Téléinformatique & Protocoles

TD 8 : Spécification incrémentale de protocoles

Corrigé

1 *protocole 1* : protocole utopique

1.1 Description du protocole

Sur le canal il n'y a ni erreur, ni perte, ni propagation, donc dès que la couche liaison émet une trame sur le canal, alors quasi-instantanément cette trame se retrouve du côté du récepteur et celui-ci est toujours prêt à la recevoir. Il ne lui reste plus qu'à délivrer le paquet d'information à la couche réseau.

1. Décrire les actions que le protocole doit réaliser
   Le récepteur ne fait qu'attendre les trames et à leur arrivée ne fait que délivrer le paquet d'information à la couche réseau.
   L'émetteur prend les paquets de la couche réseau, les met dans des trames les envoie les unes après les autres.
   Donc il n'y a nul besoin de contrôle de flux, ni de contrôle d'erreur, ni de temporisation, ni d'acquittement :
2. Écrire le protocole en pseudo-code.

```
type TypeEvenement = (ArriveeTrame);
   /* un seul type d'evenement */

procedure emetteur1;

var s: trame;        /* tampon pour trame en emission */
    tampon: paquet;

begin
   repeat
      Origine_Couche_Reseau(tampon);
         /* prendre ce qu'il y a a envoyer */

      s.info := tampon;
         /* on le copie dans s pour le transmettre */

      Vers_Couche_Physique(s);
   until FinDesTemps;
end; /*emetteur1*/


procedure recepteur1;

var r: trame;
    evenement: TypeEvenement;
```

```
begin
   repeat
      attendre(evenement);
         /*seul evenement possible : ArriveeTrame*/

      Origine_Coiuche_Physique(r);
         /* acquisition de la trame */

      Vers_Couche_Reseau(r.info);
         /* on passe les donnees a la couche reseau */

   until FinDesTemps;
end; /* recepteur1 */
```

2 *protocole 2* : envoyer et attendre

2.1 Se reposer les bonnes questions

Le récepteur ne peut plus recevoir les données à n'importe quel rythme. Donc il serait nécessaire et suffisant que le protocole effectue un contrôle de flux.

Le type de contrôle de flux proposé n'est pas de type XON, $XOFF$. Toutes les trames de données doivent être acquittées par une trame (quelconque) envoyée par le récepteur vers l'émetteur. Ainsi, quelques soient les vitesses de propagation de ces informations de contrôle, aucune trame ne peut être perdue pour cause de tampons pleins.

Attention : l'acquittement ne signifie pas "J'ai bien reçu la trame précédente" puisque de toute façon il n'y a jamais de pertes, mais : "Je suis prêt pour la prochaine trame."

2.2 Description du protocole

2.2.1 Décrire les actions que le protocole doit réaliser

Le seul type d'événement est l'arrivée d'une trame.

L'émetteur commence par chercher un paquet sur la couche réseau, le met dans une trame et l'envoie. Ensuite avant de continuer, il attend un événement. Dès que cet événement se produit, il recommence.

Le récepteur attend un événement. Dès que cet événement se produit, il va chercher la trame, lit les données, les transmet à la couche réseau et envoie une trame pour acquitter la trame reçue.

2.2.2 Écrire le protocole en pseudo-code

```
type TypeEvenement = (ArriveeTrame); /* Un seul type d'evenement */

procedure emetteur2;

var s: trame;
tampon : paquet;
evenement : TypeEvenement;

begin
   repeat
      Origine_Couche_Reseau(tampon}
         /* Acquisition du paquet venant de la couche reseau */

      s.info := tampon;
         /* copie dans s pour transmission */
```

```
      Vers_Couche_Physique(s); /* on envoie la trame */

      Attendre(evenement);     /* On attend un signal */
   until FinDesTemps;
end; /*emetteur2*/


procedure recepteur2;

var r,s: trame;
evenement : TypeEvenement;

begin
   repeat
      Attendre(evenement);     /* On attend un signal */
      Origine_Couche_Physique(r};
         /* Acquisition de la trame arrivee */

      Vers_Couche_Reseau(r.info);
         /* Transmission du paquet a la couche reseau */

      Vers_Couche_Physique(s); /* Envoi acquittement */

   until FinDesTemps;
end; /*recepteur2*/
```

3 *protocole 3* : une ligne avec perte

3.1 Se reposer les bonnes questions

3.1.1 Temporisateur

Puisqu'il y a pertes et erreurs, il faut un contrôle d'erreurs. Les trames erronnées sont rejetées et corrigées par retransmission.

