	C	D	E	F
2	2	1	3	0	2
2	2	3	3	0	2
3	4	4	4	0	2
5	5	6	4	0	2
6	5	6	4	0	2

Ce n'est que trois périodes après avoir annoncé une nouvelle distance pour 1.2.4/22 que C va obtenir la route par B avec distance 6.

Avec du trafic de 1.2.8.1 vers 1.2.4.1, on a d'abord une boucle entre A et C (quand C annonce une distance de 3), puis entre A, B et C (quand A annonce une distance de 3, puis entre A et B (quand B annonce une distance de 4 et jusqu'à ce que B annonce 5). On a donc un risque de stress momentané sur les liens AC, AB, BC et CA (surtout si un grand TTL est utilisé).

d. Quand C détecte la cassure de lien, il n'aura plus de route vers 1.2.4/22 puisque A et B l'utilisent comme next hop et ne lui annoncent pas de route vers 1.2.4/22 selon le mécanisme de split horizon .

A	B	C	D	E	F
2	2	1	3	0	2
2	2	∞	3	0	2
3	∞	4	4	0	2
∞	5	∞	4	0	2
6	5	6	4	0	2

La convergence n'est pas plus rapide, mais on n'a plus de boucles sur la route de 1.2.8.1 vers 1.2.4.1.

Question 4: Contrôle de congestion TCP

TCP Tahoe utilise trois algorithmes pour assurer le contrôle de congestion : Slow-Start, Congestion Avoidance et Fast Retransmit. Le fonctionnement de ces algorithmes est décrit ci-dessous :

Slow-Start

- On ajoute une fenêtre de congestion de taille cwnd à l'information conservée pour chaque connexion.
- Lorsqu'on dèmarre ou qu'on redèmarre après une perte, on met cwnd à un paquet.
- Pour chaque ACK d'un nouveau paquet, càd un ACK avec une valeur plus grande que la plus grande valeur déjà reçue, on augmente cwnd d'un paquet.
- Lors de l'émission, on envoie un nombre de paquets n'excédant pas le minimum des fenêtres de transmission (receiver's advertised window) et de congestion (cwnd).

Congestion Avoidance

- Pour chaque ACK d'un nouveau paquet, on augmente cwnd par 1/cwnd (croissance additive).
- Lors de l'émission, on envoie un nombre de paquets n'excédant pas le minimum des fenêtres de transmission et de congestion (cwnd).

Les algorithmes de Slow-Start et de Congestion Avoidance sont indépendants et ont des objectifs différents. En pratique ces deux algorithmes sont utilisés conjointement de la façon suivante (où cwnd est la taille de la fenêtre de congestion et ssthresh est une valeur seuil permettant de passer de l'algorithme de Slow-Start à celui de Congestion Avoidance lorsqu'elle est atteinte):

- Sur un timeout, la moitié de la fenêtre courante est stockée dans ssthresh et, ensuite, cwnd est mis à un paquet.
- Pour chaque ACK d'un nouveau paquet, l'émetteur exécute le code suivant:

  ```
  if (cwnd < ssthresh) cwnd += 1; else cwnd += 1/cwnd;
  ```

- Lors de l'émission, on envoie un nombre de paquets n'excédant pas le minimum des fenêtres de transmission et de congestion (cwnd).

Fast Retransmit

Après avoir reçu un petit nombre d'ACKs en double (duplicate acks) pour le même segment TCP (typiquement trois), l'émetteur infère qu'un paquet est perdu et retransmet le paquet sans attendre l'expiration d'un temporisateur de retransmission. On fait l'hypothèse en fait que s'il ne s'agit que de la réception de segments dans le mauvais ordre, il n'y aura qu'un ou deux ACKs en double avant que le segment réordonné ne soit traité ce qui provoquera la génération d'un nouvel ACK.

TCP Reno

TCP Reno est similaire à TCP Tahoe, excepté qu'il modifie l'algorithme de Fast Retransmit pour inclure le mécanisme de Fast Recovery. Le nouvel algorithme permet d'éviter que le "tuyau" soit vide après un Fast Retransmit évitant ainsi le besoin de démarrer le Slow-Start pour le remplir de nouveau après la perte d'un seul paquet.