Deux solutions :

Première solution :

Envoyer à l'émetteur une trame pour lui dire que la trame envoyée était erronnée. Mais comme cette même trame risque de se perdre, cette méthode n'est pas sure.

Autre solution :

avoir une temporisation. L'émetteur après avoir émis une trame attend pendant une durée spécifique. Si l'acquittement n'est pas arrivée avant l'expiration de cette durée, la trame sera considérée comme perdue. Elle est donc retransmise.

3.1.2 Numérotation des trames

Si ce qui est perdue est l'acquittement et non l'information, alors du point de vue du récepteur tout s'est bien déroulé. La trame a été bien reçue et il ne reste plus qu'à attendre la trame suivante.

Par contre du point de vue de l'émetteur, l'acquittement n'arrive pas, donc il faut retransmettre la trame.

Le problème est alors que la trame retransmise est considéré par le récepteur comme une nouvelle trame, et elle sera délivrée à la couche réseau.

Pour éviter ce problème on numérote les trames de données.

3.1.3 Numérotation jusqu'à combien ?

réponse : jusqu'à 2. Il faut pouvoir distinguer $I(k)$ et $I(k+1)$. Mais il n'est pas nécessaire de distinguer $I(k)$ et $I(k+2)$. Si le récepteur a reçu $I(k)$, alors les seules trames qui risquent de lui parvenir sont $I(k)$ ou $I(k+1)$. S'il reçoit $I(k+n)$ alors cela signifierait que l'émetteur a envoyé I(k+n) avant d'avoir reçu l'acquittement de I(k+1) : Impossible. Donc un seul bit suffit pour numéroter les trames.

3.2 Description du protocole

3.2.1 Décrire les actions que le protocole doit réaliser

Les types possibles d'événement sont l'arrivée d'une trame et l'expiration du temporisateur.

L'émetteur commence par chercher un paquet sur la couche réseau, le met dans une trame et l'envoie. Ensuite avant de continuer, il attend un événement.

Si cet événement est l'arrivée d'une trame alors la trame de donnée a été bien reçue et on peut passer à la trame suivante : on cherche un nouveau paquet à la couche réseau, on augmente de 1 le numéro de séquence de la trame et on recommence.

Si cet événement est l'expiration du timeout, alors il faut simplement retransmettre la trame précédente.

Le récepteur n'accepte plus toutes les trames correctes, mais seulement celles qui ont un numéro de séquence correcte.

Le récepteur attend un événement. Il ne peut être que l'arrivée d'une trame. Il vérifie le numéro de séquence de la trame. S'il est correcte, alors on peut délivrer le paquet d'information à la couche réseau, sinon, IL FAUT RE-EMETTRE L'ACQUITTEMENT ! ! ! Pourquoi ? parce que si le numéro de séquence n'est pas correcte, alors cela signifie que le dernier acquittement s'est perdue et que le récepteur a déjà reçu cette trame. Si on n'envoie pas d'acquittement, alors l'émetteur continuera à réémettre toujours la même trame.

3.2.2 Écrire le protocole en pseudo-code

```
const MaxSeq = 1;

type TypeEvenement = (ArriveeTrame, Erreur, Timeout)

procedure emetteur3;

var PTAE: NumSequence; /*numero de la prochaine trame a envoyer */
   s: trame;
   tampon: paquet;
   evenement: TypeEvenement;

begin
   ProchaineTrameAEnvoyer:=0; /* initialisation */
   Origine_Couche_Reseau(tampon); /* chargement du premier paquet */

   repeat
      s.info := tampon;
      s.seq  := PTAE;
      Vers_Couche_Physique(s); /* emission trame */
      StartTimer(s.seq);
      Attendre(evenement);

      if(evenement=ArriveeTrame) then
      begin
         Origine_Couche_Reseau(tampon);
```

```
         inc(PTAE);
      end

   until FinDesTemps;
end; /* emetteur3 */


procedure recepteur3;

var TA : NumSequence; /*numero de la trame attendue */
   r, s: trame;
   evenement: TypeEvenement;

begin

   TrameAttendue:=0;     /* initialisation */
   repeat
      Attendre(evenement);
      if(evenement=ArriveeTrame) then
      begin
         Origine_Couche_Physique(r);
         if(r.seq = TrameAttendue) then
         begin
            Vers_Couche_Reseau(r.info);
            inc(TA);
         end;
         Vers_Couche_Physique(s); /* acquittement */
      end;
   until FinDesTemps;
end; /* recepteur3 */
```

3.3 Questions sur le protocole 3

Est-ce qu'il est possible avec ces hypothèses, qu'une trame de données soit acquittée par un acquittement destiné à un autre trame ?

Réponse : Supposons que la trame $I(k)$ vient d'être émise. Que cette trame arrive en bon état ou non, le récepteur ne pourra pas envoyer d'acquittement pendant ce temps. L'émission de $I(k)$ fait suite, soit à l'arrivée d'un acquittement, soit à un timeout.

Dans le premier cas, l'acquittement qu'enverra le récepteur sera bien desinée à $I(k)$. Et dans le deuxième cas, la trame de donnée que nous envoyons n'est qu'une retransmission de la trame précédente. Ainsi, il n'est pas impossible que l'acquittement que le récepteur voudrait envoyer ait été destiné à la trame précédente, mais cette autre trame avait nécessairement le même contenu que $I(k)$. ET SURTOUT, l'acquittement ne peut être émis que lorsque $I(k)$ est arrivée à destination.

Donc il est impossible qu'en half-duplex, une trame soit acquittée par un acquittement destinée à une autre trame.

4 *protocole 3bis* : Un canal full duplex

4.1 Se reposer les bonnes questions

La figure 1 représente un scénario où le protocole 3 est pris en défaut avec un canal full-duplex.

Si par accident le récepteur met plus de temps que d'habitude à acquitter une trame, alors la temporisation de l'émetteur peut expirer avant l'arrive de l'acquittement.

Mais cet aquittement n'est pas perdue et à son arrivée l'émetteur le considère comme l'acquittement de la deuxième trame $I(0)$ qu'il a envoyée. Donc il envoie la trame $I(1)$. Seulement la seconde trame $I(0)$ envoyée par l'émetteur arrive aussi au récepteur et le récepteur réagit de la même manière envoyant un acquittement. Cette fois-ci l'acquittement acquitte la trame $I(1)$ alors que la trame $I(1)$ peut très bien se perdre.

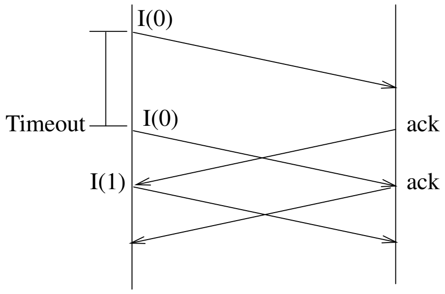

FIG. 1: Un scénario où un acquittement envoyé par le récepteur à l'intention de la trame $I(0)$ est interprété par l'émetteur comme un acquittement de la trame $I(1)$.

4.2 Description du protocole

4.2.1 Décrire les actions que le protocole doit réaliser

Les actions sont les mêmes, à l'exception du fait que le récepteur indique dans chaque acquittement, le numéro de séquence de la trame qu'il attend. D'autre part, l'émetteur n'ira chercher un nouveau paquet que si le numéro de l'acquittement est bien celui de la prochaine trame a envoyer.