On entre en phase de Fast Retransmit après avoir reçu un certain nombre (`tcprexmtthresh`) d'ACKs en double (typiquement trois). Les opérations suivantes sont alors exécutées:

- L'émetteur retransmet le dernier segment non acquitté, met ssthresh à la moitié de la taille de la fenêtre de congestion et réduit sa fenêtre de congestion de moitié.
- On utilise ensuite les ACKs en double pour rythmer l'émission des paquets TCP (ce qui est conforme à la philosophie du contrôle de congestion TCP selon laquelle les ACKs sont les pulsations du cœur du réseau). Plus précisement, la fenêtre utilisable (usable window) devient min(awin, cwnd + ndup) où awin est la taille de la fenêtre de transmission (c'est à dire la fenêtre maximale annoncée par le récepteur) et ndup est le nombre d'ACKs en double reçus depuis la réception du premier ACK indiquant la réception d'un paquet hors-séquence et est maintenu à zéro tant que le nombre d'ACKs en double n'a pas atteint le seuil `tcprexmtthresh`.
- Lorsque l'émetteur reçoit un ACK d'un nouveau paquet (appelé un recovery ACK), il sort du Fast Recovery en mettant ndup à zéro. Le mécanisme du Slow-Start n'est pas utilisé. Comme il s'agit ici de pertes aléatoires (qui ne sont peut être pas dues à la congestion), l'attitude optimiste du protocole respecte la philosophie du contrôle de congestion TCP. la

TCP New-Reno

TCP New-Reno modifie le comportement de TCP Reno lorsqu'on reçoit un ACK partiel càd un ACK qui acquitte au moins *le segment perdu* (celui qui nous a fait entrer en Fast Recovery) mais pas nécessairement tous les segments envoyés. Lors de la réception d'un tel ACK, TCP Reno quitte le mode de Fast Recovery ce qui pose un problème de performance lorsque plusieurs segments d'une même fenêtre sont perdus. TCP New-Reno ne quitte le mode de Fast Recovery que si l'ACK reçu acquitte *tous les segments envoyés*. A la réception d'un ACK partiel, New-Reno retransmet immédiatement le paquet suivant le dernier paquet acquitté dans cet ACK, diminue la taille de la fenêtre du nombre de paquets acquittés par cet ACK partiel et retransmet un paquet si c'est permis par la taille de cwnd.

Question 5: Réseaux avec borne de délai

Les composantes sont un ordonnanceur FQ dans les routeurs, un régulateur de trafic leaky bucket à l'entrée et un contrôle d'admission afin de refuser les flux si les contraintes en délai ne peuvent pas être respectées.

La borne du délai de transmission et de queuing dans le réseau dépend de la taille maximum de burst, de la bande passante allouée à la connexion dans le réseau et de la capacité des liens ainsi que de la taille maximale des paquets. D'après la formule donnée : si Pmax est négligeable alors le délai sera dû au burst (premier terme de côté droit de l'équation). Sinon, deux termes s'ajoutent : le premier est dû à l'attente dans les routeurs causés par la transmissions des paquets de la même connexion (il s'agit de paquets qui arriveraient à un nœud intermdiaire juste après le temps qu'il faut de façon qu'il devrait attendre la transmission d'un paquet de la même connexion; le dernier terme provient de l'attente due aux paquets des autres connexions.

Comme le délai de propagation est de 30 ms, il reste un budget délai de 120 ms. Pour respecter ce délai la réservation g devrait être telle que :

$D(i) \leq \sigma(i)/g + 2P_{max}/g + 3P_{max}/r$, d'où $g \geq (\sigma + 2P_{max})/(D - 3P_{max}/r)$.

Le poids de la connexion est : g/r. Le facteur de sur-réservation est : g/ρ.

Application numérique : g = 406.992 Kbps.

Pour $P_{max} = 1000$ le facteur de sur-réservation est de 4.

Pour $P_{max} = 100$ le facteur de sur-réservation est de 2.86.

Pour $P_{max} = 10000$ le facteur de sur-réservation est de 16.27.

Question 6: Multiplexage statistique

a) Le débit normalisé minimal devrait permettre d'écouler tous les flux en entrée, en considérant qu'il n'y a aucun chevauchement et il correspond à une cellule par unité de temps. Le débit maximal est en cas de chevauchement maximal et il est égal à 4 cellules par unité de temps.

b) Il y aura des trames avec des slots vides, ce qui montre l'inefficacité du multiplexage synchrone et le besoin du multiplexage asynchrone.

c) En organisant les transmissions de cellules de façon à éviter complètement le chevauchement, on peut transmettre les flux avec une capacité de sortie de 1 cellule. Il y aura des trames avec deux cellules de A.

A compléter...