4.2.2 Écrire le protocole en pseudo-code

```
const MaxSeq = 1;

type TypeEvenement = (ArriveeTrame, Erreur, Timeout)

procedure emetteur3bis;

var PTAE: NumSequence; /*numero de la prochaine trame a envoyer */
   r, s: trame;
   tampon: paquet;
   evenement: TypeEvenement;

begin

   ProchaineTrameAEnvoyer:=0; /* initialisation */
   Origine_Couche_Reseau(tampon); /* chargement du premier paquet */

   repeat
      s.info := tampon;
      s.seq  := PTAE;
      Vers_Couche_Physique(s);
      StartTimer(s.seq);
      Attendre(evenement);

      if(evenement=ArriveeTrame) then
```

```
      begin
         Origine_Couche_Phys(r);
         if(r.ack=PTAE+1) then
         begin
            Origine_Couche_Reseau(tampon);
            inc(PTAE);
         end
      end
   until FinDesTemps;
end; /* emetteur3bis */


procedure recepteur3;

var TA : NumSequence; /*numero de la trame attendue */
   r, s: trame;
   evenement: TypeEvenement;

begin

   TrameAttendue:=0;     /* initialisation */
   repeat
      Attendre(evenement);
      if(evenement=ArriveeTrame) then
      begin
         Origine_Couche_Physique(r);
         if(r.seq = TrameAttendue) then
         begin
            Vers_Couche_Reseau(r.info);
            inc(TA);
         end;
         s.ack := TA;
         Vers_Couche_Physique(s); /* acquittement */
      end;
   until FinDesTemps;
end; /* recepteur3bis */
```

5 *protocole 4* : Informations circulant dans les deux sens

5.1 Se reposer les bonnes questions

Le protocole3bis marche toujours dans ces conditions.

5.2 Description du protocole

Décrire les actions que le protocole doit réaliser. Écrire un protocole en pseudo-code qui soit identique pour les deux couches liaisons pairs.

5.3 Améliorations

A présent chaque extrémité envoie des données et des acquittements. On se propose de les envoyer sur une même trame. Réecrire le pseudocode en tenant compte de cette amélioration.

5.4 Scénario

Donner un scémario possible lorsqu'une des extrémités émet une trame commence l'émission avant que l'autre extrémité ait envoyé la première trame.

Donner un scénario possible lorsque les deux extrémités commencent par émettre une trame avant d'avoir reçu quoi que ce soit.

6 *protocole 5* : Propagation

6.1 Hypothèses

Les mêmes qu'auparavant sauf qu'à présent le temps de propagation d'une trame d'une extrémité à l'autre est considérable.

6.2 Se reposer les bonnes questions

Est-ce qu'il serait toujours possible d'utiliser le protocole 4 ?

En quoi le changement sur les hypothèses influe-t-il sur les réponses que nous avions données ?

6.3 Description du protocole

Décrire les actions que le protocole doit réaliser. Écrire un protocole en pseudo-code qui soit identique pour les deux couches liaisons pairs.

6.4 Question supplémentaire

Jusqu'à quelle valeur les trames doivent-elles être numérotées ?

7 *protocole 6* : Canal très bruité

7.1 Hypothèses

Les mêmes qu'auparavant sauf qu'à présent on suppose que le canal de transmission est très bruité. Concrètement, on supposera que dans un groupe de 5 trames (par exemple) au moins une des trames est perdue.

7.2 Se reposer les bonnes questions

Est-ce qu'il serait toujours possible d'utiliser le protocole 5 ?

En quoi le changement sur les hypothèses influe-t-il sur les réponses que nous avions données ?

7.3 Description du protocole

Décrire les actions que le protocole doit réaliser. Écrire un protocole en pseudo-code qui soit identique pour les deux couches liaisons pairs.

7.4 Question supplémentaire

Jusqu'à quelle valeur les trames doivent-elles être numérotées ?

Pour chaque numéro de séquence reçu, donner la manière dont le récepteur peut interpréter cela.